PROCEEDINGS

XVI Workshop de Visão Computacional

Uberlândia - MG

October 7-8, 2020

Proceedings of
XVI Workshop de Visão Computacional

Federal University of Uberlândia

ISSN 2175-6120

Publishers:
Bruno Augusto Nassif Travençolo
Thiago Pirola Ribeiro

Uberlândia, October 7-8, 2020

WVC 2020

Steering Committee Coordinator

Mauricio Cunha Escarpinati - Federal University of Uberlândia (UFU)

Steering Committee

Mauricio Cunha Escarpinati	UFU
Aparecido Nilceu Marana	UNESP
Hemerson Pistori	UCDB
Mauricio Marengoni	MACKENZIE
Paulo Sérgio Rodrigues	FEI

Local Organizing Committee

Maurício Cunha Escarpinati	UFU
André Ricardo Backes	UFU
Bruno Augusto Nassif Travençolo	UFU
Ana Claudia Martinez	UFU
Thiago Pirola Ribeiro	UFU
João Fernando Mari	UFV
Jocival Dantas Dias Jr.	UFU
Leandro Henrique Furtado Pinto Silva	UFU

Conference Chairs

Mauricio Marengoni	MACKENZIE
João Fernando Mari	UFV
Ana Cláudia Martinez	UFU
Marcelo Zanchetta do Nascimento	UFU
Paulo Sérgio Rodrigues	FEI

WVC 2020

Program Committee

Adilmar Dantas (UFU)
Adilson Gonzaga (USP)
Alexandre Levada (UFSCAR)
Ana Sequeira (INESC TEC)
Ana Martinez (UFU)
André Backes (UFU)
Antônio Apolinário (UFBA)
Aparecido Marana (UNESP)
Arthur Costa (USP)
Bruno Travençolo (UFU)
Carlos Thomaz (FEI)
César Pariente (UESC)
Cid Santos (UFSCAR)
Claudio Linhares (UFU)
Dali dos Santos (UFU)
Daniel Abdala (UFU)
Daniela de Sousa (UFU)
Danilo Eler (UNESP)
Deborah Fernandes (UFG)
Dibio Borges (UnB)
Evandro Rodrigues (USP)
Gastão Miranda Junior (UFS)
Guilherme Wachs (FEI)
Gustavo Borba (UTFPR)
Helio Pedrini (UNICAMP)
Hemerson Pistori (UCDB)
Iális de Paula (UFC)
Ines Domingues (ISEC)
Isabel Harb Manssour (PUCRS)
Jean Ponciano (UFU)
Joao Papa (UNESP)
Joao Mari (UFV)
Jocival Dias Junior (UFU)
José Castanho (UNESP)
Jurandy Almeida (UNIFESP)
Leandro Neves (UNESP)
Leandro Couto (UFU)
Leonardo Ferreira (FEI)
Leonardo Matos (UFS)
Lucas Oliveira (UFPR)
Luiz Antonio Neves (UFPR)
Marcelo Vieira (USP)
Marcelo Nascimento (UFU)
Marcos Piteri (UNESP)
Mauricio Galo (UNESP)
Mauricio Marengoni (MACKENZIE)
Maximiliam Luppe (USP)
Nelson Mascarenhas (UFSCAR)
Paulo Ambrosio (UESC)
Paulo Sergio Rodrigues (FEI)
Rafael Santos (INPE)
Reinaldo Bianchi (FEI)
Roberta Oliveira (UnB)
Rodrigo Vimieiro (USP)
Thaína Tosta (UFABC)
Thiago Ribeiro (UFU)
William Schwartz (UFMG)

Contents

Rain Gutter Detection in Aerial Images for *Aedes aegypti* Mosquito Prevention

Lucas Rossi[1], André R. Backes[1], Jefferson R. Souza[1]
[1]School of Computer Science, Federal University of Uberlândia, Brazil
arbackes@yahoo.com.br

***Abstract*—The detection of *Aedes aegypti* mosquito is essential in the prevention process of serious diseases such as dengue, yellow fever, chikungunya, and Zika virus. Common approaches consist of surveillance agents who need to enter residences to find and eliminate these outbreaks, but often they are unable to do this work due to the absence or resistance of the resident. This paper proposes an automatic system that uses aerial images obtained through a camera coupled from an Unmanned Aerial Vehicle (UAV) to identify rain gutters from a shed that may be mosquitoes' foci. We use Digital Image Processing (DIP) techniques to differentiate the objects that may or may not be those foci of the mosquito-breeding. The experimental results show that the system is capable of automatically detecting the appropriately mosquito-breeding location.**

***Keywords*—*Aedes aegypti*, aerial images, image processing.**

I. Introduction

In Brazil, in 2016, more than one million cases of the three central diseases transmitted by the *Aedes aegypti* mosquito (Dengue, Zika, and Chikungunya) were recorded. The number of reported deaths reached 794, among them, 629 per dengue, 159 per chikungunya and six per Zika [1]. In 2018 were 32,161 probable cases of dengue in the country, 132 cases of Dengue with signs of alarm and a confirmed death in the State of Paraíba. The regions with the highest incidence of probable cases are the regions: Southeast (40.2% of occurrences) and Center-West (32.5%). During this same period, 705 probable cases of Zika virus fever, 7,406 of Chikungunya fever were registered, one confirmed death and seven other deaths under investigation [2].

In Uberlândia, in the year 2016, up to the epidemiological week 50, 12,949 dengue cases were reported. In the year 2017, there was a decrease of 82% concerning the previous year, with 3,747 cases reported [3]. In 2018, 335 cases of dengue, five of Chikungunya and four of Zika were reported by epidemiological week [4].

The life cycle of *Aedes Aegypti* begins after the laying of eggs by a female on the wall of a breeding ground with water (eggs are not deposited directly in water). Such eggs can remain without hatching for a long time, are resistant to dryness and can last up to 450 days. After the egg hatch, the larval stage of the mosquito begins, the larva feeds mainly on the organic matter present in the breeding ground. After about five days the pupal phase begins, this period lasts on average three days, during which time the pupa remains on the surface of the water to facilitate the flight as an adult. It is during the adult phase that mosquito can transmit diseases to man [5].

According to a survey carried out by the Ministry of Health in 2013, 90% of mosquitoes are found in homes, and in 45 days a mosquito can contaminate up to 300 people [6]. Therefore, there are monitoring agents able for the houses to prevent the reproduction of the insect by removing and destroying objects that could become mosquito breeding sites. Some of the objects that can be *Aedes aegypti* breeding are: plant pots, dumpsters, plants that accumulate water, bottle caps, eggshell, cans, plastic bags, glass containers, disposable cups, lakes, waterfalls, water tanks, abandoned old tires, PET and glass bottles, shards of glass on the walls, buckets [7].

The contribution of this work is the detection of objects to be breeding sites of the *Aedes Aegypti* mosquito, which may be difficult to access for inspection agents or even for the population (rain gutters, Figure 1). The objective is to use the images captured by the camera coupled to the UAV to identify if the object or location is a possible focus of the mosquito, to carry out the preventive actions in the place and to reduce the chances of transmission of the diseases transmitted by it.

Fig. 1. Example of an image with rain gutters captured with a UAV.

The remainder of this paper is organized as follows: Section II discusses the related work to the *Aedes aegypti* mosquito detection; material and methods is shown in Section III; the proposed methodology is presented in Section IV; the results and analysis are shown in Section V; finally, Section VI concludes and suggests directions for future work.

II. Related Work

In [8], the authors offer a method for the identification of mosquito breeding sites using wireless networking and the removal of stagnant water through electromechanical pumping systems. The inactive water areas are identified and reported by users using a web-based portal. A vehicle was carrying a Global Positioning System (GPS), on-board Camera and a pumping system with a tank for removing the stagnant water. Finally, they removed stagnant water using a pumping system. The results show the effectiveness of the proposed approach.

In [9], it is shown an approach for detecting the presence of stagnant water bodies in images obtained in settings. Stagnant water can become sites for mosquitos to grow, such as *Aedes aegypti*. They present a method that can identify puddles in these images with about 88% successfully accuracy. The method is robust to image focus, making it a good choice for datasets produced by different kinds of cameras, including UAVs.

The work proposed by [10] uses crowdsensing techniques coupled with the medical professional's diagnosis of Zika to impute data to provide a location for Zika outbreaks. Results show that the approach has the potential to create impactful results. If adequately tested this system helps prevent Zika infections.

Dengue is one of the rapidly spreading and deadly diseases in Sri Lanka. A UAV was used to capture the mosquito breeding. It can inspect both accessible and inaccessible places to a human. [11] shows an approach to detect dengue mosquito via UAV images. The method captures the images of the water retention areas. Results produced on the field test were satisfactory of accuracy in identifying water retention areas.

Annually thousands of people die from dengue fever, chikungunya, and Zika. The majority of mosquito-breeding are bottles, tires, barrels, or any stagnant water. The work proposed by [12] shows a system to aid the mosquito-breeding habitats employing computer vision on aerial images. Initially, a dataset was created with video sequences from a UAV and the manual annotation in several scenarios. The features extracted from the images were HSV color space, histograms, and edge detection to train a random forest classifier. Results demonstrate that the classifier resulted in an accuracy higher than 99% in the test set. Then, the system is capable of automatically determining the GPS coordinates of the mosquito breeding location.

III. Material and Methods

A. Image acquisition

For image acquisition, we used DroneDeploy software to upload a map with a predefined route that must be covered by the UAV. The UAV flies over the route and takes pictures, which are uploaded to a computer after the flight and merged into an image containing the whole area observed. To obtain the aerial images, we used an aircraft Phantom 4 Pro. Aerial images were captured by an onboard camera of 4864×3648 pixel resolution and 72 dpi, flying at an altitude of 30 meters. The dataset consists of 207 images captured by the UAV.

B. Mathematical morphology

Proposed by Jean Serra in its Ph.D. thesis, mathematical morphology describes different techniques to process digital images using many concepts from set theory [13]. Mathematical morphology is usually applied to binary images, where each image is defined as a subset of a two-dimensional integer grid, $\mathbb{Z}^2$. Given an input image, we desire to process; mathematical morphology requires a second binary image of pre-defined shape called a structuring element. It is the structuring element that guides how the input image will be processed, analyzed and/or have extracted its geometrical structures. In the following paragraphs we describe basic mathematical morphology operations used in this work:

Dilation: Given an input binary image A and a structuring element B, this operation increases the area of the objects in A according to the shape of an element B, as shown in Equation 1. Depending of the shape and size of B, different objects in A may be fused into a single one.

$$A \oplus B = \{x \in Z^2 | c = a + b, a \in A \wedge b \in C\} \qquad (1)$$

Connected component labeling: this process groups image pixels based on their connectivity and similar intensity. Basically, for a binary image, this method scans the image and attributes the same label to all foreground pixels that are in some way connected with each other. Each label represents a different and separated structure in the original image.

C. HSV color space

In general, acquisition devices capture images in the RGB color space due to its simplicity and its ability to satisfactorily expresses the captured scene. Nevertheless, RGB space does not reproduce how humans interpret color, in particular, the luminance and the chrominance of the color. Depending on the application, HSV color space emerges as an alternative to represent color. Three components define HSV model: hue (H), the color component; saturation (S), the amount of gray; and value (V), the brightness or intensity of the color. This is a perceptual color model which mimics human color response, i.e., it models the color in a way that is closer to how human vision perceives color attributes [14].

D. Hough Transform

Developed by Paul Hough in 1962, Hough transform is an essential tool to detect parametric objects in digital images such as lines, circles, and ellipses. It is usually applied in a preprocessed image, for example, in the resulting image of a border detection method.

Basically, the Hough transform maps an image pixel into a parametric space organized as an n-dimensional accumulator. For the case of line detection, each pixel (x, y) of the object has a parametric line $y = a * x + b$ associated to it, where a is the slope, and b is the intercept.

Duda e Hart [15] showed that it is possible to fully represent these lines using polar coordinates, where each line is defined by its length, r, and orientation, θ, of the normal vector to the original line:

$$r = x * cos(\theta) + y * sin(\theta) \quad (2)$$

By using this Equation 2 each point of the original image is mapped into a sine curve in the polar space. Different aspects of the object in the original image result in different sine curves that intercept into a single point at the polar space, and each (r, θ) point in polar space represents a line in the original image. These properties enable us to detect the line equation of the original space by identifying the point where most sine curves intercept each other in the polar space.

IV. Proposed Methodology

In this section, we present the proposed methodology to detect rain gutter on shed roofs. Usually, it is possible to detect the rain gutter by detecting other materials accumulated in its structure, such as soil residue, leaves etc. To accomplish this task, we first convert the image from RGB to HSV color space. Then, applied a threshold over hue (H) and saturation (S) channels to extract objects that may indicate the presence of a rain gutter in the image. Due to the characteristics of the region, we used the following equation to convert the hue and saturation of the input image into a binary image B:

$$B_{i,j} = \begin{cases} 1 & \text{if } (H_{i,j} \geq 0.86 \vee H_{i,j} \leq 0.20) \wedge S_{i,j} \geq 0.20 \\ 0 & \text{otherwise} \end{cases}$$

where, H,$S \in [0, 1]$ and i, j represent a pixel in the image. These values were manually defined to select the more reddish shades in the image. In the sequence, we performed a dilation of the binary image B using a disk of radius $r = 10$ as structuring element. The dilation is necessary as the HSV segmentation may result in a disconnected rain gutter and, as we don't know the orientation of the gutter previously, the disk is the better option to connect nearby elements in all directions. Figure 2(a)-(c) shows the result of the segmentation and the dilation process.

The segmentation process may result in multiples elements. Since the rain gutter is a thin and elongated structure, characteristics such its area and its aspect ratio are useful to discriminate it from other objects. We used connected component labeling to identify each object in the dilated image. The elements are sorted according to its area value, i.e., the number of pixels contained in that object. Empirical analysis showed that the best candidates to rain gutter are objects whose area is smaller than 3% of the image. Starting from the largest to the smallest component, we selected the largest component whose area falls into this criteria. Figure 2(d) shows an example of the largest connected component in an image.

In the next step, we verify if the selected object is a rain gutter. To accomplish that we computed the bounding rectangle of the selected object and used the rectangle sides to calculate the object's aspect ratio and the rectangle area, as shown in Figure 2(e). We defined the aspect ratio as

$$ar = 100 * A/B \quad (3)$$

where A and B are, respectively, the smallest and largest sides of the rectangle. We consider any object as a rain gutter if the aspect ratio is equal to or greater than 21% and its area equals to or smaller than 10% of the area of the bounding rectangle. If the object fails into fitting these criteria, it is discarded, and the next object is evaluated. This process ends when we find an object fitting both criteria our when there are no other objects to test. Finally, we use the Hough transform to compute the line segment which corresponds to the object selected and use this information to highlight the rain gutter in the image, as shown in Figure 2(f).

V. Experimental Results

We applied the proposed methodology in a set of 207 images captured by the UAV. From these, only 56 images presented a rain gutter. Remaining images consisted of images of shed roofs without the presence of rain gutter or images of the region near the building, which consists basically of terrain. Figure 3 shows examples of the three situations.

Using our approach, we were able to detect rain gutters in 60 images from which 51 images had rain gutters that were effectively identified while 9 are the false positive results. In 5 images, our approach identified was not able to detect the existing rain gutters or it identified erroneous other structure. It is crucial to emphasize that some missed rain gutters are in images that contain only a small portion of the shed roof, so that the structure we aim to detect is extremely small, hinders the detection process. Some cases of the false positive are due to presence of larger structures that fit into the parameters defined to detect rain gutters.

One example of this situation is shown in Figure 4. This figure shows the presence of a wire-like structure in the roof shed which is falsely detected as a rain gutter. As one can see, this structure presents a length and aspect ratio similar to our target objects, which explains its detection. This situation indicates that a post-processing step may still be necessary to filter false candidates.

Figure 5 shows some examples of the correct detections. Figure 5(a) is the easiest case as the target structure is well isolated and it is the largest connected component detected by our approach. Figure 5(b) displays a large amount of terrain in the image. Initially, our approach considers this terrain as our target structure. However, since this object doesn't fit the established parameters of our algorithm, it is discarded, and other structures are evaluated until one fits (the small structure at the top right corner). Finally, Figure 5(c) shows a situation where both rain gutter and terrain are feasible candidates after the segmentation. In this particular case, the object which corresponds to terrain is discarded when we evaluate its bounding rectangle and notice its aspect ratio fails to our given criteria.

VI. Conclusions

We propose a simple detection of objects to be breeding sites of the *Aedes Aegypti* mosquito aiming to prevent several types of diseases, such as dengue fever, yellow fever, chikungunya,

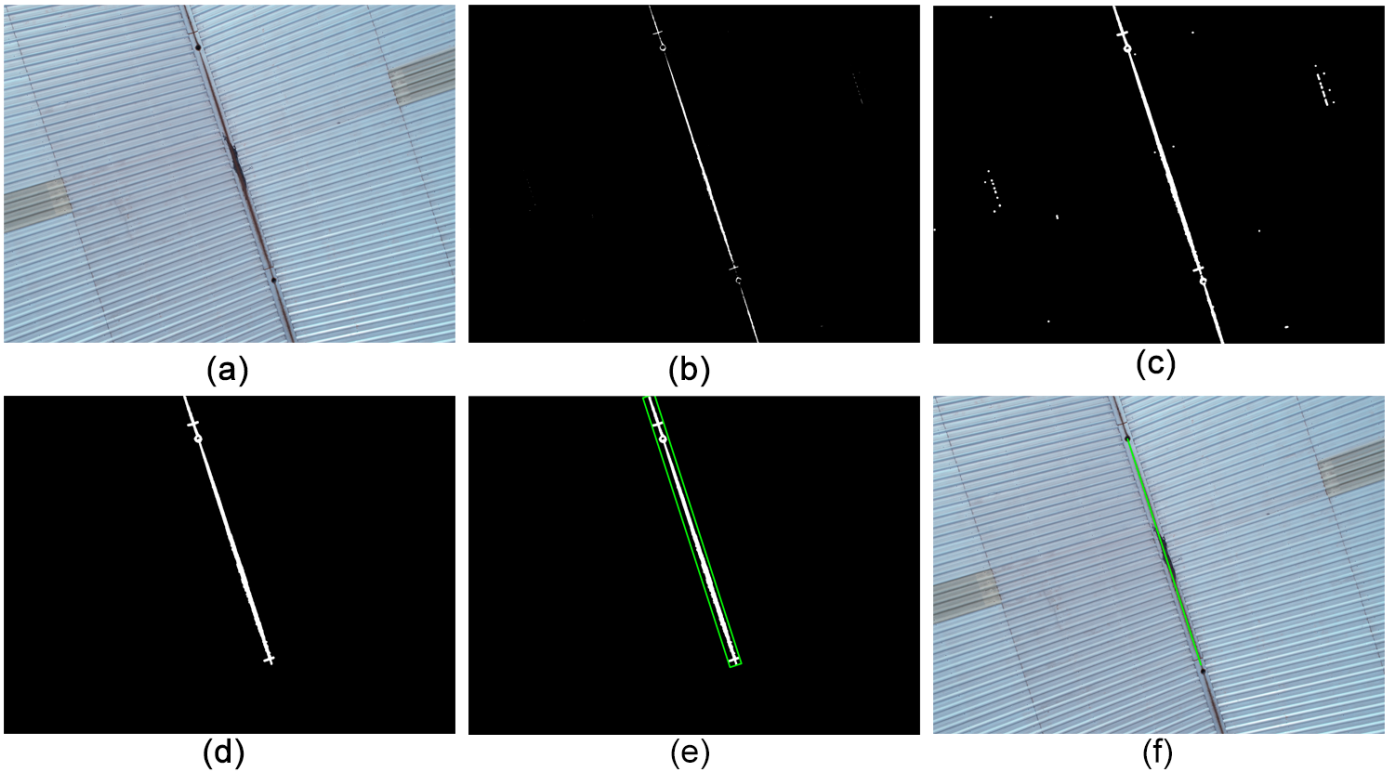


Fig. 2. Rain gutter candidate detection process: (a) Original image; (b) HSV segmentation; (c) Dilation of the segmented image; (d) Largest connected component; (e) Bounding rectangle; (f) Line detected using Hough transform.

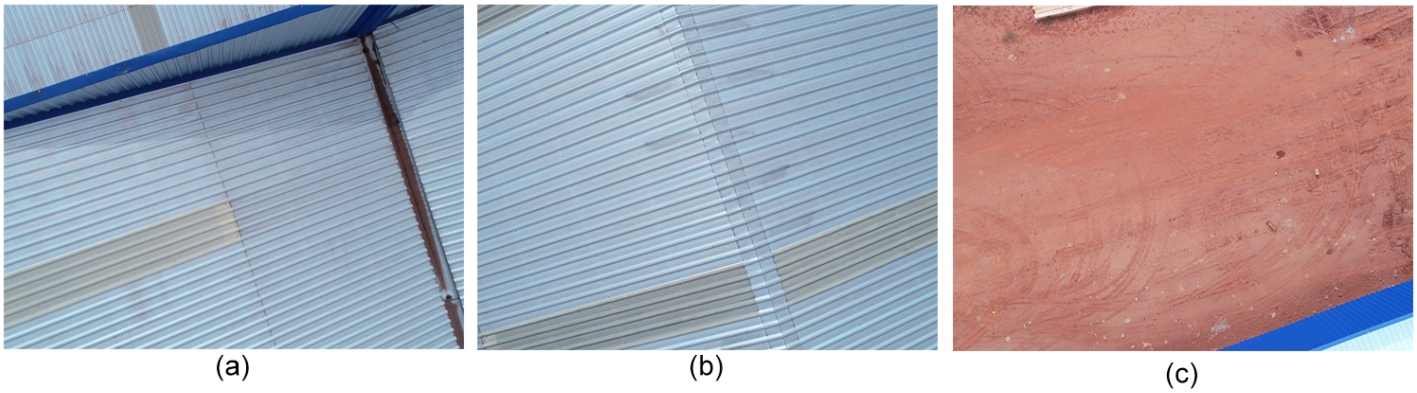


Fig. 3. Examples of the dataset: (a) shed roofs with rain gutter; (b) shed roofs without rain gutter; (c) terrain near building.

and Zika virus. The purpose is to use the images captured by the camera coupled to the UAV to identify if the location is a possible focus of the *Aedes Aegypti* mosquito, to carry out the preventive actions in the place and to reduce the chances of transmission of the diseases transmitted by it.

We believe the contributions made in this paper are an essential step towards the prevention of the *Aedes Aegypti* mosquito. Also, results demonstrate that the system is capable of automatically detecting the appropriately mosquito-breeding location. Lastly, false positive results can be handled on post-processing step useful to filter false candidates.

As future work, we will consider more objects to be explored aiming the prevention of the *Aedes Aegypti* mosquito. Objects as water tanks, swimming pool, abandoned old tires, and glass bottles. Furthermore, we also intend to apply machine learning techniques to compare with the proposed approach of this work.

Acknowledgment

The authors would like to acknowledge the Federal University of Uberlândia, FAPEMIG (Minas Gerais Research Foundation) under Grant #APQ-03437-15, CNPq (National Council for Scientific and Technological Development) under Grant #400699/2016-8, #302416/2015-3 and #301715/2018-1, and PROPP-UFU. This study was financed in part by the

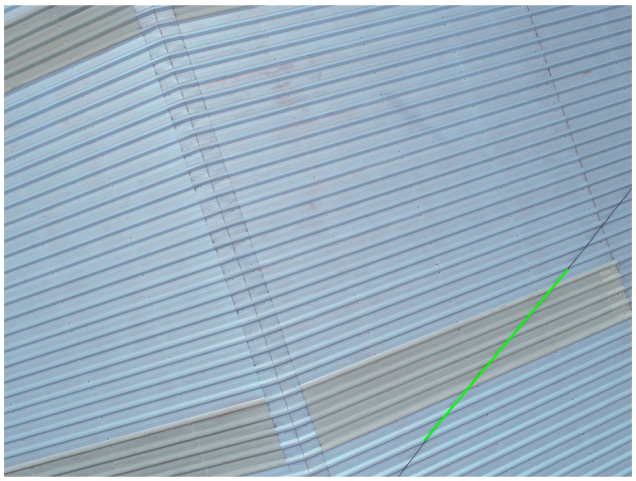


Fig. 4. Example of the false positive result.

Fig. 5. Examples of the rain gutters detected.

Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) - Finance Code 001.

REFERENCES

[1] A. L. Cavalcante, S. M. F. Brito, and A. S. Benzaken, "Monitoramento dos casos de dengue, febre de chikungunya e febre pelo vírus zika até a semana epidemiológica," 2018, [accessed Feb 13]. [Online]. Available: http://portalarquivos2.saude.gov.br/images/pdf/2018/janeiro/23/Boletim-2018-001-Dengue.pdf

[2] E. Duarte, G. D. S. Ferreira, and M. B. D. Turcato, "Monitoramento dos casos de dengue, febre de chikungunya e febre pelo vírus zika até a semana epidemiológica 7," 2018, [accessed Feb 13]. [Online]. Available: http://portalarquivos2.saude.gov.br/images/pdf/2018/marco/06/2018-008-Publicacao.pdf

[3] A. A. P. Neto, A. S. Souza, and J. H. Arruda, "Boletim de vigilância em saúde. boletim epidemiológico," 2018, [accessed Feb 13]. [Online]. Available: http://www.uberlandia.mg.gov.br/uploads/cms_b_arquivos/18763.pdf

[4] E. M. G. Paula, R. A. Oliveira, and G. A. Correa, "Boletim de vigilância em saúde," 2018, [accessed Feb 13]. [Online]. Available: http://www.uberlandia.mg.gov.br/uploads/cms_b_arquivos/18854.pdf

[5] V. S. Santos, "Ciclo de vida do aedes aegypti," 2018, [accessed Feb 13]. [Online]. Available: https://brasilescola.uol.com.br/animais/ciclo-vida-aedes-aegypti.htm

[6] A. Mendonça, "Brasil: 2 milhões de casos de doenças causadas pelo aedes aegypti," 2018, [accessed Feb 13]. [Online]. Available: https://www.tarobanews.com/noticias/ciencia-e-saude/brasil-2-milhoes-de-casos-de-doencas-causadas-pelo-aedes-aegypti-pMW21.html

[7] Brasil, "Nota a imprensa: Papel dos agentes comunitários de saúde," 2018, [accessed Feb 13]. [Online]. Available: http://combateaedes.saude.gov.br/pt/profissional-e-gestor/orientacoes/141-papel-dos-agentes-comunitarios-de-saude

[8] P. Anupa Elizabeth, S. M, P. Paulraj, S. Pandian, and B. Tyagi, "Identification and eradication of mosquito breeding sites using wireless networking and electromechanical technologies," 12 2014.

[9] M. Mehra, A. Bagri, X. Jiang, and J. Ortiz, "Image analysis for identifying mosquito breeding grounds," in *2016 IEEE International Conference on Sensing, Communication and Networking (SECON Workshops)*, June 2016, pp. 1–6.

[10] J. Livingston and R. Steele, "A crowdsensing algorithm for imputing zika outbreak location data," in *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*, Oct 2017, pp. 334–340.

[11] A. Amarasinghe, C. Suduwella, L. Niroshan, C. Elvitigala, K. De Zoysa, and C. Keppetiyagama, "Suppressing dengue via a drone system," in *2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer)*, Sep. 2017, pp. 1–7.

[12] T. Dias, V. Alves, H. Alves, L. Pinheiro, R. Pontes, G. Araujo, A. Lima, and T. Prego, "Autonomous detection of mosquito-breeding habitats using an unmanned aerial vehicle," in *2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education*, Nov 2018, pp. 351–356.

[13] J. Serra, *Image Analysis and Mathematical Morphology*. Book, 1982.

[14] R. C. Gonzalez and R. C. Woods, *Processamento Digital de Imagens*, 3rd ed. Pearson, 2010.

[15] R. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *CACM*, vol. 15, pp. 11–15, 1972.

Impacts of Color Space Transformations on Dysplastic Nuclei Segmentation Using CNN

Dalí F. D. dos Santos*, Adriano B. Silva, Paulo R. de Faria, Bruno A. N. Travençolo, Marcelo Z. do Nascimento
Federal University of Uberlândia, Uberlândia, Minas Gerais, Brazil
*dalifreire@gmail.com

***Abstract*—Oral epithelial dysplasia is a common precancerous lesion type that can be graded as mild, moderate and severe. Although not all oral epithelial dysplasia become cancer over time, this premalignant condition has a significant rate of progressing to cancer and the early treatment has been shown to be considerably more successful. The diagnosis and distinctions between mild, moderate, and severe grades are made by pathologists through a complex and time-consuming process where some cytological features, including nuclear shape, are analysed. The use of computer-aided diagnosis can be applied as a tool to aid and enhance the pathologist decisions. Recently, deep learning based methods are earning more and more attention and have been successfully applied to nuclei segmentation problems in several scenarios. In this paper, we evaluated the impact of different color spaces transformations for automated nuclei segmentation on histological images of oral dysplastic tissues using fully convolutional neural networks (CNN). The CNN were trained using different color spaces from a dataset of tongue images from mice diagnosed with oral epithelial dysplasia. The CIE L*a*b* color space transformation achieved the best averaged accuracy over all analyzed color space configurations (88.2%). The results show that the chrominance information, or the color values, does not play the most significant role for nuclei segmentation purpose on a mice tongue histopathological images dataset.**

***Index Terms*—CNN, deep learning, dysplasia, nuclei segmentation, color spaces**

I. Introduction

Cancer can be defined as a group of non-communicable diseases (NCD) that can start almost anywhere in the body. The disease is caused when cells begin to divide uncontrollably with potential to invade other parts of the body and/or spread to other organs and surrounding tissues. According to the World Health Organization (WHO), cancer is the second leading cause of death globally, accounting for 18.1 million cases and 9.6 million deaths worldwide in 2018 [1]. Projections from the Instituto Nacional do Cancer (INCA) show that in 2020, 686 thousand new cancer cases will be registered in Brazil and, from those cases, 15 thousand will be oral cavity-derived cancer [2].

Tumors of the oral cavity can be grouped into two broad categories: benign and malignant tumor (cancer). Benign tumors are not considered cancer, as they do not invade other tissues and do not spread to other parts of the body. Precancerous conditions, on the other hand, are still harmless lesions, but some of those precancerous conditions can give rise to cancer over time.

Dysplasia is an important precancerous condition characterized by the presence of abnormal cells in the oral mucosa. The histological evaluation, i.e. the study of tissue samples of affected region under the microscope, remains the most reliable way for diagnosing and grade oral epithelial dysplasia [3], [4]. Despite being referred as gold-standard in cancer diagnosis, the histological evaluation is plagued by inter- and intra-observer variability problem. This difficulty requires experienced pathologists and is an expensive, highly skilled and very time-consuming process.

Fast scanners can be used to capture tissue samples into digital images to obtain the so-called digital histological images, allowing to view tissue samples on computer rather than through a microscope. Digitized histological samples can be analyzed by computational image processing techniques to aid and enhance the pathologist decision making, minimizing human interventions, discovering measurable and traceable clinical information, providing reliable prognostic factors and eliminating the inter- and intra-observer variability.

There are several potential applications of digital pathology, such as nuclei detection and segmentation that are critical prerequisite steps for diagnosing and grading dysplasia in image-based computer-aided diagnosis (CAD). Nuclei extracted features are critical for evaluating the existence of diseases and its severity. Furthermore, nuclei commonly appear in overlapping clusters, have heterogeneous aspects and remain a challenging problem, which keeps nuclei segmentation methods under investigation [5]. Particularly, deep learning techniques, such as convolutional neural networks (CNN), has been successfully applied in medical and biological researches [6].

As the color is one of the most dominant and visually distinguishable visual properties, color variations could play a high influence on automated analysis of histological images. This paper evaluates the impact of different color space transformations applied to our previously proposed method for automated nuclei segmentation on dysplastic oral tissue histological images using fully convolutional neural networks [7]. For this, CNN models were trained using different color spaces from a dataset of tongue images from epithelial dysplasia-harboring mice.

The rest of the paper is organized as follows. The next

This work was supported by the National Council for Scientific and Technological Development - CNPq (Grant 304848/2018-2) and the State of Minas Gerais Research Foundation - FAPEMIG (Grant APQ-00578-18).

section describes some important background concepts. Section III presents the experimental evaluation and the Section IV presents the main results achieved and its discussion. Finally, the Section V concludes the paper and presents further work directions.

II. Background Review

A. Color Spaces

The purpose of a color space is to facilitate the specification of colors in some standard providing a coordinate system and a subspace in which each color is represented by a single point. There have been numerous different color spaces in use today. Some of these color spaces are ideally suited for hardware implementations and others for the way that humans describe and interpret colors [8]–[10].

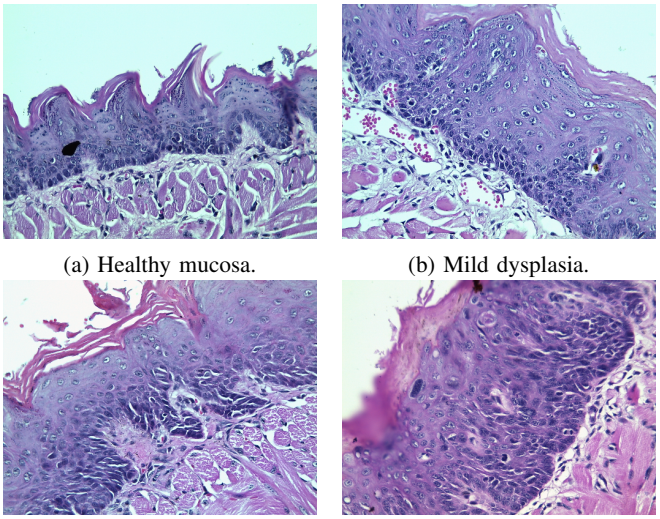
In this paper, we focused on using RGB, HSV and CIE L*a*b* color spaces in hematoxylin-eosin (H&E) stained histological images for nuclei segmentation purposes:

- RGB (red, green, blue) color space defines each color as a combination of the three primary spectral components: red, green, and blue. The RGB color space is the hardware-oriented color space most widely used for a broad class of video cameras and color monitors.
- HSV (hue, saturation, value/brightness) color space is a nonlinear transformation of the RGB that describes the pure color (hue) in terms of gray presented in each color (saturation) and how bright the color is (value). The HSV color space corresponds closely with the way humans describe and interpret color.
- CIE L*a*b was defined by the International Commission on Illumination (CIE) and is also based on human perception. The L* channel indicates lightness and a* and b* channels indicate chromaticity directions: a* indicates the color value between green and red and b* indicates the color value between blue and yellow.

B. Histological Images Dataset

The histological images dataset was built using H&E-stained tongue slides extracted from 30 mice previously diagnosed with oral epithelial dysplasia. The images were digitized using a Leica DM500 light microscope with original magnification of 400×. A total of 66 images were scanned and saved in TIFF format using the RGB color space and resolution of 2048×1536 pixels. A experienced pathologist used the criteria described by Lumerman et al. [11] to classify each image into four predominant classes: healthy tissue, mild, moderate or severe dysplasia.

The digitized images were then cropped into regions of interest (ROI) with size of 448×256 pixels, totalling 120 ROI images – 30 ROI images for each class. Examples of the produced histological images are shown in Fig. 1 and examples of the extracted ROI images are shown in the first column in Fig. 3 (3a, 3e, 3i, 3m).



(a) Healthy mucosa. (b) Mild dysplasia. (c) Moderate dysplasia. (d) Severe dysplasia.

Fig. 1: H&E stained histological images of mice oral epithelial tissues.

C. Automated Nuclei Segmentation Using CNN

The method used for nuclei segmentation in oral tissue histological images was originally proposed by dos Santos et al. [7]. The proposed CNN architecture, which was slightly modified to support images with one or three color channels as input, is depicted in Fig. 2.

Deep learning segmentation models requires a large number of samples and their corresponding segmentation masks to be properly trained. Since producing these image samples is a very hard and time consuming-task, data augmentation becomes an essential technique to overcome this general problem of scarcity of available training samples [13]. Six different image transformation techniques were used together as data augmentation: horizontal/vertical flip, rotation, elastic transformation, grid distortion and optical distortion. Some examples of data augmentation applied to the ROI images and their corresponding targets are illustrated in the most right two columns in Fig. 3 (3c, 3d, 3g, 3h, 3k, 3l, 3o, 3p).

In the training step, we directly apply the histological ROI images (e.g., Figs. 3a, 3c, 3e, 3g, 3i, 3k, 3m and 3o) and their corresponding masks (e.g., Figs. 3b, 3d, 3f, 3h, 3j, 3l, 3n and 3p) to the deep neural network to train the model. After the last convolution step be performed, the Otsu [14] threshold is applied to binarize the resulting mask image that predicts the nuclei for the input ROI image.

In order to investigate the influence of color space transformations on the evaluated nuclei segmentation method, we transformed the 120 ROI images from the original training and test sets into three different color spaces (RGB, HSV, and CIE L*a*b*). We retrained the nuclei segmentation model using the grayscale image and each color space and their respective individual channels separately, resulting in 13 new training datasets (thirteen sets of 96 images) and performed evaluations

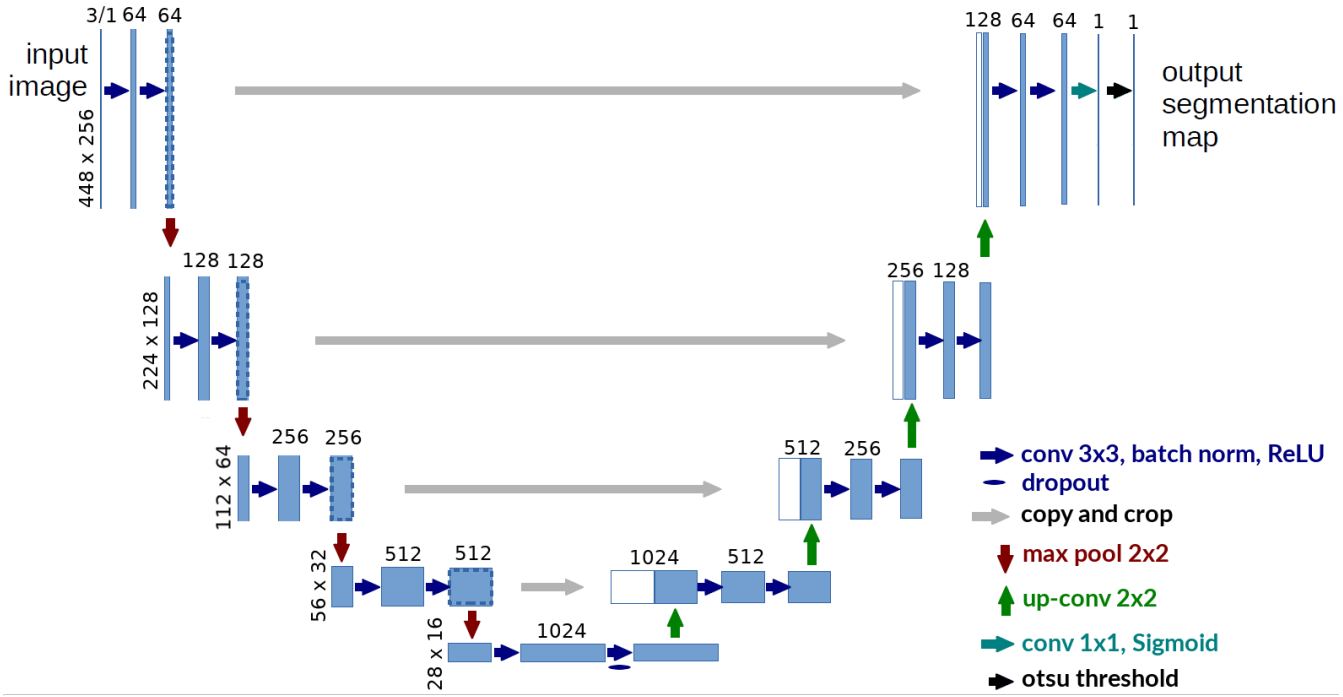


Fig. 2: Adaptation of the CNN architecture proposed in [7], which is based on U-Net [12] model. The number of convolutional feature channels and the height×width of images for each layer are denoted on their corresponding boxes.

using the test set (24 images transformed thirteen times for each corresponding color space configuration).

The nuclei segmentation model was implemented using the PyTorch framework [15]. The models were trained using a desktop computer (Intel Core i7 3.4GHz×8 processor, 32 GB memory, 1TB SSD) equipped with GeForce GTX 1050 Ti graphic card and Ubuntu 20.04 operational system. The elapsed time to train the models with 500 epochs for each color space transformation was about 400 minutes. After training, the elapsed time to process an input image was about 0.3 seconds. To provide better understanding and make this work as reproducible as possible, the source code is publicly available at: https://github.com/dalifreire/dysplastic_oral_tissues_segmentation.

III. Experimental Evaluation

We performed experimental evaluations using thirteen configurations for the 24 ROI images from the test subset: Grayscale image, RGB, only the R channel from RGB, only the G channel from RGB, only the B channel from RGB, HSV, only the H channel from HSV, only the S channel from HSV, only the V channel from HSV, CIE L*a*b*, only the L* channel from CIE L*a*b*, only the a* channel from CIE L*a*b* and only the b* channel from CIE L*a*b*.

The influence of different color space transformations on the automated nuclei segmentation using fully convolutional neural network [7] was investigated. The automated nuclei segmentation results were compared quantitatively and qualitatively (visually) with the nuclei manually segmented by the specialist. The segmentation performance was measured calculating the overlapping regions of the resulting automated segmented image and the regions of a reference image segmented by the specialist. The average performance was measured by six of the most commonly used quantitative criteria with respect to pixel classification (nuclei or non-nuclei). Accuracy is defined as:

$$Accuracy = \frac{\text{TP + TN}}{\text{TP + TN + FP + FN}}. \quad (1)$$

Precision indicates if the segmentation results bring only nuclei areas and does not bring any non-nuclei areas:

$$Precision = \frac{\text{TP}}{\text{TP + FP}}. \quad (2)$$

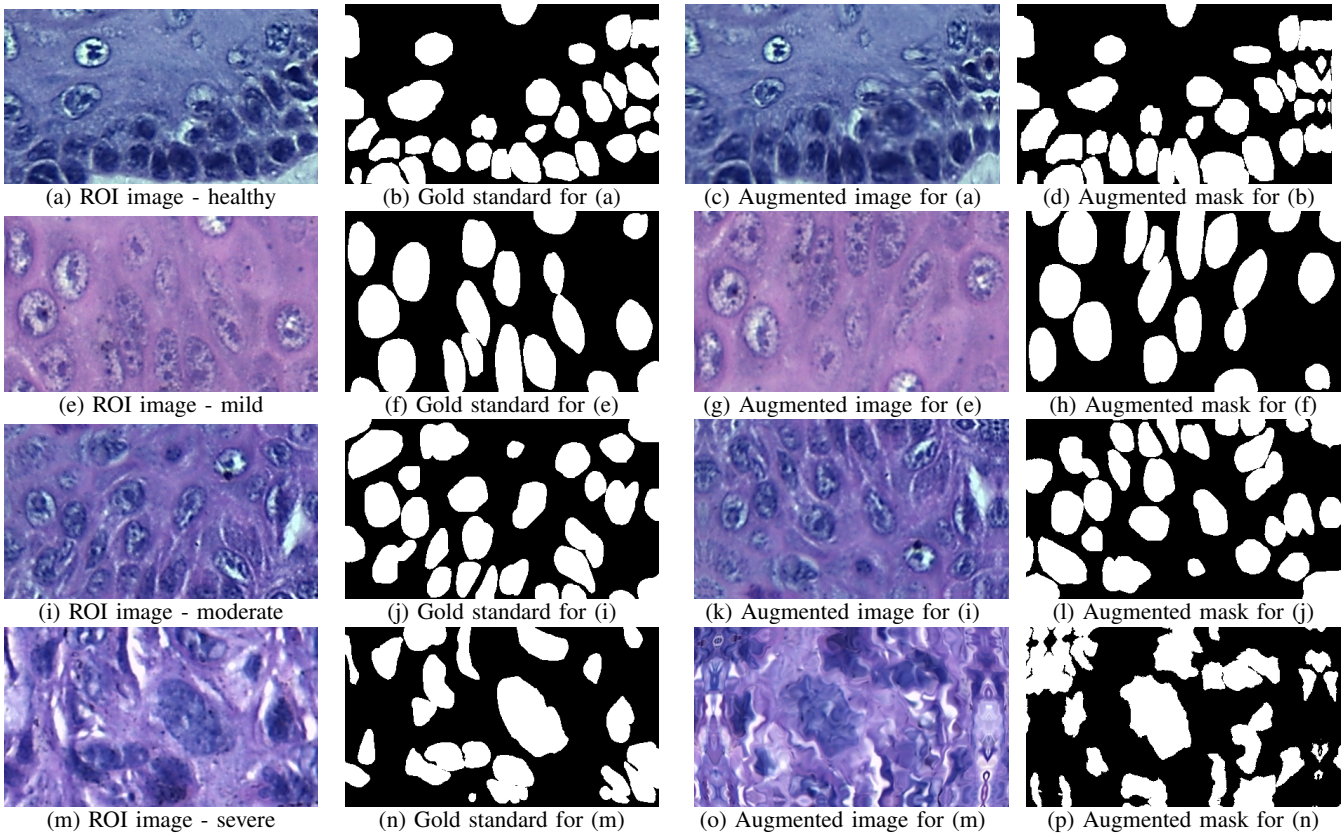
Sensitivity (or Recall) indicates if the segmentation results bring all the nuclei areas and is defined as:

$$Sensitivity = \frac{TP}{TP + FN}. \quad (3)$$

F1 score (or Dice Coefficient) is the harmonic mean of the precision and recall and can be defined as:

$$F1 = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}. \quad (4)$$

Jaccard index emphasizes similarity between gold-standard and segmentation results for both nuclei and non-nuclei areas and is defined as:



(a) ROI image - healthy (b) Gold standard for (a) (c) Augmented image for (a) (d) Augmented mask for (b)
(e) ROI image - mild (f) Gold standard for (e) (g) Augmented image for (e) (h) Augmented mask for (f)
(i) ROI image - moderate (j) Gold standard for (i) (k) Augmented image for (i) (l) Augmented mask for (j)
(m) ROI image - severe (n) Gold standard for (m) (o) Augmented image for (m) (p) Augmented mask for (n)

Fig. 3: ROI images from healthy and epithelial dysplasia-harboring tongue mice: 1st row hows healthy mucosa class; 2nd row shows mild dysplasia class; 3rth row shows moderate dysplasia class and 4rth row shows severe dysplasia class.

$$Jaccard = \frac{TP}{TP + FP + FN}. \quad (5)$$

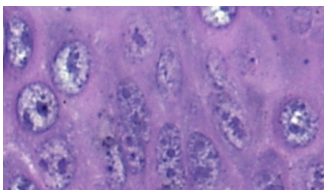
Specificity measures the proportion of non-nuclei areas correctly identified and is defined as:

$$Specificity = \frac{TN}{TN + FP}, \quad (6)$$

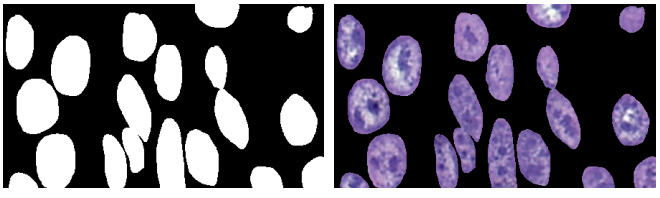
where *TP* means true positives (the amount of correctly detected pixels), *TN* means true negatives (the amount of correctly undetected pixels), *FP* means false positives (number of incorrectly detected pixels) and *FN* means false negatives (the number of incorrectly undetected pixels).

IV. Results

In this section qualitative and quantitative results are shown. Fig. 4 shows one selected image from the dataset to serve as a reference for the qualitative analysis. The original image is shown in Fig. 4a, its corresponding nuclei mask manually delimited by the specialist is shown in Fig. 4b, and the segmentation result using the masks manually delimited by the specialist – referenced as gold-standard – is shown in Fig. 4c. Fig. 5 reveals the results obtained by the automated segmentation method for each configuration on RGB color space and the Grayscale image. Fig. 6 depicts the results for HSV color space transformation and its respective individual channels. Fig. 7 shows the results obtained by CIE L*a*b* color space transformation and its respective individual channels.



(a) Original input ROI image.



(b) Manually segmented nuclei. (c) Nuclei segmentation result.

Fig. 4: Gold standard segmentation.

Regarding quantitative analysis, the Table I summarizes the average results for each quantitative measure in the test subset. The same 24 ROI images – test subset transformed thirteen

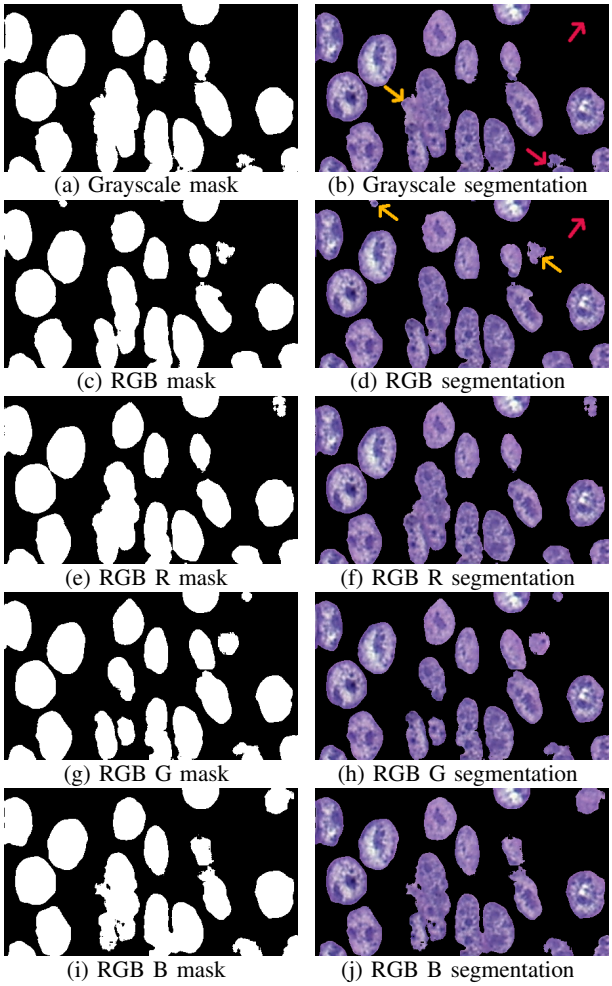


Fig. 5: Qualitative analysis for RGB color space segmentation: first column (a, c, e, g, i) shows nuclei masks obtained; second column (b, d, f, h, j) shows the final segmentation results – red arrows indicate some false negative areas; yellow arrows indicate some false positive areas.

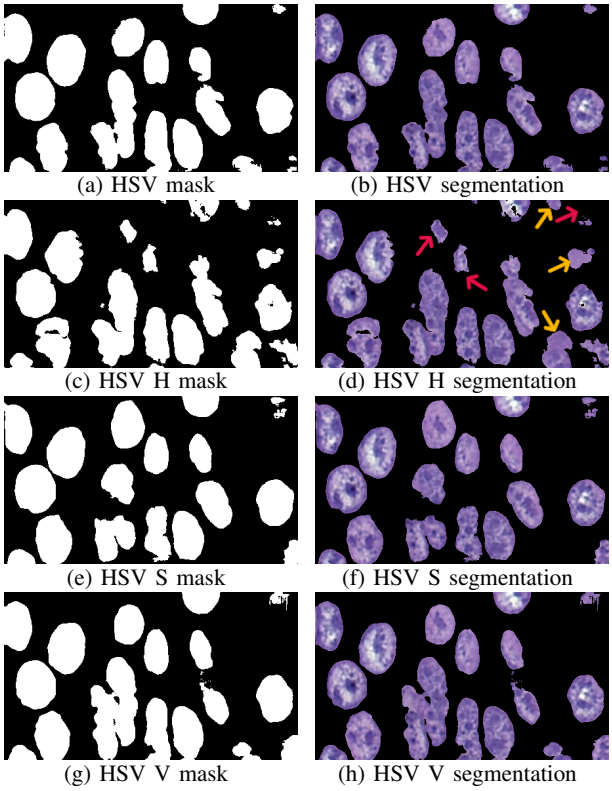


Fig. 6: Qualitative analysis for HSV color space segmentation: first column (a, c, e, g) shows nuclei masks obtained; second column (b, d, f, h) shows the final segmentation results – red arrows indicate some false negative areas; yellow arrows indicate some false positive areas.

times for each corresponding color space configuration – were employed to evaluate all configurations.

As we can see in Figs. 5, 6 and 7 and Table I, the original RGB and the CIE L*a*b* color spaces present the best results, outperforming all the configurations. It is important to note that despite visually presenting good results and contours close to the gold-standard, as indicated in the images by the red and yellow arrows, all configurations present some false negative and false positive regions.

The worst results were presented by the CIE a* and HSV H color space configurations. It is worth noting that these results indicate that the chrominance information alone, or the color values, does not play the most significant role for nuclei segmentation purpose on oral epithelial dysplasia-harboring tongue mice image datasets. The color space channels that indicate lightness information (Grayscale, CIE L*, HSV S/V) performed very close to the best configurations, revealing that lightness information plays a pivotal role for nuclei segmentation purposes (note also that Grayscale and L* channel are computed from a very similar equation from the RGB color space).

V. Conclusions

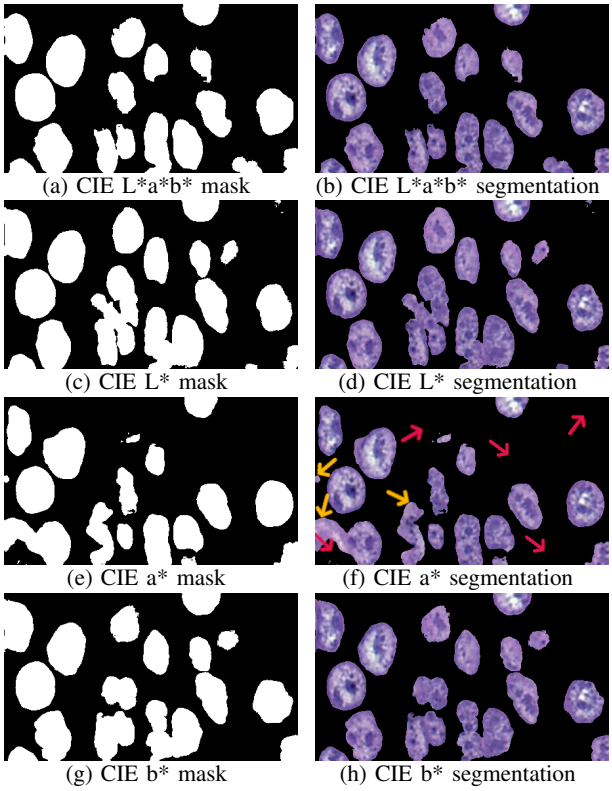
In this paper we evaluated the impact of different color space transformations applied to H&E-stained histological images for nuclei segmentation purposes. A fully convolutional neural networks model for automated nuclei segmentation was trained and run using thirteen different color space configurations of tongue mice-derived epithelial dysplasia image datasets.

Experimental results revealed that the chrominance information does not play the most significant role for nuclei segmentation purposes in H&E-stained histological images. Furthermore, the results indicates that most significant role for nuclei segmentation purposes may be played by the lightness information contained in the color spaces.

The dataset employed in this study has a reduced number of images and, in future works, the number of images will be expanded and images of human oral tissues will also be employed.

Color Space	Accuracy	Precision	F1 / Dice	Jaccard	Sensitivity / Recall	Specificity
Grayscale	0.862	0.769	0.795	0.667	0.842	0.877
RGB	0.879	0.793	**0.820**	**0.699**	**0.860**	0.891
R channel from RGB	0.874	0.788	0.811	0.689	0.848	0.893
G channel from RGB	0.860	0.794	0.780	0.647	0.782	0.908
B channel from RGB	0.862	0.775	0.794	0.666	0.832	0.885
HSV	0.881	0.807	0.819	0.698	0.844	0.904
H channel from HSV	0.784	0.691	0.659	0.503	0.648	0.858
S channel from HSV	0.842	0.752	0.765	0.628	0.801	0.871
V channel from HSV	0.863	0.783	0.792	0.663	0.817	0.895
Lab	**0.882**	**0.853**	0.814	0.689	0.788	**0.937**
L channel from Lab	0.865	0.775	0.797	0.670	0.835	0.886
a channel from Lab	0.758	0.637	0.648	0.490	0.679	0.803
b channel from Lab	0.819	0.711	0.738	0.590	0.790	0.841

TABLE I: The average quantitative results by each color space transformation applied on the test set (24 images).



(a) CIE L*a*b* mask (b) CIE L*a*b* segmentation (c) CIE L* mask (d) CIE L* segmentation (e) CIE a* mask (f) CIE a* segmentation (g) CIE b* mask (h) CIE b* segmentation

Fig. 7: Qualitative analysis for CIE L*a*b* color space segmentation: first column (a, c, e, g) shows nuclei masks obtained; second column (b, d, f, h) shows the final segmentation results – red arrows indicate some false negative areas; yellow arrows indicate some false positive areas.

REFERENCES

[1] F. Bray, J. Ferlay, I. Soerjomataram, R. Siegel, L. Torre, and A. Jemal, "Global cancer statistics 2018: Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries: Global cancer statistics 2018," *CA: A Cancer Journal for Clinicians*, vol. 68, 09 2018. 1

[2] INCA. (2020) Instituto nacional de câncer josé alencar gomes da silva (inca). [Online]. Available: https://www.inca.gov.br/tipos-de-cancer/cancer-de-boca 1

[3] J. Baik, Q. Ye, L. Zhang, C. Poh, M. Rosin, C. Macaulay, and M. Guillaud, "Automated classification of oral premalignant lesions using image cytometry and random forests-based algorithms," *Cellular Oncology*, vol. 37, p. 193 – 202, 2014. [Online]. Available: https://doi.org/10.1007/s13402-014-0172-x 1

[4] D. K. Das, S. Bose, A. K. Maiti, B. Mitra, G. Mukherjee, and P. K. Dutta, "Automatic identification of clinically relevant regions from oral tissue histological images for oral squamous cell carcinoma diagnosis," *Tissue and Cell*, vol. 53, pp. 111 – 119, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0040816618301137 1

[5] H. Irshad, A. Veillard, L. Roux, and D. Racoceanu, "Methods for nuclei detection, segmentation and classification in digital histopathology: A review current status and future potential," *IEEE reviews in biomedical engineering*, vol. 7, pp. 97–114, 05 2014. 1

[6] F. Xing, Y. Xie, H. Su, F. Liu, and L. Yang, "Deep learning in microscopy image analysis: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 10, pp. 4550–4568, 2018. 1

[7] D. F. D. dos Santos, T. A. A. Tosta, A. B. Silva, P. R. de Faria, B. A. N. Travençolo, and M. Z. do Nascimento, "Automated nuclei segmentation on dysplastic oral tissues using CNN," in *Proceedings of the 2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, July 2020, pp. 45–50. 1, 2, 3

[8] A. R. Robertson, "The CIE 1976 color-difference formulae," in *Color Research & Application*, vol. 2, 1977, pp. 7–11. 2

[9] R. C. Gonzalez and R. E. Woods, *Digital Image Processing (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006. 2

[10] M. M. Fernández-Carrobles, G. Bueno, O. Déniz, J. Salido, M. García-Rojo, and L. González-López, "Influence of texture and colour in breast tma classification," *PloS one*, vol. 10, 2015. 2

[11] H. Lumerman, P. Freedman, and S. Kerpel, "Oral epithelial dysplasia and the development of invasive squamous cell carcinoma," *Oral Surgery, Oral Medicine, Oral Pathology, Oral Radiology, and Endodontology*, vol. 79, no. 3, pp. 321 – 329, 1995. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1079210405802264 2

[12] T. Falk, D. Mai, R. Bensch, Özgün Çiçek, A. Abdulkadir, Y. Marrakchi, A. Böhm, J. Deubner, Z. Jäckel, K. Seiwald, A. Dovzhenko, O. Tietz, C. D. Bosco, S. Walsh, D. Saltukoglu, T. L. Tay, M. Prinz, K. Palme, M. Simons, I. Diester, T. Brox, and O. Ronneberger, "U-net – deep learning for cell counting, detection, and morphometry," *Nat. Methods*, vol. 16, pp. 67–70, 2019. 3

[13] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, Jul 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0197-0 2

[14] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, Jan 1979. 2

[15] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS Autodiff Workshop*, 2017. 3

RECOGNITION OF SOYBEAN DISEASES USING MACHINE LEARNING TECHNIQUES BASED ON SEGMENTATION OF IMAGES CAPTURED BY UAVS

Gercina Gonçalves da Silva
Administração/CPAQ/UFMS
Aquidauana-MS, Brasil
gercina.silva@ufms.br

Alessandro Santos Ferreira
Faculdade de Computação/UFMS
Campo Grande-MS, Brasil
asf2005kn@hotmail.com

Denilson de Oliveira Guilherme
Universidade Católica Dom Bosco
Campo Grande-MS, Brasil
denilsond@gmail.com

José Fernando Jurca Grigolli
Fundação MS
Campo Grande-MS, Brasil
fernando@fundacaoms.org.br

Vanessa Aparecida de Moraes Weber
Universidade Católica Dom Bosco (UCDB) e Universidade Estadual de Mato Grosso do Sul (UEMS)
Campo Grande-MS, Brasil
vamoraes@gmail.com

Hemerson Pistori
Universidade Católica Dom Bosco
Campo Grande-MS, Brasil
hpistori@gmail.com

Abstract - Soybean is an important product for the Brazilian economy, however it has factors that can limit its productive income, like the diseases that are generally difficult to control. Thus, this article aims to use a computer program to recognize diseases in images obtained by a UAV in a soybean plantation. The program is based on computer vision and machine learning, using the SLIC algorithm to segment the images into superpixels. To achieve the objective, after the segmentation of the images, an image dataset was created with the following classes: mildew, target spot, Asian rust, soil, straw and healthy leaves, totaling 22,140 images. Diagrammatic scales were used to assess disease severity. The disease recognition computer program explored four supervised learning techniques: SVM, J48, Random Forest and KNN. The techniques that obtained the best performance were SVM and Random Forests, taking into account the results obtained with all the evaluation metrics used. It was found that the program is efficient to differentiate the classes of diseases treated in this article.

Keywords: soybean diseases, segmentation, UAVs.

I. INTRODUÇÃO

A agropecuária brasileira é uma das principais bases econômicas do país. A realização de investimentos em ciência e tecnologia, colocou o Brasil entre os maiores produtores mundiais de alimentos, fibras e energias renováveis [1]. Focando especificamente na produção brasileira de grãos, a safra 2019/2020 foi estimada em 124,8 milhões de toneladas [2]

A soja se destaca pela série de produtos e subprodutos que são derivados da sua cadeia produtiva, demonstrando sua relevância para o agronegócio brasileiro. Para a obtenção de níveis de produtividade satisfatórios, é necessário realizar um bom manejo de pragas e doenças, sendo essas últimas de fundamental importância devido ao seu grande impacto negativo na cultura da soja. Dentre as doenças de maior destaque nas regiões produtoras, como o estado de Mato Grosso do Sul, pode-se citar a ferrugem asiática (*Phakopsora pachyrhizi*), a antracnose (*Colletotrichum truncatum*) e a mancha alvo (*Corynespora cassiicola*) [3].

Para o manejo adequado das doenças na cultura da soja deve-se realizar inspeções diárias nas lavouras para identificar sinais do ataque de doenças. O monitoramento deve ser realizado por profissionais treinados [4]. Todavia o monitoramento realizado visualmente, por humano, pode não ser adequado, já que duas avaliações, realizadas por pessoas diferentes, podem ter resultados diferentes considerando a percepção de cada um quanto à patologia e sua severidade. Além disso, essa técnica de monitoramento pode causar enfado ao observador.

[5] afirmam que alguns fatores podem dificultar a correta identificação das doenças foliares da soja, destacando alguns sintomas comuns a várias doenças em sua fase inicial. Além disso, a correta identificação é dificultada por monitoramentos superficiais, que deixam passar alguns sintomas das patologias despercebidas quando feitas a olho nu, já que a evolução de alguns sintomas não são perceptíveis ao olho humano e a visualização, com o auxílio de uma lupa que amplia de 10 a 20 vezes a imagem, pode significar um melhor auxílio a observação.

Assim, novas tecnologias são propostas objetivando facilitar a identificação e a tomada de decisão quanto ao controle de doenças em grandes culturas assim como a soja. O uso de veículos aéreos não tripulados, VANTs, tem se destacado como uma nova tecnologia que pode ser empregada para o monitoramento agrícola a exemplo de [6] que propuseram o uso do VANTs para monitorar o crescimento do arroz através de imagens multiespectrais.

A visão computacional tem se mostrado como uma importante aliada para a análise das imagens obtidas através dos VANTs, sendo empregada em programas de computador para mapear e identificar, por exemplo, focos de pragas ou doenças em cultivos agrícolas [7], [8], [9] e [19].

Este artigo tem como objetivo a utilização de um software que emprega um sistema de visão computacional para automatizar a detecção de doenças na cultura da soja. Essa

pesquisa contribui academicamente pela construção de banco de imagens anotadas contendo 22.140 imagens distribuídas entre as seguintes classes: míldio, macha alvo, ferrugem asiática, folhas saudáveis, solo e palhas. O banco de imagens será disponibilizado para colaborar com o desenvolvimento de novos sistemas de visão computacional.

II. MATERIAL E MÉTODOS

A condução do experimento passou por procedimentos necessários para o alcance dos objetivos propostos. Cada uma das etapas do desenvolvimento da pesquisa pode ser verificada nas seções subsequentes.

2.1 Instalação da lavoura de soja

O ensaio foi conduzido em uma fazenda. O campo com o plantio da soja foi instalado em uma área de um hectare. A cultivar utilizada foi BMX Potencia RR. O delineamento experimental adotado foi em blocos ao acaso, com quatro tratamentos (níveis de doença) e cinco repetições. As parcelas tiveram dimensões de 6x10m.

As condições meteorológicas, durante a condução da lavoura, foram monitoradas pela estação meteorológica instalada em uma fazenda ao lado da área experimental na qual foram monitorados dados de temperatura e umidade relativa do ar. Também foi monitorada a pluviometria. Destaca-se que as chuvas, se constantes, podem ocasionar encharcamento do solo, levando a problemas diversos nas plantas, como a vulnerabilidade das raízes ao ataque de patógenos.

2.2 Delineamento de Vôos e Captura de Imagens

Os registros de imagens foram realizados utilizando o equipamento VANT DJI Phantom 3 Professional, equipado com uma câmera Sony EXMOR 1/2.3", 12.4 M, lente FOV 94° 20 mm, suportando os formatos de arquivo FAT32/exFAT, JPEG, DNG e MP4, MOV (MPEG-4 AVC/H.264), e possui também um gimbal com estabilização nos 3 eixos e suporte a Micro SD com capacidade máxima de 64 GB.

Considerando o tamanho da área experimental (24x50m), a demarcação de cada parcela de 60m² foi sinalizada com estacas de bambu (1,3 metros de altura). Essa sinalização não permaneceu em sua totalidade devido à entrada de máquinas para aplicação dos defensivos.

Os vôos foram realizados a uma altura de 5 metros do solo e, a cada início de coleta de imagens, uma imagem única de todo o experimento foi obtida. Os vôos foram realizados entre os meses de dezembro de 2015 e março de 2016, pelo menos uma vez por semana, no período das oito às dez horas da manhã.

Devido à grande quantidade de chuvas durante o mês de janeiro, algumas visitas foram canceladas. Foram realizadas coleta de imagens e filmagem da área experimental, sendo a proporção da imagem utilizada 4x3 com resolução de 4000x3000px, e a filmagem realizada em Full HD com todos os parâmetros na configuração original de fábrica.

2.3 Seleção de imagens para composição do Banco de Imagens de Doenças na Soja

Com a realização dos voos na área experimental de soja, as imagens foram coletadas e posteriormente foram armazenadas em 12 pastas diferentes, conforme a data de captura. No total foram coletadas 711 imagens da plantação de soja durante a safra 2015/2016, correspondendo a 3,3GB.

Cada imagem possuía uma dimensão de 4000 x 3000 pixels e, em média 4,7MB, sendo necessário, devido a lentidão da máquina utilizada, o particionamento das imagens. Para tanto, foi implementado um programa que executou um *script* para automatizar a ação. Com isso, cada imagem foi particionada em 12 novas imagens (1000 x 1000 = 1 MP) alcançando o total de 8532 imagens da plantação de soja.

Posteriormente, efetuou-se o cálculo do tamanho da amostra aleatória simples para descrição da proporção populacional [10], utilizando um intervalo de confiança (IC) de 95% e erro padrão (EP) de 5%, chegando-se a uma amostra de 368 imagens. Como as imagens foram obtidas em 12 datas diferentes, a amostra calculada foi dividida pelo total de datas: $\frac{368}{12} = 30{,}7$ imagens de cada data. Devido a essa impossibilidade, optou-se pelo sorteio de 31 imagens de cada data, totalizando 372 imagens. O sorteio das imagens foi realizado através da função =ALEATORIOENTRE (x,y) do software Excel. A máquina utilizada para o desenvolvimento do trabalho foi um Notebook Samsung NP-RV411-AD3 c/ Intel Core i3 e 3GB de memória RAM.

2.4 Segmentação das imagens

Na segmentação das imagens para anotação posterior e criação do banco de imagens manualmente anotadas sob a supervisão de um agrônomo, foi utilizado o algoritmo para geração de superpixel, que tem sido cada vez mais utilizado em estudos relacionados à visão computacional. O superpixel refere-se ao desenvolvimento de um segmentador que combina características como contorno, textura, brilho e continuidade [11].

O SLIC (Simple Linear Iterative Clustering) foi introduzido por [12] e é uma adaptação do método de agrupamento *k-means* para geração de superpixels. O algoritmo superpixel SLIC agrupa locais de pixels no espaço 5-D definido por L, a, b (valores da escala CIELAB de cor) e as coordenadas x e y dos pixels [12] e [13]. Para tanto, tendo a imagem de entrada, realiza-se o particionamento da imagem em regiões, definindo-se o número k correspondente a quantidade de superpixels, levando cada superpixel a ter aproximadamente $\frac{N}{k}$ pixels, onde N é o número de pixels da imagem.

O processo do agrupamento começa com a etapa de inicialização, na qual os centros dos agrupamentos de superpixel $C_i = [l_i a_i b_i x_i y_i]$, com i=[1, k] são escolhidos, espaçados em um grid regular para formar os agrupamentos de tamanho aproximado S^2. Para que os superpixels tenham aproximadamente o mesmo tamanho, o intervalo da matriz é determinado pela seguinte equação:

$$S = \sqrt{\frac{Largura * Altura}{k}} \quad (1)$$

Assim, os centros do superpixel são movidos para locais com baixa magnitude de gradiente, numa vizinhaça 3x3, para evitar que um superpixel tenha seu centróide colocado sobre regiões de borda e para reduzir as chances dele conter pixels ruidosos. Posteriormente, no passo de atribuição, cada pixel i é associado com o centro mais próximo do agrupamento, cuja região de busca se sobrepõe à sua localização; esta é a chave para acelerar esse algoritmo, pois ao limitar o tamanho da região da busca, reduz-se significativamente o número de

cálculos de distância, o que resulta em uma vantagem de velocidade significativa sobre o agrupamento *k-means* convencional, no qual cada pixel deve ser comparado com todos os centros de agrupamento [12]. O processo anteriormente descrito só é possível através da introdução da medida de distância D, a qual determina o centro mais próximo para cada pixel:

$$d_{lab} = \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2} \quad (2)$$

$$d_{xy} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2} \quad (3)$$

$$D_s = d_{lab} + \frac{m}{s} * d_{xy} \quad (4)$$

Onde D é a soma da distância d_{lab} e a distância d_{xy} normalizada pelo intervalo S. A variável *m* corresponde ao controle de compactação do superpixel, quanto maior o seu valor, mais a proximidade espacial é enfatizada e mais compactado é o agrupamento. Este procedimento é repetido até a convergência ou até um número máximo de iterações T.

Na etapa de pós-processamento, os superpixels não representam necessariamente componentes ligados, de tal forma que o algoritmo precisa reforçar a conectividade através da re-atribuição de pixels disjuntos para superpixels próximos [12].

Por padrão, o único parâmetro de entrada do algoritmo SLIC Superpixel é o número de superpixels, de aproximadamente mesmo tamanho, *k* [12]. Todavia, opcionalmente é possível ajustar o parâmetro compacidade, *m*, que permite controlar a forma do superpixel tornando-a mais quadrada/cúbica.

Neste trabalho também foi utilizada a configuração do parâmetro sigma, que permite aplicar uma suavização na imagem, utilizando filtros gaussianos, antes da segmentação, através da utilização da biblioteca scikit-image (http://migre.me/wutyR), como pode ser visto na Figura 1.

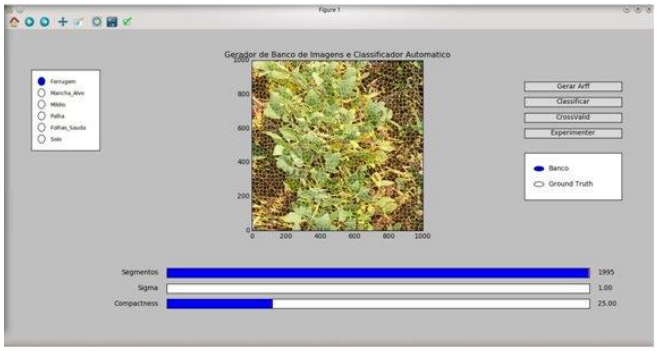


Figura 1 – Software segmentador para geração do banco de imagens.

Nesta pesquisa, os parâmetros utilizados para a segmentação das imagens foram: Segmentos (*k*) = 1995, Sigma = 1 e Compacidade (*m*) = 25. As doenças foliares da soja, principalmente no estágio inicial, possuem sintomas discretos, como pequenas manchas, conforme demonstrado na Figura 2, o que justifica o número de segmentos, 1995, que foi adequado para separar os sintomas conforme patologia verificada nos folíolos da soja. Além disso, o parâmetro *m* = 25 para o contorno da patologia, e o valor 1 do sigma proporciona a suavização necessária à imagem.

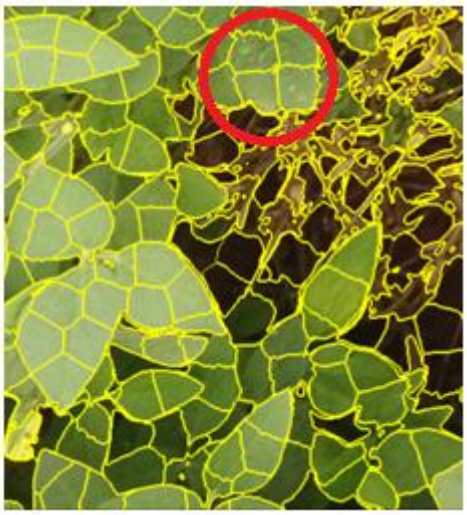


Figura 2 - Fragmento de uma imagem segmentada demonstrando visualmente a presença do míldio no folíolo da soja.

Após a segmentação, as imagens foram rotuladas manualmente nas classes: palha, solo, míldio, mancha alvo, ferrugem asiática e folhas saudáveis, sob supervisão de um agrônomo. A Figura 3 expõe algumas amostras que compõem o banco de imagens. O banco de imagens final conta com 22.140 imagens distribuídas em classes com as seguintes quantidades:

Ferrugem Asiática: 3894	Palha: 7170
Mancha Alvo: 75	Folha Saudável: 5049
Míldio: 1819	Solo: 4133

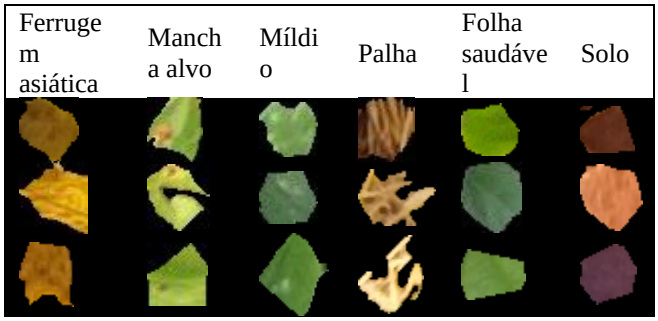


Figura 3 – Exemplo de imagens de classes que compõem o banco de imagens

2.5 Extração de atributos

Após a conclusão do banco de imagens, passou-se a etapa de extração de atributos utilizados como entrada para os classificadores explorados neste trabalho. Essa extração foi realizada utilizando uma coleção de extratores de forma, cor, textura e orientação da imagem implementados nas bibliotecas OpenCV e scikit-image.

Foram utilizados os seguintes extratores: atributos de cor RGB, HSV, Cielab (mín., máx., média e desvio); descritor de forma, invariante a escala, translação e rotação: 7 momentos de Hu; atributos de textura – Matriz de Co-ocorrência GLCM (Gray-Level Co-occurrence Matrix) (contrastes, dissimilaridades, homogeneidades, asm, energias, correlações); forma e orientação: HOG - Histogramas de Gradientes Orientados; atributos de textura: LPB - Padrões Binários Locais [18].

2.6 Classificação de imagens

Após segmentar e formar o banco de imagens anotadas, passou-se a classificação utilizando o programa de computador proposto para reconhecimento de doenças na soja. O software Weka versão 3.8 executado no Windows 64 bits foi utilizado. O Weka é um conjunto de algoritmos de aprendizado de máquina para tarefas de mineração de dados e tem como entrada arquivos no formato ARFF (Attribute-Relation File Format), que é um arquivo de texto ASCII que descreve uma lista de instâncias que compartilham um conjunto de atributos.

Os ARFFs utilizados como entrada para os testes no Weka neste trabalho foram gerados a partir dos extratores de atributos citados na seção 2.5. Os algoritmos utilizados para os testes comparativos foram Máquina de Vetores de Suporte (SVM), J48 (evolução do algoritmo C4.5), Florestas Aleatórias (no inglês Random Forest) e KNN (IBK no Weka) [18]. Todos os algoritmos foram executados com as configurações definidas por padrão no software Weka.

III. Resultados e Discussão

Para a construção de um sistema de visão computacional capaz de reconhecer patologias em imagens de produção da soja quatro técnicas de aprendizagem supervisionada foram exploradas: SVM, J48, FA (Floresta Aleatória) e KNN, e a técnica de amostragem utilizada foi a validação cruzadas com 10 dobras.

Todos os algoritmos foram executados com as configurações definidas por padrão no software Weka. Após a segmentação das imagens através do SLIC e extração de atributos para aprendizagem supervisionada, procedeu-se a classificação do banco de imagens no ambiente Weka. Os resultados das classificações foram avaliados por meio das métricas Porcentagem de Classificação Correta – PCC, Medida-F, Área Sob a Curva ROC, Kappa, Revocação e Precisão [18]. Os resultados estão demonstrados na Tabela 1.

Tabela 1 - Avaliação dos classificadores através de métricas

Métricas	SVM	J48	FA	KNN
PCC	**94,75**±0,42	89,76±0,57	92,90±0,45	83,38±0,68
Medida F	**0,950±0,00**	90,00±0,01	92,00±0,01	83,00±0,01
Curva ROC	0,950±0,00	78,00±0,03	**97,00±0,00**	68,00±0,02
Kappa	**93,00±0,01**	87,00±0,01	91,00±0,01	78,00±0,01
Revocação	**95,00±0,03**	90,00±0,04	93,00±0,04	83,00±0,03
Precisão	**94,00±0,03**	90,00±0,03	93,00±0,04	83,00±0,03

As métricas PCC, Medida F, Kappa, Revocação e Precisão indicaram que o classificador SVM obteve melhores resultados nas avaliações.

A Figura 4 apresenta a matriz de confusão gerada com a utilização do classificador SVM por obter melhores desempenhos com ambas as métricas onde as imagens em tom vermelho representam os valores mais altos.

Destaca-se que quanto mais próximos os valores das diagonais principais da matriz de confusão sejam do valor total de imagens contidas na classe, menor foi a confusão entre as classes.

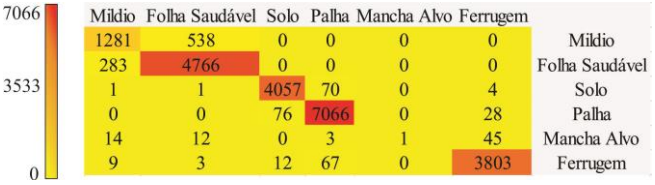


Figura 4 - Matriz de confusão gerada no ambiente Weka com o classificador SVM

Com as informações contidas na matriz, foram calculadas as métricas Coeficiente de Jaccard (CJ), Coeficiente de Yule (CY), precisão e revocação [18] para cada classe - míldio, folhas saudáveis, solo, palha, mancha-alvo e ferrugem asiática- conforme resultados apresentados na Tabela 2.

Considerando se tratar de um problema de classificação com mais de duas classes, o cálculo das medidas quantitativas foi realizado considerando a classe de interesse como a classe positiva, enquanto todas as outras foram consideradas como negativas [14].

Tabela 2 - Classes similares de acordo com os coeficientes de similaridade

Classes	CJ	CY	Precisão	Revocação
Míldio	0,6025	0,7805	0,8067	0,70423
Folhas Saudáveis	0,8506	0,8790	0,8959	0,94395
Solo	0,9611	0,9745	0,9788	0,98161
Palha	**0,9666**	0,9736	0,9806	**0,98550**
Mancha Alvo	0,0133	**0,9967**	**1,0000**	0,01333
Ferrugem	0,9577	0,9752	0,9802	0,97663

As métricas calculadas (Coeficiente de Jaccard (CJ), Coeficiente de Yule (CY), precisão e revocação) têm como base a comparação dos pixels das imagens segmentadas com as imagens de referência [15]. Os resultados apresentados para o Coeficiente de Jaccard e revocação indicaram que a melhor classificação foi com a classe palha, todavia entre as doenças, a melhor classificação ocorreu na classe ferrugem asiática. A mancha-alvo obteve melhor classificação quando observada através do coeficiente de Yule e da métrica precisão.

Para verificar se houve diferença significativa entre os classificadores foi aplicada a Análise de Variância, através do software R. Considerando o valor-p=0,0311, a um nível de significância de 0,05%, pode-se afirmar que há evidências estatísticas de que os desempenhos sejam diferentes entre os classificadores.

Dado que o teste ANOVA indica evidências estatísticas de que há diferença no desempenho entre os classificadores mas não indica quais classificadores se diferem, foi aplicado o teste de Tukey para essa identificação [16]. A Tabela 3 indica que há evidências estatísticas de que o desempenho é diferente entre os algoritmos SVM e KNN.

Tabela 3 - Teste de Tukey para os classificadores

Classificadores	Valor-p
KNN-J48	0.1267587
FA-J48	0.9921373
SVM-J48	0.7903252
FA-KNN	0.2012547
SVM-KNN	0.0226619
SVM-FA	0.6320562

A diferença entre os classificadores pode ser confirmada através da visualização da Figura 5 onde se pode analisar as medianas da taxa de acerto de cada classificador.

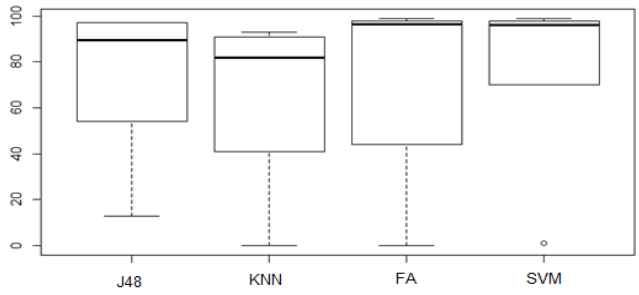


Figura 5 – Diagramas de caixa representando as diferenças entre os desempenhos dos classificadores

Após as técnicas de aprendizagem supervisionada (SVM, J48, Floresta Aleatória e KNN) serem avaliadas com o uso das métricas, o programa foi utilizado para realizar classificação de algumas imagens obtidas através do VANT na plantação de soja, conforme demonstrado na Figura 6, onde a primeira imagem da esquerda para a direita refere-se à imagem original e a segunda refere-se à imagem já classificada pelo programa. As imagens foram obtidas pelo VANT na plantação de soja durante o experimento, sendo capturadas, respectivamente nas seguintes datas: 25/02/2016, 04/03/2016 e 08/03/2016.

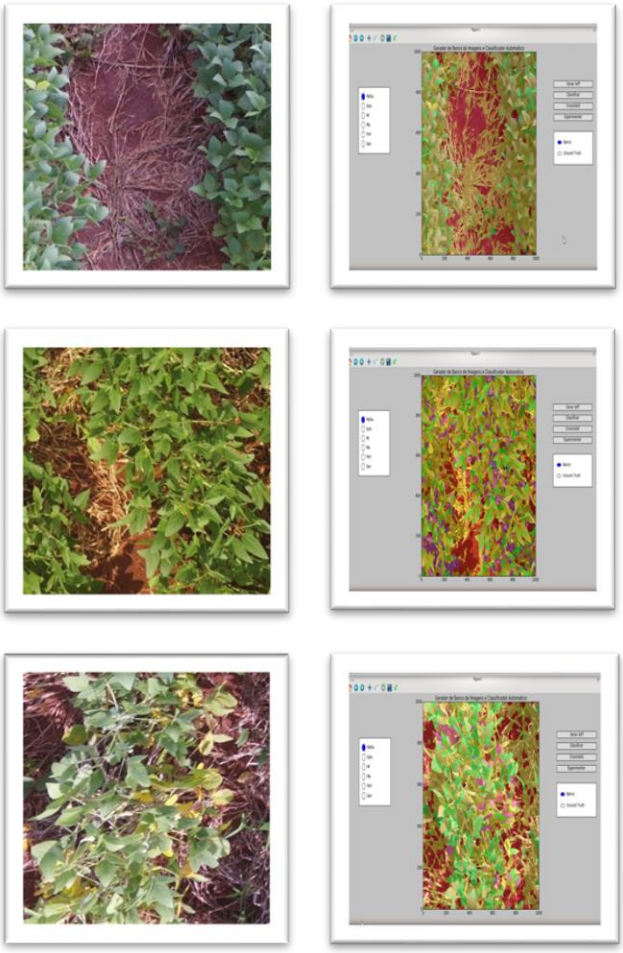


Figura 06 – Classificação visual das doenças da soja nas classes: míldio , macha alvo , ferrugem asiática , folha saudável , solo e palha realizada pelo programa de computador.

Para a realização dessa classificação visual a imagem foi segmentada pelo algoritmo SLIC (utilizando os mesmos parâmetros empregados na segmentação da criação do banco de imagens) e, posteriormente, os segmentos foram classificados de maneira automática pelo programa, como pertencentes a uma das classes: míldio, macha alvo, ferrugem asiática, folha saudável, solo e palha. O algoritmo utilizado na classificação foi o SVM.

A primeira imagem não apresentou a presença da mancha alvo da soja, mas teve evidências de ferrugem e míldio (6,71% e 0,32%, respectivamente). Na segunda imagem, o ataque das doenças míldio, mancha alvo e ferrugem foi de, respectivamente, 13,83%, 0,62% e 14,79%. Na última imagem, as doenças míldio, mancha alvo e ferrugem alcançaram o total de 21,56%, 0,04% e 5,91%.

A observação visual das imagens demonstra uma variação de cor para as patologias. Todavia, no caso da ferrugem asiática, a coloração das regiões foliares afetadas são predominantemente castanho-claras ("TAN"). Todavia, cultivares onde o patógeno é resistente as lesões são predominantemente castanho-avermelhadas (("reddish-brown - RB") [17].

Além disso, no final do ciclo algumas doenças podem ser confundidas [5]. A semelhança nos sintomas da ferrugem e de outras doenças pode levar a confusão e, consequentemente, a não identificação correta das patologias [17]. Nesse sentido, a classificação dos segmentos que compõem cada imagem em suas respectivas classes, conforme aprendizado recebido pelo programa de computador, proporciona maior confiabilidade acerca dos diagnósticos obtidos.

IV. Conclusão

Esse artigo teve como objetivo reconhecer doenças no experimento de soja e, para tanto, foi implementado um programa que teve como base aprendizagem automática e visão computacional. No experimento foram verificadas a presença das doenças: ferrugem asiática, mancha-alvo e míldio.

Na classificação das imagens anotadas pertencentes ao banco de imagens realizadas com os algoritmos de classificação do Weka, por meio da avaliação com as métricas, os resultados indicaram que o melhor desempenho foi obtido pelo classificador SVM, exceto para a métrica Área Sob a Curva ROC que obteve o melhor desempenho com o classificador Florestas Aleatórias.

O classificador utilizado para a construção das matrizes de confusão, para cálculo das demais métricas e avaliação entre as classificações foi o SVM por obter melhor desempenho na maioria das métricas utilizadas na avaliação. Entre as doenças, os resultados apresentados para o coeficiente de Jaccard e revocação na classe ferrugem asiática. A mancha-alvo obteve melhor classificação quando observada através do coeficiente de Yule e da métrica precisão.

Assim, verifica-se que o programa é apto a diferenciar as classes, onde as técnicas que obtiveram melhor desempenho foi o SVM e Florestas Aleatórias, levando em consideração os resultados obtidos com todas as métricas de avaliação utilizadas.

Destaca-se a importância de um bom resultado na classificação principalmente porque um erro pode induzir um produtor de soja a prejuízos, seja prejuízo econômico, social

ou ambiental, já que a demora no uso do defensivo pode levar a um resultado, e o apressar em defender o plantio com o uso dos defensivos pode levar a outros prejuízos.

A proposta demonstra alcançar o objetivo proposto para esta pesquisa e contribui com outros pesquisadores pela disponibilização de um banco de imagens robusto com as classes trabalhadas nesse artigo. Além disso, o programa pode atuar enquanto um redutor da subjetividade existente nas avaliações realizadas por técnicos ou especialistas, no momento de reconhecer as doenças, melhorando o nível de acerto em relação a avaliação humana na classificação das imagens.

V. AGRADECIMENTOS

Este trabalho recebeu apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES), da Universidade Católica Dom Bosco (UCDB), e da Universidade Federal de Mato Grosso do Sul (UFMS).

REFERENCIAS BIBLIOGRÁFICAS

[1] EMBRAPA - EMPRESA BRASILEIRA DE PESQUISA AGROPECUÁRIA - EMBRAPA SOJA. Tecnologias de produção de soja – região central do Brasil 2014. - Londrina: Embrapa Soja, 2014.

[2] COMPANHIA NACIONAL DE ABASTECIMENTO – CONAB. Acompanhamento da Safra Brasileira de Grãos. V7. Safra 2019/2020. Brasília. Set/2020. Disponível em <https://www.conab.gov.br/info-agro/safras/graos/boletim-da-safra-de-graos>. Acesso em: 12 de setembro de 2020.

[3] J. F. J. GRIGOLLI, Manejo de Pragas e Doenças na Cultura da Soja. Palestra: Fitossanidade Safra 2015. Fundação MS. Disponível em <http://migre.me/rdAYC> Acesso em Ago/2015.

[4] M. J. AFRIDI; X. LIU; J. MITCHELL MCGRATH. "An Automated System for Plant-level Disease Rating in Real Fields," in Proceedings of the 22nd International Conference on Pattern Recognition (ICPR 2014), Stockholm, Sweden, August 24-28, 2014.

[5] M. S. BONALDO; I. C. RIEDO; A. R. LIMA. Monitoramento e diagnóstico de doenças foliares da cultura da soja na região COMCAM na safra 2007/2008. Campo Digital, Campo Mourão, v. 4, n. 1, p. 127–136, 2009.

[6] Y. JIA; Z. SU; W. SHEN; J. YUAN; Z. XU. UAV Technology and Its Application in Agriculture. Advanced Science and Technology Letters. Vol.137 (SUComS 2016), pp.107-111 http://dx.doi.org/10.14257/astl.2016.137.20

[7] J. M. PEÑA-BARRAGAN; K. M de CASTRO; F. LOPEZ-GRANADOS. Object-based approach for crow row characterization in UAV images for site-specific weed management. In Queiroz-Feitosa et al.., editors. 4th International; Conference on Geographic Object-Based Image Analysis (GEOBIA 2012), Rio de Janeiro, Brazil, p. 426-430

[8] D. GÓMEZ-CANDÓN, A.I.D CASTRO; F. LÓPEZ-GRANADOS. Assessing the accuracy of mosaics from unmanned aerial vehicle (UAV) imagery for precision agriculture purposes in wheat. Precis. Agric. 2014-15, 44–56.

[9] V. UGALE; D. GUPTA. A Comprehensive Survey on Agricultural Image Processing. International Journal of Science and Research (IJSR). Volume 5 Issue 1, January 2016

[10] W. BUSSAB; P. A. MORETTIN. (2011). Estatística básica (7a ed.). São Paulo: Saraiva.

[11] X. REN; J. MALIK. Learning a classification model for segmentation. IEEE ICCV, pp. 10–17, 2003.

[12] R. ACHANTA; K. SMITH; A. LUCCHI; P. FUA; S. SUSSTRUNK. SLIC superpixels. Technical report, EPFL, Tech.Rep. 149300, 2010.

[13] J. LV. An Improved SLIC Superpixels using Reciprocal Nearest Neighbor Clustering. International Journal of Signal Processing, Image Processing and Pattern Recognition Vol. 8, No. 5 (2015), pp. 239-248. Disponívelem<http://dx.doi.org/10.14257/ijsip.2015.8.5.25> Acesso em 20 de jul. de 2015

[14] B. LANTZ. Machine Learning with R. Packt, 2° Edição, 2015.

[15] W. T. ANDRADE; L. N. B. QUINTA; A. B. GONCALVES; M. P. CEREDA; H. PISTORI. Segmentação Baseada em Textura e Watershed aplicada a Imagens de Pólen. In: SIBGRAPI 2012 - Conference on Graphics, Patterns and Images, Workshop of Undergraduate Work (WUW), 2012, Ouro Preto - MG. Anais do SIBGRAPI, 2012.

[16] H. ABDI; L. WILLIAMS, L. Newman-Keuls; and Tukey test. In Salkind, N., Frey, B., & Dougherty, D. (Eds.), Encyclopedia of Research Design, pp. 897–904. Sage, Thousand Oaks, CA, (2010).

[17] EMBRAPA SOJA. Tecnologias de produção de soja: região Central do Brasil, 2003. Londrina, 2004. 239p.

[18] G. G. SILVA. Superpixel e aprendizagem supervisionada para a identificação de doenças da soja em imagens obtidas por veículos aéreos não tripulados - Tese (doutorado em Ciências Ambientais e Sustentabilidade Agropecuária) – Universidade Católica Dom Bosco, Campo Grande -MS. 114 p. 2017.

[19] E. C. TETILA, B. B. MACHADO, G. K. MENEZES, A. OLIVEIRA, M. A. VEGA, W. P. AMORIM, N. A. S. BELETE, G. G. SILVA; H. PISTORI. Automatic recognition of soybean leaf diseases using UAV images and deep convolutional neural networks, IEEE Geoscience and Remote Sensing Letters, 2019.

Unsupervised Segmentation of Breast Infrared Images in Lateral View Using Histogram of Oriented Gradients

Thays Lacerda Correa*, Fabíola Freitas de Oliveira *, Matheus de Freitas Oliveira Baffa†, Lucas Grassano Lattari *
*Instituto Federal de Educação, Ciência e Tecnologia do Sudeste de Minas Gerais (IF Sudeste MG)
Rio Pomba, MG, Brasil
†Universidade de São Paulo (USP)
Ribeirão Preto, Brasil
Email: thays.lacerdac@gmail.com, ffabiolafreitas@gmail.com, mfreitas826@gmail.com, lucas.lattari@ifsudestemg.edu.br

***Resumo*—Breast cancer is the second most common type of cancer in the world. It is estimated that 29.7% of new cases diagnosed in Brazil occur in any structures of the breasts. However, the disease has a good prognosis if detected early. Thus, the development of new technologies to help doctors to provide an accurate diagnosis is indispensable. The goal of this work is to develop a new method to automate parts of computer-aided diagnosis systems, performing the unsupervised segmentation of the Region of Interest (ROI) of infrared breast images acquired in lateral view. The segmentation proposed in this paper consists of three stages. The first stage pre-processes the infrared images of the lateral region of breasts. Later, features are extracted from a descriptor based on Histogram of Oriented Gradients (HOG). Concluding, a Machine Learning algorithm is used to perform the segmentation of the sample. The current method obtained an average of 89.9% accuracy and 94.3% specificity in our experiments, which is promising compared to other works.**

***Index Terms*—breast cancer, histogram of oriented gradients, image segmentation, infrared imaging, computer vision, computer aided diagnosis**

I. Introdução

O carcinoma mamário, ou câncer de mama, é uma doença caracterizada pelo surgimento de uma neoplasia em algumas das estruturas que compõem a mama. Esta pode atingir homens e mulheres, geralmente acima dos 35 anos, sendo mais frequentes em mulheres após o período da menopausa [9].

De acordo com o Instituto Nacional de Câncer (INCA), o câncer de mama é o segundo tipo mais frequente [11]. Estima-se que 29,7% dos novos casos de câncer registrados no Brasil, dentre as mais de 100 variações, sejam do tipo mamário e, embora raro, 1% do total de casos ocorrem em homens. Além disto, estima-se que uma em cada oito mulheres desenvolverão a doença em algum momento de sua vida [7], [8].

Desde 2013 notou-se um aumento de 4,7% na taxa de incidência da doença no Brasil. Embora o número de casos registrados tenha aumentado, o número de mortes pelo câncer têm reduzido. Este comportamento é visível quando se compara a taxa de incidência do câncer de mama e sua taxa de mortalidade, por ano, entre os países desenvolvidos e em desenvolvimento [3]. Isso ocorre devido à detecção precoce da doença, possibilitando tratamentos menos agressivos, como os tratamentos sistêmicos e proporcionando uma melhor qualidade de vida ao paciente.

Para o ano de 2018, o INCA registrou, aproximadamente, 17 mil mortes pelo câncer de mama no Brasil. Esse valor é referente a aproximadamente 16,4% dos casos registrados [7]. Embora possua uma alta taxa de incidência e óbito, o câncer de mama também é considerado o tipo com melhor prognóstico, possibilitando ao paciente até 95% de chances de cura quando diagnosticado precocemente [6], [7].

Existem diferentes maneiras de se diagnosticar o câncer de mama. Um desses meios é através do autoexame, no qual a própria pessoa percebe sinais de anomalias ou a presença de caroços na pele. Outra forma de detecção é através dos exames por imagem, que possuem um alto grau de eficácia. Um exemplo de exame por imagem bastante utilizado é a mamografia, que emprega Raios-X para registrar as estruturas internas da mama e, assim, localizar possíveis anormalidades.

As imagens de infravermelho ou termografia, por sua vez, possuem grande potencial para a detecção precoce dessa doença. Isso ocorre devido ao processo de formação deste tipo de neoplasia. Diversos fatores como a atividade metabólica e o processo de angiogênese fazem com que determinadas regiões da mama produzam calor de forma anormal e contribuem com desbalanceamento da distribuição térmica entre as mamas [1], [10]. Além disto, as imagens de infravermelho podem ser sintetizadas utilizando dispositivos mais baratos economicamente e menos desconfortáveis quando comparados por exemplo, com o mamógrafo e a máquina de Ressonância Magnética, equipamentos amplamente usados em exames de imagem.

Neste âmbito, sistemas de apoio ao diagnóstico (do inglês, *Computer-Aided Diagnosis* – CAD) são desenvolvidos com o intuito de prover um diagnóstico mais preciso e eficaz. Considerando a disponibilidade de dados e o poder de processamento das máquinas atuais, é viável o desenvolvimento de sistemas CAD que integram imagens infravermelhas e algoritmos de Aprendizado de Máquina sofisticados.

Típicos sistemas CAD são compostos por quatro etapas, sendo elas (i) a Pré-Processamento de Imagens, (ii) a Segmentação da Região de Interesse (ROI - *Region of Interest*), (iii) a Extração de Atributos ou Características (*features*) e (iv) a Classificação [12]. No primeiro estágio pode-se aplicar algum procedimento que realce ou reduza alguma característica da imagem originalmente obtida, a fim de se potencializar os resultados das etapas posteriores. A seguir, a etapa de Segmentação remove toda a informação não necessária para a análise da imagem a posteriori, como o fundo, permanecendo apenas a ROI. Na terceira etapa, informações relevantes para a análise são extraídas. Por fim, as características extraídas são usadas para o treinamento de algoritmos de Aprendizado de Máquina que produzirão modelos de classificação para que se possa diagnosticar exames, considerando um conjunto de amostras de pacientes saudáveis e doentes.

Esse trabalho propõe uma metodologia para a etapa de Segmentação da ROI de Imagens Infravermelhas das Mamas atendendo ao protocolo lateral, em que o registro de fotos é rotacionado em 90º. Essa proposta é não supervisionada (automática) e sua avaliação metodológica é feita com imagens advinda de bases de dados médicas públicas. Para ilustrar, a Figura 1 demonstra exemplos de imagens dos protocolos mais comumente adotados, que são: o frontal e o lateral. A Figura 2 ilustra qual o tipo de imagem de entrada utilizada e sua saída esperada (segmentação da ROI).

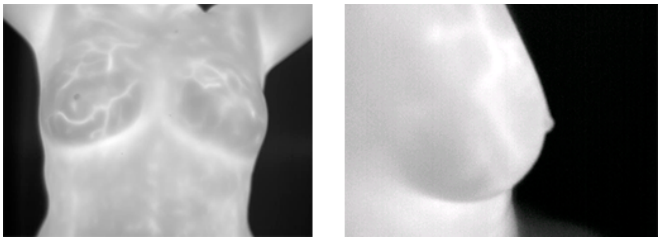


Figura 1: Imagens infravermelhas das mamas adquiridas com o protocolo frontal (à esquerda) e lateral (à direita). Fonte: autor.

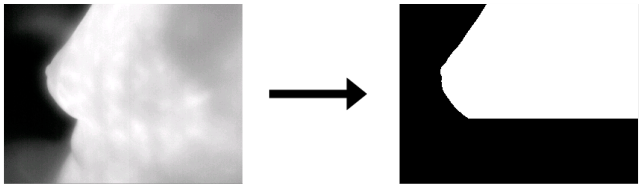


Figura 2: Exemplo de imagem infravermelha usada (à esquerda) e segmentação da ROI esperada que o método realize (à direita). Fonte: autor.

Algumas contribuições dessa proposta:

- Descrição de um novo método para a segmentação automática de imagens em protocolo lateral das mamas. Isso é importante considerando que um número diminuto de pesquisas envolvendo o protocolo lateral são desenvolvidas.
- Possibilidade de generalização da proposta para problemas com outras regiões de interesse.
- Discussão e comparação qualitativa com outras estratégias que empregam o protocolo lateral.
- Apresentação de resultados numéricos competitivos com outros trabalhos que lidam com o mesmo problema.
- Portabilidade para outras arquiteturas dada a utilização de algoritmos amplamente conhecidos e adotados, como é o caso de Histograma de Gradientes Orientados [4].

Este trabalho está organizado da seguinte forma: a Seção II retrata o estado da arte em segmentação de imagens infravermelhas laterais das mamas; as seções III e IV, respectivamente, detalham o método proposto, os experimentos empregados, o desempenho quantitativo dos mesmos e suas limitações; finalmente, a Seção V apresenta as conclusões obtidas e os trabalhos futuros da proposta.

II. Trabalhos Relacionados

A suma maioria das metodologias de segmentação de imagens infravermelhas são desenvolvidas com maior enfoque para imagens frontais das mamas [2], [16], [18]. No entanto, o escopo deste trabalho objetiva produzir resultados em imagens sob uma visualização em ângulos distintos da região das mamas (protocolo lateral). É de comum conhecimento que a visualização de imagens médicas é sensível à posição que o exame é gerado ou ao recorte realizado, trazendo diferentes informações conforme o ângulo se altera.

Em Oliveira [19] é proposta uma metodologia de segmentação automática de imagens infravermelhas laterais. Inicialmente, as 328 imagens utilizadas pelo autor são convertidas em preto e branco (pré-processamento). Em seguida, são identificados e eliminados os elementos indesejáveis no fundo das imagens. Por fim, a última etapa é responsável pela detecção e refinamento de cantos, identificando a prega inframamária e, assim, atingindo a segmentação das imagens automaticamente. O método obteve uma acurácia de 93% e especificidade de 96%.

Recentemente, o trabalho de Josephine et al. [13] propõe um fluxo de trabalho que consiste em: remover informações textuais e de temperatura por meio de um algoritmo *in-paint*, realce de imagens por meio de um filtro de difusão anisotrópico e extração de ROI por meio de um algoritmo baseado na abordagem de *level sets*. Para avaliação de resultados, foram usadas as métricas de índice de Jaccard, coeficiente de Sørensen–Dice, índice de Similaridade Estrutural (*Structural Similarity Index*), dentre outros. Apenas 15 imagens foram usadas em sua proposta, obtendo 95% de correlação entre as áreas segmentadas pelo trabalho e as regiões reais esperadas.

Outro trabalho de suma importância fora apresentado por Morales-Cervantes et al [17], que combinaram imagens de protocolos laterais e frontais em seu método. Por meio de uma abordagem que emprega o filtro Sobel e o espaço de cor L*a*b*, estes obtiveram uma sensibilidade de 100% e acurácia de 69,9%.

Finalmente, pode-se mencionar os trabalhos de Santana et al. [5], Mambou et al. [15] e Kakileti et al. [14] que, ainda que não estejam diretamente relacionados com a segmentação de imagens das mamas usando protocolo lateral, são propostas recentes importantes para a literatura. Respectivamente, o primeiro é um estudo profundo sobre classificação de imagens de infravermelho (inclusive com o protocolo lateral), o segundo aplica a classificação diagnóstica utilizando Aprendizado Profundo (*Deep Learning*) e o último descreve os avanços recentes envolvendo diagnóstico de imagens infravermelhas das mamas em formato *survey*.

III. Materiais e Métodos

Essa seção será subdividida em três partes. A primeira subseção descreve a base de dados empregada para os experimentos a fim de validar a proposta; a segunda subseção enfatiza o funcionamento do classificador utilizado para reconhecer determinadas seções da imagem; finalmente, a última parte explica como funciona o estágio de segmentação de imagens do trabalho proposto.

A. Base de Dados

O método proposto neste trabalho foi testado em um banco público de imagens da UFF/UFPE que encontra-se disponível em PROENG [20]. Estas imagens são proveniente de estudos realizados por pesquisadores da Universidade Federal de Pernambuco e seus parceiros. Essa base de dados contém imagens com protocolo estático e foram capturadas pelo Hospital das Clínicas da mesma. Na base supracitada, há imagens infravermelhas capturadas em oito ângulos distintos. No entanto, no contexto desse trabalho, apenas as imagens de infravermelho das laterais das mamas foram empregadas no processo de segmentação, que possuem ângulo de 90º. Devido a algumas inconsistências na base, como falta de informações sobre o diagnóstico das pacientes, algumas imagens foram eliminadas dessa análise. No total, 214 imagens foram utilizadas para os experimentos apresentados nas seções III-B e III-C.

Inicialmente, são usadas imagens limiarizadas da base original, convertidas para tons de cinza. Esse processo produz uma imagem em formato *bitmap*.

B. Classificação do Modelo e Reconhecimento de Contorno das Pregas Inframamárias

Após obter as imagens de infravermelho em preto e branco, segmentos das mesmas são extraídos manualmente das imagens originais. Isso faz-se necessário para assim treinar um modelo classificador que distinguirá regiões diversas das imagens das mamas.

Uma região fundamental para o funcionamento desse métodos são os cantos contendo pregas inframamárias. Um exemplo de tal região é assinalada na Figura 3.

Conjuntos de segmentos extraídos manualmente são então introduzidos em dois subconjuntos: imagens cujo canto da prega inframamária encontra-se na posição central da imagem e imagens com segmentos diversos das mamas que não contenham cantos. Chamaremos esses conjuntos, respectivamente, de $IMGS_{pos}$ e $IMGS_{neg}$ (Figura 4). Nessa proposta, 61 segmentos fazem parte de $IMGS_{pos}$ e 189 segmentos foram atribuídos a $IMGS_{neg}$.

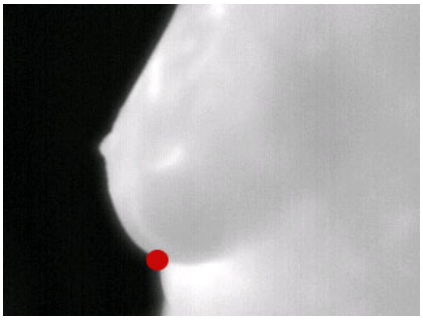
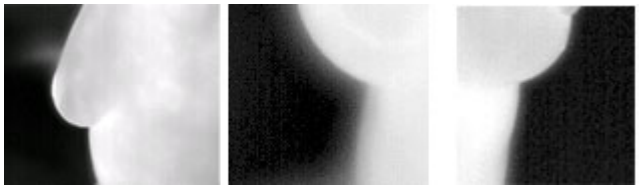
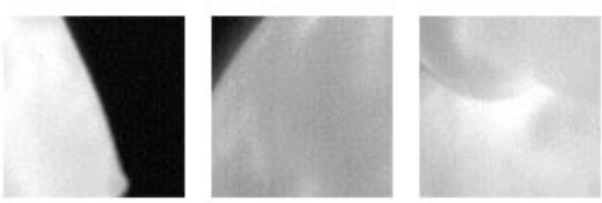


Figura 3: Amostra com as regiões de canto consideradas na análise descrita nessa seção. Fonte: autor.



(a) Imagens introduzidas em $IMGS_{pos}$.



(b) Imagens introduzidas em $IMGS_{neg}$.

Figura 4: Exemplos de imagens usadas para treinar o modelo classificador dessa seção. Em (a), o canto das imagens (prega inframamária) estão centralizados, a fim de se especificar ao modelo que essas regiões precisam ser localizadas nas amostras de entrada. Quanto a (b), tem-se imagens sem os cantos assinalados na região central. Dessa forma, o modelo irá compreender que não se deve assinalar que essas sejam regiões rotuladas como cantos. Fonte: autor.

Os segmentos contidos em $IMGS_{pos}$ e $IMGS_{neg}$ serão parte do conjunto de treino de um classificador descrito por Histogramas de Gradientes Orientados. Esse descritor foi originalmente proposto para o reconhecimento automático de pedestres em imagens [4]. Esse vetor de características é bastante utilizado em problemas de Visão Computacional para detectar ou reconhecer objetos dos mais diversos que se diferenciem visualmente, principalmente por meio da geometria e textura. Sua proposta baseia-se em um conjunto robusto de características para discriminar regiões com variações de iluminação e pose.

Para o treinamento do classificador, regiões diversas das

imagens são recortadas atendendo a um tamanho padrão. Esses recortes possuem aproximadamente 30% da área original dessas imagens, sendo distinguidas manualmente em $IMGS_{pos}$ e $IMGS_{neg}$.

A seguir, são computados os descritores de Histograma de Gradientes Orientados de cada uma das imagens de ambos os conjuntos. Os parâmetros aplicados são: 8 histogramas de orientação dos gradientes, grades de células da ordem de 16x16 pixels, 4x4 células por bloco e, finalmente, os blocos são normalizados usando a norma L2. Essa configuração é tradicional para problemas desse tipo e funciona adequadamente para a atual proposta, conforme apresentado na seção de resultados (Seção IV).

Estes descritores das imagens são vetorizados e introduzidos como amostras em um classificador baseado em Máquina de Vetor Suporte (*Support Vector Machine* – SVM). O SVM corresponde a um algoritmo de Aprendizado de Máquina supervisionado que pode ser usado tanto para classificação quanto para regressão. Nesse artigo, o SVM é empregado para fins de classificação, com os seguintes parâmetros: o termo de penalidade *C* (usado para ajustar a distância entre seu hiperplano separador e a primeira amostra de cada classe $IMGS_{pos}$ e $IMGS_{neg}$) é da ordem de 1.0. A função de kernel empregada é a RBF e o coeficiente de kernel γ é computado automaticamente (de forma que o mesmo é $\frac{1}{n_f}$, em que n_f é o número de características advindas dos descritores). A importância de γ define a influência em que uma única amostra do treinamento terá no modelo como um todo. Após aplicado o SVM, tem-se o modelo M.

Finalmente, M será usado para reconhecer automaticamente o segmento s_i que contém os candidatos a canto da prega inframamária de cada uma das imagens i. Espera-se que o canto esteja centralizado em s_i. Uma vez encontrado, anota-se uma coordenada local (x, y) exatamente na posição em que se encontra o canto em s_i para que seja usada na etapa seguinte, de segmentação. Assim, ao final da etapa de classificação, tem-se as posições x e y que indicam a localização de cada prega inframamária de cada amostra i. Alguns exemplos dessas imagens encontram-se apresentadas na Figura 5.

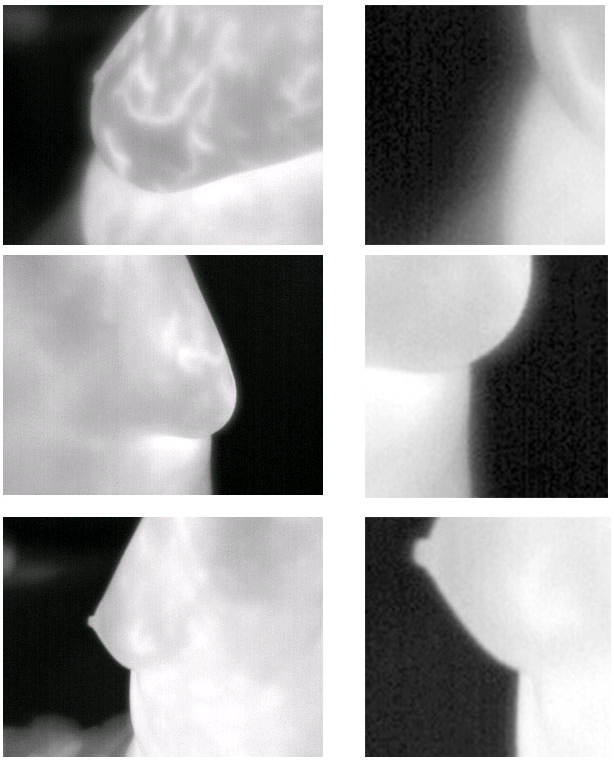


Figura 5: Algumas imagens da base de dados de mama lateral e suas versões computadas contendo a região da prega inframamária próxima a sua região central. Fonte: autor.

C. Segmentação das Imagens de Infravermelho

Na etapa de segmentação cada imagem da base de dados é analisada e, de cada uma, é extraída a região lateral contendo a mama. Alguns exemplos de imagens segmentadas pelo método podem ser vistas na Figura 6.

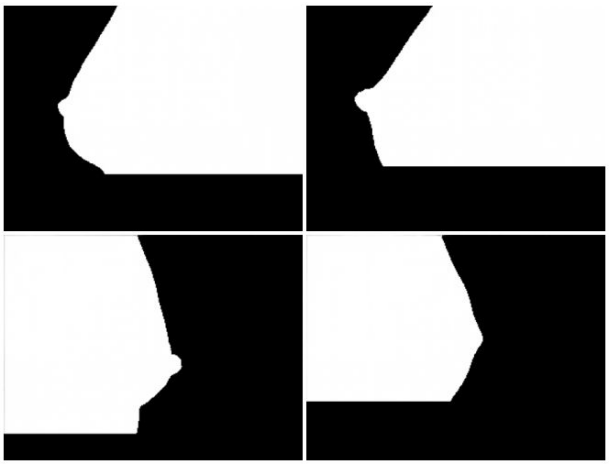


Figura 6: Exemplo de amostras segmentadas pelo método proposto. Fonte: autor.

Inicialmente, computa-se a binarização da imagem original em formato de tons de cinza. O limiar empregado que comportou-se bem na proposta foi o de 150, considerando que inicialmente cada imagem possui 8 bits de nível de cor (255).

A seguir, computa-se o histograma de linha $h_{si}(y)$ da versão computada s_i no passo anterior (Seção III-B). Como é dito, cada s_i contém a região da prega inframamária centralizada, tendo sido detectada por meio do modelo SVM usando o descritor HOG. Devido ao fato de, usualmente, a região da prega inframamária conter a menor densidade do histograma de linha, este é um bom indicativo para localizar essa região. Assim, a coordenada y contendo o $h_{si}(y)$ de valor mínimo para s_i sinaliza a posição vertical em que se localiza a prega inframamária.

No entanto, convém mencionar que deve-se iniciar a computação de $h_{si}(y)$ na posição $y = 0.3 \times i_{altura}$ da imagem, sendo que i_{altura} é a altura de i. Isso é necessário pois, anali-

sando empiricamente, foi constatado que, na figura mamária, a prega inframamária encontra-se costumeiramente na região limítrofe inferior da imagem. Todos os pixels localizados em posições iguais ou abaixo de y na imagem original i em que foi detectada a prega inframamária são sumariamente removidos da imagem. As constantes apresentadas aqui foram calculadas empiricamente,, após exaustivos experimentos a fim de obter o melhor resultado. Um exemplo de tal procedimento é demonstrado na Figura 7.

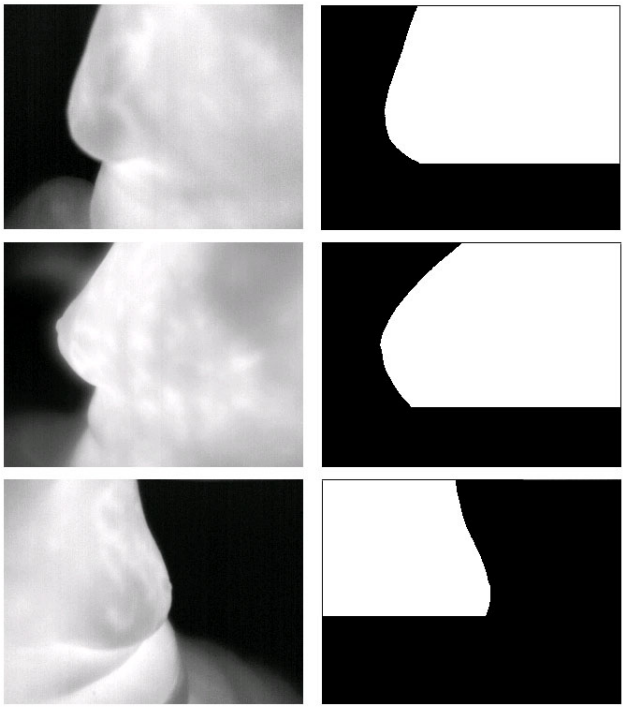


Figura 7: Comparativo entre a imagem original e sua contraparte após a remoção dos pixels proporcionada pela etapa de segmentação. Fonte: autor.

IV. Resultados e Discussões

A avaliação do método proposto baseia-se na comparação dos resultados numéricos obtidos com as segmentações automáticas realizadas. Uma análise quantitativa foi realizada utilizando as seguintes métricas de avaliação quantitativa: acurácia, sensibilidade, especificidade, preditividade positiva e preditividade negativa.

A Tabela I contém um comparativo dos resultados obtidos neste trabalho em contraste com os resultados obtidos por Oliveira [19] para a mesma base de imagens infravermelhas das laterais da mama utilizadas neste trabalho. Ambos os trabalhos compararam as segmentações automáticas resultantes utilizando uma segmentação manual previamente realizada por um médico especialista para fins de comparação.

Considerou-se também os resultados obtidos por Morales-Cervantes et al. [17], ainda que não tenham sido usadas as mesmas imagens em nossos experimentos, visto que as imagens usadas por eles não encontram-se publicamente disponíveis. A comparação do método proposto com o trabalho de Morales-Cervantes et al. pode ser visto na Tabela II. Esse problema também ocorreu em relação ao trabalho de Josephine et al. [13], com o agravante das métricas adotadas serem distintas das nossas. De qualquer forma, optou-se por compará-las para mostrar resultados de outras aplicações envolvendo segmentação de imagens médicas das mamas usando protocolo lateral.

Tabela I: Comparação de resultados entre o trabalho de Oliveira [19] e o método proposto usando a mesma base de dados. Fonte: autor.

Métrica	**Oliveira [19]**	**Este Trabalho**
Acurácia	93%	89,6%
Sensibilidade	95%	85,7%
Especificidade	96%	94,3%
Preditividade Positiva	96%	92,7%
Preditividade Negativa	96%	89,1%

Tabela II: Comparação de resultados entre o trabalho de Morales et al. [17] e o método proposto sem que seja usada a mesma base de dados. Fonte: autor.

Métrica	**Morales-Cervantes [17]**	**Este Trabalho**
Acurácia	69,9%	89,6%
Sensibilidade	100%	85,7%
Especificidade	Não avaliado	94,3%
Preditividade Positiva	11,42%	92,7%
Preditividade Negativa	100%	89,1%

O método proposto nesse artigo (Seção III) foi implementado usando a linguagem Python em sua versão 3.5.4 x64 com as bibliotecas OpenCV para manipulação de imagens e Scikit-Learn para a utilização do Histograma de Gradientes Orientados e da Máquina de Vetor Suporte. O equipamento empregado para a aquisição de resultados foi um computador com processador Intel Core i5-2450M com 2.50Ghz e 6GB de Memória RAM, com sistema operacional Windows 7 Home Premium.

Nessa proposta, a acurácia da segmentação aproximou-se de 90%, sendo bastante competitiva com relação ao trabalho de Oliveira [19]. A sensibilidade, por sua vez, apresenta uma relação para a classificação correta dos pixels considerados positivos nas imagens. Dessa forma, essa métrica mostra o quanto de acerto houve com relação aos pixels que são efetivamente pertencentes as mamas. A porcentagem de 85,7% descreve que o método proposto acarreta em maiores erros ao classificar os pixels que pertencem às regiões mamárias do que o outro trabalho comparado.

Um caso típico de falha do método refletido na sensibilidade é visível na Figura 8(a). Ao aplicar a limiarização na figura 8(a), tem-se a imagem 8(b). Nela, é possível ver que a posição y com o histograma de linha com valor mínimo está fora da prega inframamária (Figura 8(c)). O resultado esperado encontra-se na imagem 8(d).

Por sua vez, a proporção de acertos considerando apenas os pixels que pertencem aos verdadeiros negativos (fundo da imagem e região abaixo da prega inframamária) apresentam bom

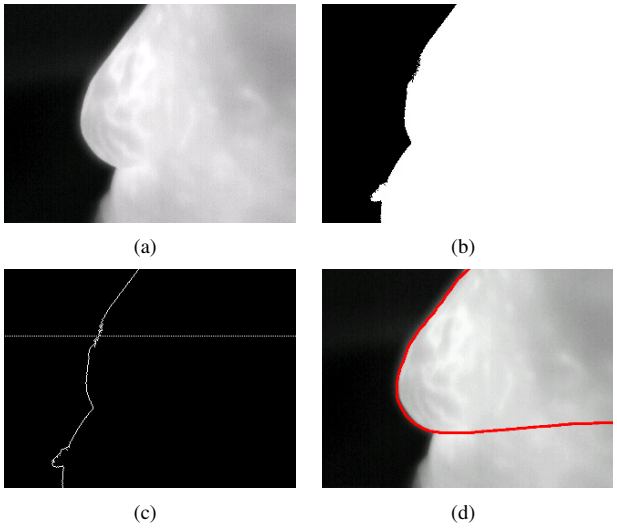


Figura 8: Avaliação visual do método considerando uma típica limitação. Em (a) a imagem original (IR_0832_grey.jpg). Em (b) a mesma imagem após a limiarização com limiar de 150. A imagem (c) contém a linha em destaque que demarca a posição y em que o método incorretamente traçou a prega inframamária. E em (d), a marcação manual demonstrando qual seria a segmentação correta. Fonte: autor.

desempenho. Isso é demonstrado na métrica de especificidade, que apresentou o valor de 94,3%.

Finalmente, as medidas de preditividade possuem taxas próximas a 90%, o que possibilita ao método a predição potencialmente correta dos pixels pertencentes ou não a região mamária.

V. Conclusões

Nesse trabalho foi apresentada uma proposta para realizar segmentação automática de imagens infravermelhas das mamas utilizando o protocolo lateral. Este se baseia em ferramentas simples bastante conhecidas da Visão Computacional, como é o caso da Máquina de Vetores Suporte e Histograma de Gradientes Orientados. Esse processo será de suma importância para análise de imagens infravermelhas das mamas e para auxiliar em diagnósticos médicos de enfermidades nas mamas, como é o caso do câncer.

A partir dos resultados obtidos em experimentos, o resultado final aproximou-se de 90%, o que possibilita um desempenho razoável para automatizar o processo de separação entre região objeto e fundo (mama e não mama). Entretanto, abre-se uma margem razoável de melhoria, sendo essa uma das expectativas com relação ao andamento desse projeto.

Os projetos futuros, por sua vez, podem ser descritos da seguinte maneira, (i) utilizar outro meio de se assinalar a prega inframamária, ao invés de usar histograma de linha. Uma possibilidade é avaliar a eficácia do SIFT (*Scale-invariant feature transform*), (ii) considerar o uso de propostas de segmentação baseadas em Deep Learning, como é o caso de abordagens que usem Redes Neurais Convolucionais (CNNs), (iii) desenvolver uma metodologia que realize a segmentação simultânea de imagens de uma mesma pessoa em múltiplos protocolos, como frontal, lateral e outros ângulos.

Referências

[1] W C Amalu, W B Hobbins, J F Head, and R L Elliot. The biomedical engineering handbook-medical devices ans systems. *Infrared The Biomedical Engineering Handbook-Medical devices ans systems*, 3, 2006.

[2] Matheus Baffa, Deivison Cheloni, and Lucas Lattari. Segmentação automática de imagens térmicas das mamas utilizando limiarização com refinamento adaptativo. In *Anais Principais do XVI Workshop de Informática Médica*, pages 39–48. SBC, 2016.

[3] A. Chagpar. An introduction to breast cancer. (11m03s), coursera., 2018. Disponível em: https://www.coursera.org. Acesso em: 10/03/2020.

[4] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[5] Maíra Araújo de Santana et al. Breast cancer diagnosis based on mammary thermography and extreme learning machines. *Research on Biomedical Engineering*, 34(AHEAD):45–53, 2018.

[6] Hospital do Câncer de Barretos. Informação: Saiba quais são os principais tipos de câncer mais comuns no brasil, 2018. Disponível em: https://www.hcancerbarretos.com.br/82-institucional/noticias-institucional/1300-%20informacao-saiba-quais-sao-os-tipos-de-cancer-mais-comuns-no-brasil. Acesso em: 10/03/2020.

[7] Instituto Nacional do Câncer INCA. Estatísticas de câncer, 2020. Disponível em: https://www.inca.gov.br/numeros-de-cancer. Acesso em: 02/08/2020.

[8] ABC Fonseca, ESRC Rodrigues, MM Nóbrega, JOC Nobre, GJ França, and LP Silva. Estimativa para o câncer de mama feminino: e a assistência de enfermagem na prevenção. *Temas em saúde*, 16(4):14–30, 2016.

[9] GLOBOCAN. Cancer fact shets: Breast cancer, 2020. Disponível em: http://globocan.iarc.fr/old/FactSheets/cancers/breast-new.asp. Acesso em: 10/03/2020.

[10] Jay P Gore and Lisa X Xu. Thermal imaging for biological and medical diagnostics. In *Biomedical Photonics Handbook*, pages 540–553. CRC press, 2014.

[11] INCA. Tipos de câncer - câncer de mama, 2020. Disponível em: https://www.inca.gov.br/tipos-de-cancer/cancer-de-mama. Acesso em: 02/08/2020.

[12] Afsaneh Jalalian, Syamsiah Mashohor, Rozi Mahmud, Babak Karasfi, M. Saripan, and Abdul Ramli. Foundation and methodologies in computer-aided diagnosis systems for breast cancer detection. *EXCLI Journal*, 16:113–137, 02 2017.

[13] J Josephine Selle, A Shenbagavalli, N Sriraam, B Venkatraman, M Jayashree, and M Menaka. Automated recognition of rois for breast thermograms of lateral view-a pilot study. *Quantitative InfraRed Thermography Journal*, 15(2):194–213, 2018.

[14] Siva Kakileti, Geetha Manjunath, Himanshu Madhu, and Hadonahalli Ramprakash. *Advances in Breast Thermography*. 10 2017.

[15] Sebastien Mambou, Ondrej Krejcar, Petra Maresova, Ali Selamat, and Kamil Kuca. Breast cancer detection using infrared thermal imaging and a deep learning model. *Sensors*, 18:2799, 08 2018.

[16] R. Marques. Segmentação automática das mamas em imagens térmicas, 2012.

[17] Antony Morales, E. Kolosovas, Edgar Guevara, Mireya Maruris Reducindo, Alix Hernández, Manuel García, and Francisco Gonzalez. An automated method for the evaluation of breast cancer using infrared thermography. *EXCLI Journal*, 17:989–998, 10 2018.

[18] L. Motta. Obtenção automática da região de interesse em termogramas frontais da mama para o auxílio à detecção precoce de doenças., 2010. Dissertação de Mestrado, Universidade Federal Fluminense, Instituto de Computação, Niterói, RJ, Brasil.

[19] J. Oliveira. Extração automática de região de interesse em imagens térmicas laterais da mama., 2012. Dissertação de Mestrado, Universidade Federal Fluminense, Instituto de Computação, Niterói, RJ, Brasil.

[20] Proeng. Image processing and image analyses applied to mastology., 2012. Disponível em: http://visual.ic.uff.br/en/proeng/. Acesso em: 10/03/2020.

Fundus Eye Images Classification for Diabetic Retinopathy Detection Using Very Deep Convolutional Neural Network

Ítalo Rodrigues Gama*, Alessandra Martins Coelho *, Matheus de Freitas Oliveira Baffa†
*Instituto Federal de Educação, Ciência e Tecnologia do Sudeste de Minas Gerais (IF Sudeste MG)
Rio Pomba, MG, Brasil
†Universidade de São Paulo (USP)
Ribeirão Preto, Brasil
Email: itrgama@gmail.com, alessandra.coelho@ifsudestemg.edu.br, mfreitas826@gmail.com

***Abstract*—Diabetic retinopathy is an anomaly responsible for causing microvascular and macrovascular damage to the retina and occurs as a consequence of the worsening of diabetes. According to the World Health Organization (WHO), diabetic retinopathy is the most common cause of avoidable blindness in patients with diabetes worldwide. Early detection is important for the efficiency of treatments. Fundus Eye Image can be used to identify early disease development and monitor the patient's clinical condition. The diagnostic process using this type of image may require some expertise from the ophthalmologist since not all retina anomalies are clearly visible. Thus, this paper proposes the development of a classification method based on Convolutional Neural Networks, but highly dense and deeper. The proposed method obtained a total of 92% AUC in the given experiments.**

I. Introdução

Considerada como uma epidemia mundial, a diabetes é um grupo de distúrbios metabólicos que se caracteriza pela hiperglicemia causada pela ação ou secreção anormal da insulina, que leva a alterações no metabolismo de carboidratos, lipídios e proteínas [4]. Segundo a Secretaria de Saúde do Estado do Paraná, esses distúrbios, ao longo do tempo, comprometem a função e a estrutura vascular de diferentes órgãos, como o coração, os rins e os olhos [5].

A diabetes pode gerar danos significativos, tanto econômicos quanto sociais. As estatísticas indicam que o número de casos em todo mundo atualmente chega a 463 milhões, com expectativa de alcançar 700 milhões de pessoas em 2045. De acordo com dados da Federação Internacional de Diabetes, em 2019 a diabetes causou 4,2 milhões de mortes e estima-se que foi responsável por 760 bilhões de dólares gastos com saúde, cerca de 10% do gasto global [2].

A retinopatia diabética é uma das principais complicações relacionadas a diabetes e a principal causa da perda de visão em pessoas com idade entre 20 e 74 anos [13]. Ela se caracteriza por lesões nos capilares da retina, causando inicialmente vazamentos que levam a inchaços e hemorragias. Mais tarde, há a proliferação de vasos sanguíneos no interior do olho, descolamento de retina e glaucoma [9]. Após 20 anos de doença, cerca de 75% dos diabéticos apresentam algum grau de retinopatia [14].

O exame mais fundamental para que o oftalmologista possa avaliar alterações oculares, inclusive em pacientes diabéticos, é o exame de fundo de olho, ou fundoscopia. Neste exame, procura-se avaliar as estruturas do fundo de olho, dando atenção ao nervo óptico, aos vasos sanguíneos e a mácula, região central da retina [3].

Caso a retinopatia diabética seja detectada, o acompanhamento deverá ser periódico, conforme a intensidade da doença no olho e o quadro clínico geral do paciente. Quando o diagnóstico é realizado em tempo adequado e o tratamento realizado corretamente, complicações graves podem ser retardadas ou impedidas por completo, reduzindo o risco de cegueira dessa anomalia a menos de 5% [10].

A partir dos anos 80, tornou-se habitual realizar diagnósticos auxiliado por computador (CAD - *Computer-aided* +). Esses sistemas enfatizam áreas de importância e apresentam possíveis diagnósticos para alguma anomalia detectada. Geralmente, um sistema CAD é composto por um algoritmo de pré-processamento da base de dados, um algoritmo de classificação e uma interface gráfica para manuseio da aplicação [15].

Os sistemas de diagnóstico auxiliado por computador têm como objetivo auxiliar o médico especialista melhorar a consistência da interpretação da imagem utilizando a resposta do computador como referência. Dessa forma, uma dupla analise realizada pelo computador em conjunto com médico especialista pode melhorar a eficiência do diagnóstico [11].

Ao longo dos anos foram desenvolvidos diferentes métodos de classificação para utilização em imagens oftalmológicas. Destas, destaca-se (i) o método de morfologia matemática para detectar exsudatos em imagens de fundo de olho de pacientes diabéticos [25], (ii) o emprego de técnicas de segmentação de bordas para detectar vasos sanguíneos e classificar imagens com retinopatia [26], (iii) a aplicação da transformada de *wavelet* para classificação da retinopatia diabética [20] e (iv) a utilização do aprendizado profundo, através da Rede Neural Convolutiva, para extrair características das imagens de fundo de olho em conjunto de um algoritmo de árvore de decisão para realizar a identificação da retinopatia diabética [12].

O presente trabalho tem como objetivo desenvolver um método de classificação para auxiliar médicos oftalmologistas na detecção da retinopatia diabética, em qualquer grau. Para isto, será empregado um algoritmo de aprendizado de máquina baseado na abordagem do aprendizado profundo, para encontrar padrões nos exames de fundo de olho e classificá-los de acordo com o estado de saúde do paciente, entre doente e saudável.

O presente trabalho contribui à comunidade acadêmica com os seguintes achados:

- o desenvolvimento de um novo método para a padronização de bases de imagens de fundo de retina
- o desenvolvimento de uma arquitetura de Rede Neural Convolucional baseada nos modelos existentes de Redes Altamente Profundas
- a avaliação das arquiteturas atuais de Redes Convolutivas Altamente Profundas em bases não padronizadas
- o desenvolvimento de um método que integra o Pré-Processamento das Imagens e as Redes Neurais Convolucionais para realizar a classificação de imagens de fundo de retina
- uma avaliação do método proposto com demais trabalhos da literatura

O artigo está organizado como se segue. Na seção II é realizado um estudo do atual estado da arte em classificação de exames de fundo de olho para detectar a retinopatia diabética. Em seguida, nas seções III e IV são descritas as metodologias de pré-processamento e da classificação da base de dados, contribuição deste trabalho. Os experimentos realizados e os resultados obtidos são apresentados na seção V e na seção VI, encontra-se a conclusão do presente trabalho.

II. Trabalhos Correlatos

Os trabalhos a seguir tratam de metodologias para a detecção da retinopatia diabética.

A. Gargeya and Leng [12]

Neste trabalho foi desenvolvido um algoritmo de classificação de imagens de fundo de olho utilizando Redes Neurais Profundas. A base de dados EyePACS utilizada neste trabalho, possuía 75 mil imagens e eram oriundas de um projeto envolvendo especialistas em oftalmologia, que coletaram e classificaram manualmente os exames.

Devido as imagens terem sido obtidas utilizando diferentes protocolos de aquisição, a base de dados era bem heterogênea. Assim, fez-se necessário realizar uma etapa de pré-processamento para padronizar essas imagens. Para o treinamento do método de classificação, utilizou-se uma rede neural convolucional, devido à sua ampla aplicabilidade e robustez em tarefas de reconhecimento em bases de grande quantidade.

Utilizando validação cruzada, esse modelo alcançou uma área abaixo da curva ROC (*Area Under the Curve* - AUC) de 0,97, com sensibilidade e especificidade de 94% e de 98% em dados locais. Para validação externa foram utilizados os bancos de dados Messidor [7] e E-Ophtha [6], alcançando uma pontuação de 0,94 e 0,95, respectivamente.

B. Pratt et al. [18]

O modelo proposto utilizou 80 mil imagens de fundo de olho adquiridas da base de dados *Kaggle*, composta por imagens de aproximadamente seis milhões de *pixels* cada. Para o treinamento utilizou-se a rede neural convolucional (CNN) processada em uma GPU NVDIA K40c (2880 núcleos CUDA).

O conjunto de dados de imagens de fundo de olho se refere a pacientes de diferentes etnias, faixas etárias e níveis variados de iluminação, fato que afeta os valores de intensidade de *pixel* nas imagens. Para combater variações desnecessárias o autor aplicou a normalização de cores usando o pacote OpenCV. As imagens foram redimensionadas para 512x512 *pixels* devido o alto custo de processamento.

Foram separadas cinco mil imagens para validação da CNN. A execução das imagens levou cerca de 188 segundos e o resultado final da rede treinada foi 95% de especificidade, 75% de acurácia e 30% de sensibilidade.

C. Sinthanayothin et al. [22]

Neste trabalho foi proposto uma metodologia para análise de imagens de fundo de olho caracterizando a retinopatia diabética não-proliferativa. Foram utilizadas 112 imagens digitais de fundo de olho, capturadas usando uma câmera não midriática, de pacientes atendidos em um centro de triagem. Foi necessário um pré-processamento das imagens para combater o desbalanceamento de contraste, uma vez que em uma mesma imagem os níveis de contraste no centro eram superiores aos níveis de contraste nas bordas. A equalização de histograma adaptativa foi aplicada para minimizar o desbalanceamento, produzindo assim uniformidade na imagem.

O algoritmo de reconhecimento de exsudatos foi aplicado em 30 imagens de fundo de olho, das quais 21 continham exsudatos e nove estavam sem indicadores. A sensibilidade e especificidade para detecção de exsudato foram 88,5% e 99,7%, respectivamente.

D. Verma et al. [23]

Neste trabalho foi proposto um método para classificar os diferentes estágios da retinopatia diabética baseado na quantificação de vasos sanguíneos e hemorragia presente na retina. Foram utilizadas 65 imagens de fundo de olho, sendo, 30 imagens de retina normal, 23 com retinopatia diabética moderada e 12 com grave.

Primeiramente é realizada a segmentação através da diferença de contraste entre os vasos sanguíneos e o fundo. Logo após são utilizadas técnicas de análise de densidade e caixa delimitadora para detectar os exames que possuíam hemorragia. Por último a classificação dos diferentes estágios da anomalia foi realizado através da técnica de Florestas Aleatórias com base na área e perímetro dos vasos sanguíneos e nas hemorragias.

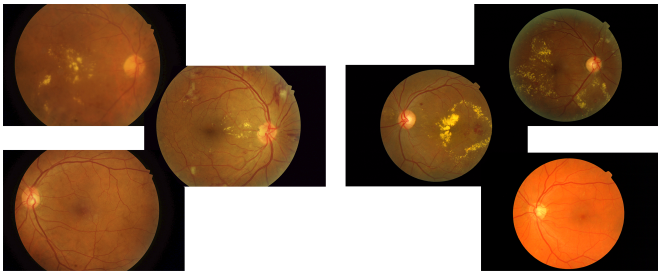
O método proposto classificou com precisão de 90% os casos normais, enquanto moderada e grave 87,5%.

III. Pré-Processamento da Base de Dados

O treinamento e a avaliação do método de classificação foi realizada utilizando duas bases de dados, o iDRIB [17] e o Messidor [7]. Ambas as bases podem ser acessadas e baixadas gratuitamente pela *internet*.

A base de dados iDRIB é composta por 516 imagens divididas em dois grupos: (i) retinas com sinais de retinopatia diabética e; (ii) retinas saudáveis sem sinais da doença. Todas as imagens foram classificadas manualmente pelos oftalmologistas de acordo com a presença ou não de anomalias. Em (a) na (Fig. 1) estão exemplos de imagens de fundo de olho obtidas através da base de dados citada anteriormente.

A Messidor por sua vez, é composta por 1200 imagens de fundo de olho coletados por três especialistas em oftalmologia. Destas, 800 imagens foram obtidas utilizando dilatação da pupila e 400 sem dilatação. As imagens nesta base de dados são disponibilizadas junto de um diagnóstico realizado pelo médico responsável pela coleta. Em (b) na (Fig. 1) estão exemplos de imagens de fundo de olho obtidas através da base de dados citada acima, é visível a diferença de padronização entre as bases de dados (a) e (b).



(a) (b)

Fig. 1. Exemplos de imagens de fundo de olho presentes nas bases de dados. Em (a), a base de dados iDRIB e em (b), a base de dados Messidor. Fonte: autor.

Ambas as bases de dados possuem amostras de retina de pacientes doentes e saudáveis. Os pacientes doentes apresentam esperadas anomalias que permitem diferenciar visualmente ambas as classes. Das anomalias presentes nas imagens doentes, destacam-se a presença de exsudatos duros e algodonosos, e de hemorragias e microaneurismas.

Além desse padrão visível, as imagens de fundo de olho são sensíveis à idade do paciente (Fig. 2). Essas podem se apresentar com um aspecto borrado, para pacientes idosos, ou com reflexos, e estruturas bem definidas e saltadas em casos de pacientes mais jovens.

Outro padrão visual presente nas imagens de fundo de olho tem relação com a raça do paciente (Fig. 3). Pacientes de pele morena ou negra tendem a possuir uma retina com pigmentação mais escura, devido a alta presença de melanina nas células que compõe esta estrutura ocular. Os pacientes de pele mais clara possuem menor pigmentação de melanina na retina que, por sua vez, se apresenta com a coloração alaranjada e/ou avermelhada.

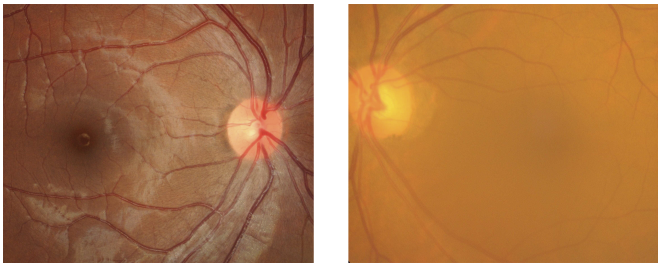


Fig. 2. A visualização da retina de um paciente jovem, com reflexo e estruturas bem definidas, e um paciente idoso com aspecto borrado e estruturas mais suaves. Fonte: autor.

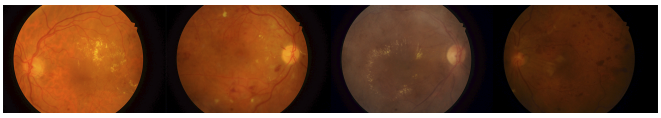


Fig. 3. Os diferentes níveis de melanina na retina, de acordo com a raça do paciente. Fonte: autor.

Tais padrões visuais que não caracterizam pacientes doentes ou saudáveis dificultam a construção de classificadores de alta eficiência. Um estudo preliminar realizado neste trabalho, utilizando diferentes arquiteturas de Redes Neurais Convolutivas, mostrou que as imagens apresentam baixas taxas de acerto, quando utilizadas sem qualquer tratamento.

A Tabela I contém a acurácia média de validação em três diferentes arquiteturas de Redes Neurais Convolutivas. Para estes testes não foram utilizadas técnicas de processamento ou melhoramento das imagens. Observa-se que, mesmo arquiteturas sofisticadas e recentes, como a *Visual Geometric Group* – VGG e a *Residual Neural Network* – ResNet não obtiveram bons resultados quando aplicadas ao problema de classificação binária de exames de fundo de olho. Então, fez-se necessário adicionar uma etapa de pré-processamento antes da construção de um classificador. O objetivo desta etapa foi padronizar a distribuição de cor entre os exames e destacar estruturas da retina que pudessem ser importantes para diferenciar os padrões saudáveis dos demais padrões relacionados a doenças.

TABELA I
Comparativo entre a acurácia de três diferentes arquiteturas de Redes Neurais Convolutivas sem o pré-processamento na base de dados. Fonte autor.

Arquitetura	Acurácia Média	Imagens Processadas?
LeNet	55%	Não
VGG	45%	Não
ResNet	51%	Não

A construção desta etapa auxiliou na criação de um método de classificação generalizado. Desta forma, tornou-se possível aplicar a presente metodologia à diferentes bases de dados, com imagens obtidas seguindo diferentes protocolos de aquisição.

A. *Visão geral do método de padronização*

A cor da retina não é uma característica importante para dizer se um paciente está doente ou saudável. Desta forma, neutralizar os efeitos que esta característica pode causar no classificador é a primeira etapa do método proposto, o qual utiliza apenas um canal de cor no formato de uma imagem em tons de cinza.

Em seguida, a imagem em tons de cinza passa por um processo de equalização do histograma e é convertida em negativo. Esta etapa auxilia na construção de imagens mais homogêneas em relação à distribuição da cor e em relação às estruturas que fazem parte da retina.

Por fim, realiza-se um recorte do interior da retina, removendo toda informação de fundo desnecessária. Esta ação também padroniza o tamanho e a informação contida nas imagens vindas de diferentes fontes de dados. Na Figura 4, um fluxograma do método de padronização desenvolvido neste trabalho é apresentado.

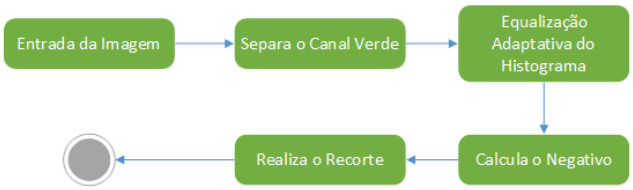


Fig. 4. Fluxograma do método de padronização das imagens de fundo de olho. Fonte: autor.

B. *Visão detalhada do método de padronização*

Nesta seção iremos detalhar as etapas que compõem o processo de padronização das bases de dados.

1) Extração do Canal Verde: A fim de reduzir os efeitos da cor da retina no processo de busca por padrões, foi proposta inicialmente a conversão da imagem para uma escala de tons de cinza. Entretanto, a partir de um estudo dos canais de cores, notou-se que o canal de cor verde apresentava visualmente informações mais ricas e detalhadas acerca da retina, das suas estruturas adjacentes e dos sinais patológicos. Desta forma, em vez de de empregar a tradicional conversão do espaço de cor RGB para tons de cinza, os canais de cores da imagem foram separados e a extração do canal verde foi utilizado como método de conversão da imagem colorida para tons de cinza.

A Figura 5 possui um exemplo de conversão de um exame em tons de cinza utilizando tanto o método tradicional quanto a separação dos canais de cores. Observa-se que, utilizando o canal verde é possível obter mais detalhes sobre as estruturas da retina, como o disco óptico e os vasos sanguíneos, além de destacar as anomalias presentes, como os exsudatos duros, algodonosos e as hemorragias.

2) Equalização Adaptativa de Histograma: Após a etapa de conversão da imagem colorida para escala de tons de cinza, as imagens continuaram se apresentando heterogêneas, dado ao fato de que algumas imagens eram mais escuras e outras mais claras. Desta forma, as imagens foram submetidas a um processo de equalização do histograma.

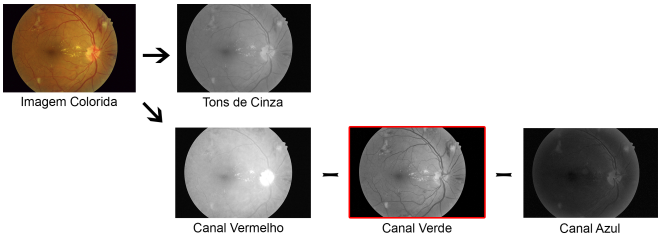


Fig. 5. Conversão da imagem colorida em tons de cinza extraindo canais de cor. Fonte: autor.

Durante o processo de equalização de histograma das imagens, notou-se o desenvolvimento de regiões, hora muito escuras, hora muito claras. A equalização resolvia o problema de homogeneização da base de dados, porém diminuía a qualidade da informação contida nas imagens. Para contornar tal problema, foi utilizado o método de Equalização de Histograma Adaptativa Limitado pelo Contraste (*Contrast-Limited Adaptive Histogram Equalization* – CLAHE) [28]. Este método realiza a equalização baseada em pequenas regiões da imagem, além de limitar o contraste local, evitando que determinadas regiões fiquem ou claras ou escuras em demasia.

3) Cálculo do negativo: Outra medida para auxiliar na homogeneização da base de dados foi a conversão das imagens em seu negativo. Desta forma, destacou-se as anomalias presentes na retina, deixando-as mais escuras. Já as estruturas da retina, como a mácula e os vasos sanguíneos, ficaram de coloração mais clara e menos evidentes.

4) Recorte da região de interesse: Embora ambas as bases de dados possuam a mesma região da retina, as imagens possuem diferentes recortes. Por exemplo, na base de dados iDRIB as imagens são ampliadas e cortadas na região inferior e superior, enquanto na base Messidor as imagens são menores, porém sem cortes. Isso ocorre porque ambas seguem padrões distintos de aquisição de imagens. Assim, para gerar uma base de dados homogênea, foi proposta uma etapa de recorte automático da região interior-central da retina, onde geralmente ocorre a presença das anomalias que permitem a classificação das imagens. Após o recorte todas as imagens ficaram com a resolução de 1100 *pixels* por 800 *pixels*.

A Figura 6 possui uma sequência que ilustra os passos da padronização. Observa-se que, independente da base de dados, as imagens finalizam de forma similar e com estruturas e anomalias de fácil identificação.

IV. *Desenvolvimento da Rede Neural Convolucional Altamente Profunda*

As metodologias tradicionais de Visão Computacional geralmente são divididas em três etapas, sendo elas (i) o pré-processamento da base de dados, com utilização de técnicas de Processamento de Imagens para realçar características ou até mesmo remover ruídos; (ii) a Análise de Imagens, através da extração de características e representação dessas imagens em um vetor descritor e; (iii) a aplicação de algoritmos de Aprendizado de Máquina para reconhecimento de padrões.

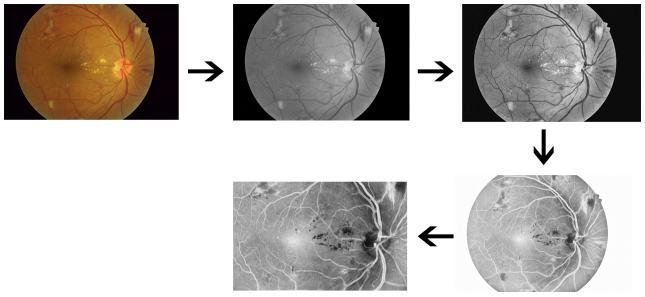


Fig. 6. Esquema visual de padronização da base de dados. Fonte: autor.

Atualmente, é comum ver trabalhos utilizando Redes Neurais Artificiais (*Artificial Neural Network* – ANN) para o reconhecimento de padrões em imagens médicas [19], [16], [8]. De modo geral, este algoritmo, combinado com uma extração de características efetiva, obtém resultados melhores que outras técnicas de Aprendizado de Máquina, como as Máquinas de Vetor Suporte (*Support Vector Machine* - SVM) ou as Árvores de Decisão.

Devido ao avanço nas tecnologias de processamento massivo paralelo utilizando placas gráficas (*Graphics Processing Unit* – GPU), abordagens de Redes Neurais Artificiais, como o Aprendizado Profundo, se fizeram possíveis. Com a alta disponibilidade de dados e poder de processamento, as Redes Neurais Profundas podem detectar padrões complexos através da associação de padrões menores combinados em suas diversas estruturas internas da Rede Neural.

A Rede Neural Convolutiva (*Convolutional Neural Network* - CNN) é um tipo de arquitetura de Rede Neural Artificial desenvolvida segundo a abordagem das Redes Profundas. Este tipo de rede é capaz de receber uma imagem, atribuir um peso aos diversos objetos e elementos que a compõe, e detectar padrões [1].

Diferente das metodologias tradicionais de Visão Computacional, a CNN reconhece padrões em uma imagem utilizando características extraídas pelas camadas de convolução. A combinação da ordem e da quantidade dessas camadas compreende-se como sub arquiteturas de CNN.

Neste trabalho foi desenvolvida uma sub-arquitetura de CNN baseada na abordagem das Redes Convolutivas Altamente Profundas (*Very Deep Convolutional Neural Network* – VD-CNN). Uma característica dessas redes é a possibilidade de encontrar padrões mais complexos e mais sensíveis. É comprovado que esta metodologia é capaz de superar a performance das Redes Convolutivas Tradicionais ou que seguem o modelo de LeNet [21]. Uma comparação da arquitetura tradicional da CNN com a CNN Altamente Profunda pode ser vista na Figura 7.

A arquitetura de rede neural utilizada neste trabalho (Fig.8) foi desenvolvida baseada na sequência de extração de características da LeNet, contendo mais profundidade baseada na VGG. Nesta arquitetura foram utilizados três grupos de convolução, cada um possuindo, respectivamente, duas camadas de convolução 2D, de 64, 128 e 256 filtros de tamanho

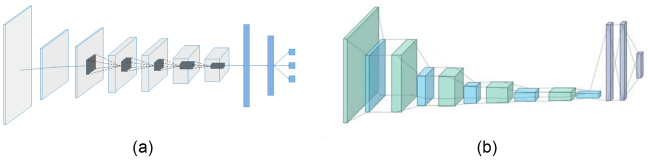


Fig. 7. Diferentes sub arquiteturas de CNN. Em (a) o modelo de LeNet, em que cada camada de convolução antecede uma camada de *Pooling*. Em (b) o modelo da VGG, no qual existem mais camadas de Convolução antecedendo as camadas de *Pooling*. Fonte: [24], [27], adaptado.

3x3, seguidas por uma camada de *MaxPooling* de tamanho 2x2. Por fim, as características são enviadas para uma Rede Neural Totalmente Profunda (*Fully Connected Neural Network* – FCNN) de quatro camadas de 4096 neurônios cada. Nas camadas internas da rede neural foram utilizadas a função de ativação ReLU (*Rectifier Linear Unit*) em conjunto com o algoritmo de otimização Adadelta. Na última camada, por se tratar de uma classificação binária, foi utilizada a função de ativação Sigmoid. Todas as camadas FCNN foram seguidas por uma camada de *Dropout*, com uma taxa de desligamento de 20%.

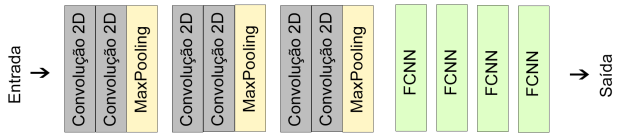


Fig. 8. Arquitetura da Rede Neural Convolutiva desenvolvida. Fonte: autor.

V. Resultados e discussão

A fim de avaliar o método proposto de classificação, os experimentos realizados foram guiados seguindo o protocolo de *Holdout*. Neste protocolo, 70% da base de dados é utilizado como treinamento enquanto os 30% restantes são utilizados como teste e validação. Desta forma, o modelo de classificação treina em uma base de dados e é avaliado em amostras diferentes dispostas em um *subset*.

Para comparação e validação do método em relação aos demais trabalhos da literatura, foram calculadas as seguintes unidades métricas: acurácia, AUC e sensibilidade. A AUC é uma métrica largamente utilizada quando existe diferença entre o número de amostras de cada classe contidas na base de dados. De forma mais sensível à acurácia tradicional, esta métrica avalia a eficiência do método como um todo e, portanto, foi a métrica escolhida neste projeto para maximização.

De modo geral, as imagens de fundo de olho, devido aos seus mais variados padrões, representam um grande desafio. As metodologias de Visão Computacional disponíveis na literatura apresentam resultados satisfatórios quanto à classificação destas imagens. A Tabela II contém um comparativo do método proposto com os demais trabalhos apresentados na Seção 2 deste trabalho. A metodologia proposta aqui, comparada ao atual estado da arte em classificação da retinopatia

diabética, apresentou resultados competitivos de classificação além de fornecer uma sequência de padronização que possa ser utilizada posteriormente em outras metodologias.

TABELA II
COMPARATIVO ENTRE O MÉTODO PROPOSTO E OS DEMAIS MÉTODOS DE CLASSIFICAÇÃO DA RETINOPATIA DIABÉTICA. FONTE AUTOR.

Autor	ACC	AUC	SEN	ESP
Sinthanayothin et al. [22]	-	-	88%	99%
Verma et al. [23]	90%	-	-	-
Pratt et al. [18]	75%	-	-	95%
Gargeya et al. [12]	-	97%	94%	98%
Método Proposto	**84%**	**92%**	**79%**	**-**

VI. CONCLUSÃO

A diabetes é uma doença considerada epidemia mundial. A retinopatia diabética é uma das principais complicações da diabetes quando não tratada adequadamente. O desenvolvimento de tecnologias que auxiliam no diagnóstico e acompanhamento desta doença é de considerável importância para evitar o desenvolvimento de cegueira no paciente.

Neste trabalho foi desenvolvida uma metodologia de classificação de imagens de fundo de olho. Dada a grande diversidade de bases de dados e as dificuldades dos classificadores tradicionais em lidar diretamente com essas imagens, uma etapa de processamento de imagens é proposta com intuito de realçar características e padronizar a informação enviada ao classificador. Obteve-se pela atual metodologia, um modelo de classificação com satisfatória eficácia no diagnóstico da retinopatia diabética.

Em trabalhos futuros, pretende-se um desenvolver uma aplicação para o apoio ao diagnóstico que possa ser utilizado, por exemplo, em lugares remotos ou onde há carência nos serviços de atenção secundária. Novas metodologias de classificação e de padronização também podem ser desenvolvidas.

REFERÊNCIAS

[1] Data Science Academy. Deep learning book, 2019. Disponível em: http://www.deeplearningbook.com.br/. Acesso em: 15/03/2020.

[2] IDF Diabetes Atlas. 9th, 2019.

[3] Kierstan Boyd. Diabetic retinopathy diagnosis, 2019. Disponível em: https://www.aao.org/eye-health/diseases/diabetic-retinopathy-diagnosis. Acesso em: 08/03/2020.

[4] Sociedade Brasileira de Diabetes. Diretrizes da sociedade brasileira de diabetes, 2020. Disponível em: https://www.diabetes.org.br/. Acesso em: 20/03/2020.

[5] Secretaria de Saúde do Estado do Paraná. Linha guia de diabetes mellitus, 2018., 2018. Disponível em: http://www.saude.pr.gov.br/arquivos/File/linhaguiadiabetes2018.pdf. Acesso em: 08/03/2020.

[6] Etienne Decenciere, Guy Cazuguel, Xiwei Zhang, Guillaume Thibault, J-C Klein, Fernand Meyer, Beatriz Marcotegui, Gwénolé Quellec, Mathieu Lamard, Ronan Danno, et al. Teleophta: Machine learning and image processing methods for teleophthalmology. *Irbm*, 34(2):196–203, 2013.

[7] Etienne Decenciere, Xiwei Zhang, Guy Cazuguel, Bruno Lay, Béatrice Cochener, Caroline Trone, Philippe Gain, Richard Ordonez, Pascale Massin, Ali Erginay, et al. Feedback on a publicly distributed image database: the messidor database. *Image Analysis & Stereology*, 33(3):231–234, 2014.

[8] BS Divya, Kamalraj Subramaniam, and HR Nanjundaswamy. Human epithelial type-2 cell image classification using an artificial neural network with hybrid descriptors. *IETE Journal of Research*, 66(1):30–41, 2020.

[9] Hospital dos Olhos. Informação: Retinopatia diabética., 2019. Disponível em: https://www.sadalla.com.br/index/especialidades-oftalmo logia/retina-tratamento-cirurgia/. Acesso em: 08/03/2020.

[10] Frederick L Ferris. How effective are treatments for diabetic retinopathy? *Jama*, 269(10):1290–1291, 1993.

[11] SS Furuie, MA Gutierrez, NB Bertozzo, JCB Figueriedo, and M Yamaguti. Archiving and retrieving long-term cineangiographic images in a pacs. In *Computers in Cardiology 1999. Vol. 26 (Cat. No. 99CH37004)*, pages 435–438. IEEE, 1999.

[12] Rishab Gargeya and Theodore Leng. Automated identification of diabetic retinopathy using deep learning. *Ophthalmology*, 124(7):962–969, 2017.

[13] R Klein. Epidemiology of eye disease in diabetes. *Diabetes and Ocular Disease*, 2000.

[14] Ronald Klein, Barbara EK Klein, and Scot E Moss. Visual impairment in diabetes. *Ophthalmology*, 91(1):1–9, 1984.

[15] RS Marques. Segmentação automática das mamas em imagens térmicas. *Master's thesis, Instituto de Computação, Universidade Federal Fluminense, Niterói, RJ, Brasil*, 2012.

[16] A Nithya, Ahilan Appathurai, N Venkatadri, DR Ramji, and C Anna Palagan. Kidney disease detection and segmentation using artificial neural network and multi-kernel k-means clustering for ultrasound images. *Measurement*, 149:106952, 2020.

[17] Prasanna Porwal, Samiksha Pachade, Ravi Kamble, Manesh Kokare, Girish Deshmukh, Vivek Sahasrabuddhe, and Fabrice Meriaudeau. Indian diabetic retinopathy image dataset (idrid): A database for diabetic retinopathy screening research. *Data*, 3(3):25, 2018.

[18] Harry Pratt, Frans Coenen, Deborah M Broadbent, Simon P Harding, and Yalin Zheng. Convolutional neural networks for diabetic retinopathy. *Procedia Computer Science*, 90:200–205, 2016.

[19] Prachi R Rajarapollu, Debashis Adhikari, and Nutan V Bansode. Use of artificial neural network for abnormality detection in medical images. In *Optimization in Machine Learning and Applications*, pages 1–12. Springer, 2020.

[20] Poonam M Rokade and Ramesh R Manza. Automatic detection of hard exudates in retinal images using haar wavelet transform. *eye*, 4(5):402–410, 2015.

[21] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[22] Chanjira Sinthanayothin, James F Boyce, Tom H Williamson, Helen L Cook, Evelyn Mensah, Shantanu Lal, and David Usher. Automated detection of diabetic retinopathy on digital fundus images. *Diabetic medicine*, 19(2):105–112, 2002.

[23] Kanika Verma, Prakash Deep, and AG Ramakrishnan. Detection and classification of diabetic retinopathy using retinal images. In *2011 Annual IEEE India Conference*, pages 1–6. IEEE, 2011.

[24] Guangfen Wei, Gang Li, Jie Zhao, and Aixiang He. Development of a lenet-5 gas identification cnn structure for electronic noses. *Sensors*, 19(1):217, 2019.

[25] Daniel Welfer, Jacob Scharcanski, and Diane Ruschel Marinho. A coarse-to-fine strategy for automatically detecting exudates in color eye fundus images. *computerized medical imaging and graphics*, 34(3):228–235, 2010.

[26] Doaa Youssef and Nahed H Solouma. Accurate detection of blood vessels improves the detection of exudates in color fundus images. *Computer methods and programs in biomedicine*, 108(3):1052–1061, 2012.

[27] Yufeng Zheng, Clifford Yang, and Alex Merkulov. Breast cancer screening using convolutional neural network and follow-up digital mammography. In *Computational Imaging III*, volume 10669, page 1066905. International Society for Optics and Photonics, 2018.

[28] Karel Zuiderveld. Contrast limited adaptive histogram equalization. In *Graphics gems IV*, pages 474–485. Academic Press Professional, Inc., 1994.

Fully-Connected Neural Network for COVID-19 Chest X-Ray Imaging Classification Using Hybrid Features

Victor Hugo Viveiros*, Rayanne Bertolace Lima *, Fernando Lucas de Lima Martins *,
Alessandra Martins Coelho * and Matheus de Freitas Oliveira Baffa†
*Federal Institute of Education, Science and Technology of Southeast of Minas Gerais (IF SudesteMG)
Rio Pomba, MG, Brazil
†University of São Paulo (USP)
Ribeirão Preto, Brazil
Email: vh.viveiros@gmail.com, rayanne_lima2010@hotmail.com, fernandolucas561@gmail.com,
alessandra.coelho@ifsudestemg.edu.br, mfreitas826@gmail.com

***Abstract*—Discovered on 31st December of 2019, the new Coronavirus has a high transmission capacity and was considered pandemic by the World Health Organization. In only six months is was able to spread all over the world and cause more than 600 thousand deaths. Early diagnosis is essential for governments to take public policies, such as social isolation, commerce control, and contact tracking. In order to make these actions, massive tests are required. On the other hand, diagnosis kits are expensive and not accessible to everyone. Medical imaging, such as thoracic x-ray and Computational Tomography (CT) has been used to visualize the lung and to verify at the first moment the presence of viral pneumonia. However, some countries have few radiologists specializing in chest x-ray analysis. The findings in the image are generally not so easy to see and can easily be confused with traditional pneumonia findings. For this reason, studies in Computer Vision are necessary, both to detect anomalies in imaging and to differentiate the other types of pneumonia. This paper addresses the initial results of a research, which developed an image classification methodology to differentiate x-ray images from sick patients, infected with Coronavirus, and healthy patients. The proposed method, based on the extraction and detection of patterns in texture and color features through a Deep Neural Network, obtained an average accuracy of 95% following a k-fold cross-validation experiment.**

***Index Terms*—coronavirus, color-features, deep learning, x-ray**

I. Introduction

SARS-CoV-2, popularly known as Coronavirus, is a virus from Coronaviridae family. It was first identified at the end of 2019 by Wuhan authorities, located in central China. The first reports recognized it as pneumonia without an already known identifiable cause. Soon after, they realized that simple proximity could contribute to the infection, as the virus spreads through droplets from coughing or sneezing of infected people, as well as contact with contaminated areas and surfaces. Then, due to its high spread capability, the virus ended up spreading rapidly to many continents, which resulted in its characterization as a pandemic disease on March, 11th by World Health Organization (WHO) [1].

More than twelve million cases of Coronavirus have been recorded worldwide so far. Also, around six million and 200 thousand patients are still sick. The lethality rate of the disease is approximately 7% and has caused a total of 600 thousand deaths [2].

This virus is responsible for the infectious disease COVID-19, in which the symptoms appear similar to a simple cold, such as dry cough, fever, runny nose, and sore throat, but may quickly evolve to a severe respiratory picture, similar to Severe Acute Respiratory Syndrome (SARS) and Middle East Respiratory Syndrome (MERS), which occurred in 2002 and 2012, respectively [3]. In this last case, the patient may have serious breathing difficulties, with the need for intubation for adequate health treatment.

The diagnosis is made in patients with symptoms characteristic of the disease. The diagnosis process is done through the collection of respiratory materials, performing an aspiration of the airways or sputum induction which is subjected to molecular biology exams in order to check for the presence of viral RNA. In addition to these, rapid exams can be an alternative for patients after a certain infection period [4].

Due to the high demand and the lack of testing kits, the use of medical imaging tests can assist doctors in the detection of lung injuries in patients with a certain degree of infection. In this case, the most recommended tests to assist the visualization of the lung are the chest x-ray and Computed Tomography (CT) exams [5].

The diagnosis of x-ray images is performed in order to find signs of abnormalities in contrast to the pattern of a healthy patient. In these cases, images from sick patients show regions with an opaque bilateral irregular effect and with a frosted-glass aspect. However, these clinical findings may be similar to other types of pneumonia, making it difficult to differentiate patients with the new Coronavirus and patients with other viral pneumonia [6].

Therefore, for the search for signs of abnormalities in the image to be effective and to have good efficacy, it must be

performed by a radiologist who specializes in chest x-rays. However, although this is a category of medical images with easy access, since the equipment is easily found in large and medium-sized centers, in some countries there is a low number of specialized professionals to perform this task, which results in the need for more professionals have to carry out such classification.

The classification of medical images, between sick and healthy patients, is commonly addressed in the Computer Vision literature applied to Health [7]. Diagnostic aid systems that use Artificial Intelligence have proven to be a powerful ally in the accurate diagnosis of diseases [8].

Computer Vision is a field of study in Computer Science that unites the Analysis and Processing of Images, from Computer Graphics to Artificial Intelligence in order to recognize elements of a scene and extract new knowledge about it. Applied to health sciences, it can assist doctors in the construction of scientific visualization systems and computer-aided diagnosis systems.

With the development of Computer Vision studies, in the context of the new Coronavirus pandemic, it is possible to develop methods that differentiate sick patients from patients with other pulmonary infections by helping general radiologists to interpret the exams indicating possible injuries.

Thus, the objective of this paper is to develop a methodology for classifying x-ray images, using Computer Vision techniques, in order to assist doctors in detecting findings in the image, indicating when a sample comes from a sick, infected patient Coronavirus, or a healthy patient. Despite the researches involving Deep Learning to distinguish healthy patients from those who have had Coronavirus, this paper stands out due to the innovative proposal of using chest x-ray imaging using hybrid features in a classifier based on Deep Neural Networks. Also, the proposed paper brings a study on how effective color features can lead the diagnosis of the new Coronavirus and to improve the results by using combined types of features.

This paper is organized as follows. In Section II we bring related papers for chest x-ray classification. Section III describes the dataset used, the preprocessing stage, and the Neural Network design. Section IV describes the experiments made in order to validate the proposed method. And finally, Section V concludes this work.

II. Related Papers

Some authors have addressed the problem of binary image classification of the new Coronavirus. In this section, we will consider works that classify x-ray images.

Narin et al. [9] developed an automatic detection system as a diagnostic alternative to COVID-19 using x-ray exams. Using chest X-ray, obtained from an open-source GitHub repository shared by Cohen et al. [10], and another 50 chest x-rays from Kaggle, three different models based on Convolutional Neural Networks (ResNet50, InceptionV3, and InceptionResNetV2) were used to detect patients infected by COVID-19. The authors ran their experiment for over 30 epochs. By the end of the experiment, the ResNet50 model demonstrated a faster training process than the other models and provided the highest classification performance, with 98% accuracy.

Apostolopoulos et al. [11] evaluated the performance of Convolutional Neural Network architectures using Transfer Learning. It collected a database from x-ray images available in public medical repositories for the experiment. The dataset contains 1428 x-ray images, including 224 from patients diagnosed with COVID-19, 700 with common bacterial pneumonia, and 504 in healthy conditions. The x-ray images were resized to 200x266, and in exams with different pixel proportions, a dark background set up to 1:1.5 ratio was added to obtain the 200x266 scale. The proposed method got 96.78%, 98.66%, and 96.46% as its best precision, sensitivity, and specificity.

Also, Ozturk et al. [12] suggested that the use of advanced Artificial Intelligence techniques and x-ray images could help them to detect COVID-19. Their goal was to overcome the problem of the lack of specialized radiologists, mainly in remote villages. The x-ray images were obtained from the image base developed by Cohen et al. [10], using images from various open access sources. The proposed model developed provides accurate diagnoses for binary (COVID-19 vs. Healthy) and multiclass classification (COVID-19 vs. Healthy vs. Pneumonia), getting the results of 98.08% accuracy for binary classes and 87.02% for the multiclass approach.

Hemdan et al. [13] introduced a new Deep Learning framework called COVIDX-Net, to assist radiologists in the diagnosis of COVID-19 through x-ray images. COVIDX-Net includes seven different architectures of Deep Convolutional Neural Network architectures, such as VGG19, DenseNet121, InceptionV3, ResNetV2, Inception-ResNet-V2, Xception, and MobileNetV2. Each Deep Neural Network is capable of analyzing the normalized intensities of the x-ray image to classify the patient's status in the negative or positive COVID-19 case. Among all tested classifiers, the accuracy of InceptionV3 model was the worst with 50%, while the VGG19 and DenseNet201 models achieved the best values of accuracy (90%).

In Elasnaoui et al. [14] it was performed a comparative study between Deep Learning models to deal with the detection and classification of COVID-19 using x-ray images, Computed Tomography (CT). The experiments found that the use of InceptionResnetV2 and Densnet201 provides better results compared to other models used, with 92.18% accuracy for InceptionResNetV2 and 88.09% accuracy for Densnet201.

Wang et al. [15] introduces COVID-Net, a Deep Convolutional Neural Network specialized for detecting COVID-19 cases from chest x-ray images. In the tests, the proposed model performed better than the other analyzed, achieving 93.3% of accuracy, while the VGG-19 and ResNet-50 get 83% and 90.6%. The work also presents a dataset of 13,975 chest x-ray images from 13,870 cases, with the most significant number of positive cases publicly available of COVID-19.

Different from the previous works in the literature, the proposed method brings a traditional Computer Vision classifi-

cation method with a novel Deep Learning approach in which extracted features were used to make up a dataset for pattern recognition using a Fully-Connected Neural Networks. In this work, we analyze the effectiveness of color feature in addition to a hybrid approach mixing color and textural features for pattern recognition.

III. Materials and Methods

In this section, we discuss the data acquisition process, the preprocessing performed in order to segment the region of interest and the Neural Network process of design.

All the experiments and development were performed on a computer with Ryzen 3600 processor, 16GB of RAM, SSD M.2 512GB, GPU RTX 2060 Super on the Manjaro 20 operating system. The proposed method were developed using Python 3.7 programming language and the frameworks Keras 2.3.1, Tensorflow 2.1, OpenCV 4, and Scikit-Learn 0.23.

A. Dataset

At the beginning of the disease spread, there was a lack of COVID-19 x-ray and CT images globally, making unfeasible the development of any work using image-based classifying methods. Works like Cohen et al. [10] made it possible by sharing those needed datasets on open data science repositories, allowing the development of several works.

To develop this work, we used datasets from Cohen et al. [10] and Tawsifur et al. [16], totalizing 525 chest x-ray images of patients, being them 263 COVID-related and 262 normal.

B. Preprocessing

The datasets' images dispose of different resolutions, angles, shapes, and color intensities, needing a process to normalize the data. Consequently, every image went through a preprocessing step before the feature extraction. The preprocessing step splits in (i) resizing, (ii) segmentation, and (iii) color-filtering.

In the first step, the OpenCV framework resizes the images to the 512x512 resolution, being this shape the default one through the entire code. Having a default resolution was needed to avoid incompatibilities through the preprocessing steps, and also to have a default model input shape.

The analyzes of the images shows that the patients bones would affect the classifier's final result, making it necessary to focus on the actual Region of Interest (ROI). There are many approaches for image segmentation, either automatic or manual, in this work we applied the U-Net proposed in Ronneberger et al. [17] since it got excellent results automatically. Their work proposes an automatic lung segmenter based on Neural Networks, which takes an x-ray or CT image as input and creates a segmentation mask separating the ROI from the rest of the image.

The dataset have different color distributions due to its different acquisition processes, cameras, and storage. Said that, at the end of the preprocessing, OpenCV is used to normalize the images through Histogram Equalization, to increase the contrast, and prevent misclassification.

C. Feature Extraction

The feature extraction stage involves recognizing shapes, colors, zones, contours, and others, to identify properties which allow differentiating classes, such as COVID-19 positive and normal. The incorrect choice of descriptive features can lead to misclassification methods.

The x-ray images diagnosis process revealed that COVID-19 manifests itself as pulmonary spots, which influence the Intensity Histogram (IH) of the image. After some tests using this feature as a discriminant, it demonstrated promising results with the classifier. In order to evaluate the effectiveness of a color-based descriptor and to know how much it gains when combined with other types of features on hybrid descriptors, this work extracted both Intensity Histogram and Haralick Textural Features [18].

The proposed method evaluates the classification performance using color-based descriptors and a hybrid descriptor based on textural and color features. The color-based descriptor had 255 values with the referring the intensity of a given histogram level and its belonging class. For every image in the image dataset, a line was inserted on a CSV, making up the feature extracted dataset. The background color (black color) was not considered in this context since it doesn't bring any information about the ROI.

Also, in order to improve results, a hybrid descriptor vector is proposed mixing color-based features and textural features. The textural features were extracted based on Haralick's textural descriptor. This descriptor is composed of 13 values, referring to 13 statistical metrics calculated over the co-occurrence matrix. The 13 values are, respectively, the Entropy of the Sum (ES), the Entropy (En), the Variance of the Difference (VD), the Entropy of the Difference (ED), Haralick's Correlation (HC), Maximum Correlation Coefficient (MCC), Second Angular Momentum (SAM), Contrast (Con), Correlation (Cor), Sum of Squares - Variance (SSV), Inverse Moment of Difference (IMD), Sum Average (SA), Variance of the Sum (VS). Finally, both descriptors were used during the experiments to evaluate and compare its performance.

D. Deep Neural Network Design

The architecture proposed in this paper is based on a Fully-Connected Neural Network (FCNN), a Deep Neural Network architecture that implements more hidden layers in order to detect complex patterns over the data. FCNN is a traditional Deep Neural Network architecture since it brings elements from an Artificial Neural Network (ANN) but with deep hidden layers.

The modeling of the FCNN proposed begins with the input layer. All the features extracted, as described above, are treated as inputs in the first layer. Also, the data contained in the hybrid vector are subjected to normalization, using the Standard Scaler function from Scikit-Learn, so that there is no pre-defined order of priority among the variables.

The model also had six hidden layers, with 180 neurons each. Each hidden layer used the Rectifier Linear Unit (ReLU) as an activation function. To estimate the error rate between

the predicted class and actual class, the loss function used was the Mean-Squared Error (MSE). Also, the optimizer used was the Stochastic Gradient Descent (SGD).

The output model had one single neuron, representing the binary classification. It's responsible to say if the given input belongs to a COVID-19 positive exam or if it's normal. The activation function used in this layer was the Hyperbolic Tangent activation function (Tanh). The whole Neural Network model is represented in Figure 1.

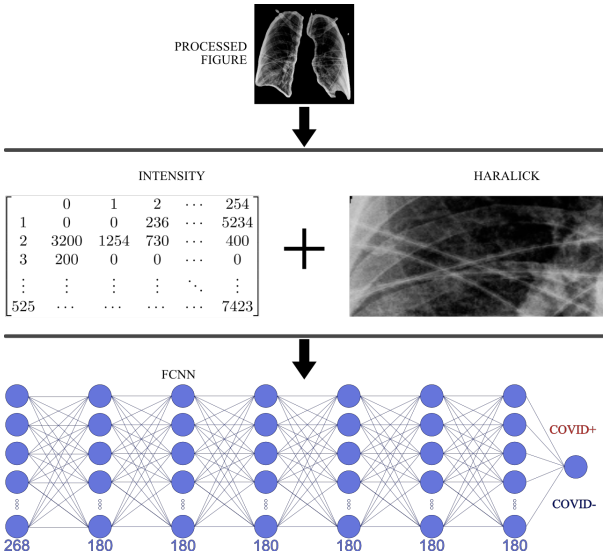


Fig. 1. Classification model for the hybrid descriptor. The input layer value 268 corresponds to the number of pixels each value the Intensity Histogram has excluding background information, which HI = 256 - 1, plus the 13 Haralick's features.

In order to avoid the occurrence of overfitting in the classification model, six dropout layers were used. They were disposed after each hidden layer and was configured to turn off 20% of the neurons of the following hidden layer. This technique helped to create a generalized classification method.

E. Hyperparameter Tuning

The Neural Network is composed of several hyperparameters. These hyperparameters, such as activation functions and optimizers are responsible to better detect patterns and create an accurate classification model.

In order to find the best hyperparameters, in this stage, we used the GridSearchCV function from the Scikit-Learn framework to test several hyperparameters combinations to find the best set. In Table I there are all the hyperparameters tested in this stage. The best set found used in the construction of the Neural Network was the Rectifies Linear Unity (ReLU) as a hidden layer's activation function, the Tanh activation function in the output layer, the SGD optimizer, the mean squared error for loss calculation, and 180 neurons on each hidden layer.

TABLE I
SET OF ALL HYPERPARAMETERS TESTED FOR SEARCH AND TUNING.

Hidden Activation	relu, elu, selu, tanh, softsign, softplus
Output Activation	sigmoid, softmax, tanh, softplus
Optimizer	adam, sgd, adadelta
Loss Function	mean_squared_error, kl_divergence, poisson, binary_crossentropy
Neurons	180, 220, 300

IV. EXPERIMENTS AND RESULTS

The cross-validation technique has been widely used in classification problems to estimate the performance of models. This protocol defines that the experiment should separate the data into the training set and the test set. There are three different protocols based on cross-validation, they are the k-fold cross-validation, holdout cross-validation, and the leave-one-out cross-validation. The choice for one of the types of validation is defined according to the size of the database and the objective of the experiment.

In order to best evaluate the method, we proposed a three-stage experiment. The first experiment evaluated the efficiency of intensity histogram to classify x-ray images between COVID-19 positive and normal images. This experiment followed the K-Fold Cross-Validation technique. The second experiment evaluated the hybrid approach, which includes the Haralick's textural feature to the intensity descriptor. The second experiment followed the Holdout Cross-Validation protocol and aimed to verify the existence of overfitting in the model. The third experiment evaluated the hybrid features as well but using K-Fold Cross-Validation. Out goal was to evaluate the method as a whole with different sets of training and validation.

To quantify the efficiency of the proposed method for each experiment, five metrics were calculated. These metrics allow us to verify the effectiveness of the model and also compare it to the other works from the literature.

A. Experiment 1: K-Fold Cross-Validation for Color-based Features

Our first concept was to develop a classification method based on Intensity Histogram. During our studies, we observed that the diagnosis was made up by analyzing the presence of certain patterns on the image. These patterns, not going into medical definitions, appear as white spots blurred in the image. This insight leads us to conduct a color-based feature investigation.

For this first test, we conducted an experiment to detect patterns over the intensity histogram. We had a medium-size dataset that allowed us to perform our experiment using the K-Fold Cross Validation protocol. This protocol says that the dataset should be split into k parts. The experiment runs over

k times and on every experiment, part of the data is used for training and part for validation. The results of this experiment can be seen in Table II.

TABLE II
RESULTS OF THE EXPERIMENT USING COLOR-BASED FEATURE ON 10-FOLD CROSS-VALIDATION.

#	Accuracy	Precision	F1-Score	Sensitivity	Specificity
0	0.8571	0.8500	0.8500	0.8500	0.8636
1	0.8571	0.8421	0.8421	0.8421	0.8696
2	0.9286	0.9200	0.9388	0.9583	0.8889
3	0.9286	0.8571	0.9231	1.0000	0.8750
4	0.8571	0.9231	0.8000	0.7059	0.9600
5	0.8810	0.8947	0.8718	0.8500	0.9091
6	0.9286	1.0000	0.9231	0.8571	1.0000
7	0.8810	0.9091	0.8889	0.8696	0.8947
8	0.9286	1.0000	0.9388	0.8846	1.0000
9	0.9512	1.0000	0.9524	0.9091	1.0000
Mean	**0.8999**	**0.9196**	**0.8929**	**0.8727**	**0.9261**

With a mean accuracy of 89%, we noted that this approach was pretty promising. After this evaluation, we tested the dataset over different other Neural Network configurations but the best result with IS stays the same. In order to improve the results, the hybrid descriptor was developed, mixing color and texture features together in order to develop a more accurate model.

B. Experiment 2: Holdout Cross-Validation for Hybrid Features

The holdout cross-validation protocol defines that the entire dataset should be separated into two parts, the training set, and the validation set. The proportion used is 80% for training and 20% for validation. Both subsets have different images in order to train and test with different contents.

Our goal with this experiment was to evaluate the method using a single execution to analyze the learning process over the epochs, get the threshold in which the method converges, and to verify if the method tends to overfit over the time.

This experiment runs for over 5000 epochs. It was observed that the Neural Network proposed quickly converged to its optimum around the epoch 100. It got an accuracy of 94.29%, a sensitivity of 94.24%, a specificity of 94.33%, a precision of 94.34% and 94.29% F-Score. The accuracy and loss were used to plot two graphs (Figure 2) representing their changes over the epochs. Note that the training and test values stay closer and stable.

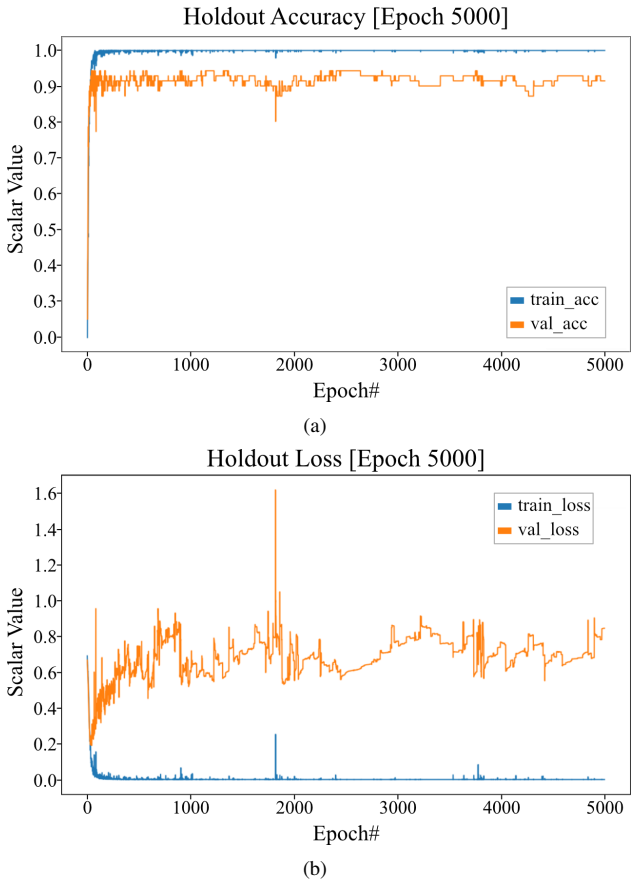


Fig. 2. Results from Holdout Cross-Validation experiment. In (a) the accuracy obtained over the epochs and, in (b) loss obtained over the epochs.

C. Experiment 3: K-Fold Cross-Validation for Hybrid Features

In order to evaluate the overall performance of the hybrid features, this third experiment was performed splitting the data into 10 parts following the K-Fold Cross-Validation. This experiment used the same architecture of Neural Network but ran for 300 epochs since it converges pretty quickly requiring fewer epochs. The results can be seen in Table III.

The overall accuracy of the proposed method using hybrid features achieved 95%. Note that comparing with the first experiment, which was used only the IH, there was an increase of 6% in the overall accuracy by using Color and Texture Features combined.

D. Comparison with other works

Most of the recent works from literature used Convolutional Neural Networks and its sub-architectures to detect pattern and classify x-ray images from COVID-19 patients. In this work, we investigate the efficiency of color-based and hybrid descriptors, bringing textural features, such as Haralick, alongside Intensity Histogram in order to analyze which one has a better performance with a Fully-Connected Neural Network.

Even with a traditional method and easy to extract features, the proposed method showed itself promising in classifying

TABLE III
RESULTS OF THE EXPERIMENT WITH HYBRID FEATURES USING 10-FOLD CROSS-VALIDATION.

#	Accuracy	Precision	F1_Score	Sensitivity	Specificity
0	0.9286	0.8696	0.9302	1.0000	0.8636
1	0.9524	0.9474	0.9474	0.9474	0.9565
2	0.9524	0.9583	0.9583	0.9583	0.9444
3	0.9762	0.9474	0.9730	1.0000	0.9583
4	0.9762	1.0000	0.9697	0.9412	1.0000
5	0.9286	0.9048	0.9268	0.9500	0.9091
6	0.9286	1.0000	0.9231	0.8571	1.0000
7	0.9524	0.9565	0.9565	0.9565	0.9474
8	0.9524	1.0000	0.9600	0.9231	1.0000
9	1.0000	1.0000	1.0000	1.0000	1.0000
Mean	**0.9548**	**0.9584**	**0.9545**	**0.9534**	**0.9579**

x-ray images between sick and healthy patients. The proposed method achieved an overall accuracy of 95%, a competitive value compared to other sophisticated methods in the literature.

In Table IV, the results obtained with the proposed method is compared to the other works in the literature, revised in Section II, which also classify x-ray images. Note that all the results obtained in this work are stable and pretty close to the others but bring an easy to extract and process method.

TABLE IV
COMPARISON BETWEEN THE PROPOSED METHOD AND THE OTHER WORKS FROM LITERATURE.

Author	ACC	PREC	F1	SENS	SPEC
Elasnaqui et al.	0.9218	0.9238	0.9207	0.9211	0.9606
Hemdan et al.	0.9000	0.9150	0.9000	0.9000	—
Narin et al.	0.9800	1.0000	0.9800	0.9600	1.0000
Ozturk et al.	0.9808	0.9803	0.9651	0.9513	0.9530
Apostolopoulos et al.	0.9875	—	—	0.9285	0.9875
Proposed Method	**0.9548**	**0.9584**	**0.9545**	**0.9534**	**0.9579**

V. CONCLUSION

COVID-19 was discovered recently but it already has become an international viral outbreak. The diagnosis of this new virus is hampered by the absence of biological kits and the delay in obtaining the results. Thus, the use of medical imaging can help doctors diagnose the patient and to better estimate the patient's contagious level. The difficulty in diagnosing the disease using x-rays is due to the fact that the findings of the Coronavirus and other viral pneumonia are quite similar. The Computer Vision systems can be used to improve diagnosis by helping the radiologist to find abnormal signs in these exams.

The proposed work brings a classification method based on hybrid features. It is shown the effectiveness of the Intensity Histogram alongside textural features, such as the Haralick descriptor. For future works, we plan to work with more complex Deep Learning architectures in order to perform a multiclassification detecting different types of pneumonia.

REFERENCES

[1] O. Brasil, "Folha informativa – covid-19 (doença causada pelo novo coronavírus)," 2020. [Online]. Available: https://www.paho.org/bra/index.php?option=com_content&view=article&id=6101:covid19&Itemid=875

[2] W. H. Organization, "Coronavirus disease situation reports," 2020. [Online]. Available: https://www.who.int/emergencies/diseases/novel-coronavirus-2019/situation-reports

[3] M. da Saúde, "Sobre o covid-19," 2019. [Online]. Available: https://coronavirus.saude.gov.br/sobre-a-doenca

[4] S. de Saúde do Estado do Paraná, "Como é feito o diagnóstico do coronavirus?" 2020. [Online]. Available: http://www.coronavirus.pr.gov.br/Campanha/FAQ/Tire-suas-duvidas-Exames

[5] A. C. of Radiology, "Acr recommendations for the use of chest radiography and computed tomography (ct) for suspected covid-19 infection," 2020. [Online]. Available: https://www.acr.org/

[6] W.-j. Guan, Z.-y. Ni, Y. Hu, W.-h. Liang, C.-q. Ou, J.-x. He, L. Liu, H. Shan, C.-l. Lei, D. S. Hui *et al.*, "Clinical characteristics of 2019 novel coronavirus infection in china," *MedRxiv*, 2020.

[7] M. d. F. O. Baffa and L. G. Lattari, "Convolutional neural networks for static and dynamic breast infrared imaging classification," in *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 2018, pp. 174–181.

[8] O. A. Paiva and L. M. Prevedello, "O potencial impacto da inteligência artificial na radiologia," *Radiologia Brasileira*, vol. 50, no. 5, pp. V–VI, 2017.

[9] A. Narin, C. Kaya, and Z. Pamuk, "Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks," *arXiv preprint arXiv:2003.10849*, 2020.

[10] J. P. Cohen, P. Morrison, L. Dao, K. Roth, T. Q. Duong, and M. Ghassemi, "Covid-19 image data collection: Prospective predictions are the future," *arXiv preprint arXiv:2006.11988*, 2020.

[11] I. D. Apostolopoulos and T. A. Mpesiana, "Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks," *Physical and Engineering Sciences in Medicine*, p. 1, 2020.

[12] T. Ozturk, M. Talo, E. A. Yildirim, U. B. Baloglu, O. Yildirim, and U. R. Acharya, "Automated detection of covid-19 cases using deep neural networks with x-ray images," *Computers in Biology and Medicine*, p. 103792, 2020.

[13] E. E.-D. Hemdan, M. A. Shouman, and M. E. Karar, "Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images," *arXiv preprint arXiv:2003.11055*, 2020.

[14] K. Elasnaoui and Y. Chawki, "Using x-ray images and deep learning for automated detection of coronavirus disease," *Journal of Biomolecular Structure and Dynamics*, no. just-accepted, pp. 1–22, 2020.

[15] L. Wang and A. Wong, "Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images," *arXiv preprint arXiv:2003.09871*, 2020.

[16] M. C. A. K. Tawsifur Rahman, Muhammad Chowdhurynovice, "Covid-19 radiography database," 2020. [Online]. Available: https://www.kaggle.com/tawsifurrahman/covid19-radiography-database

[17] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[18] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural features for image classification," *IEEE Transactions on systems, man, and cybernetics*, no. 6, pp. 610–621, 1973.

Automatic Detection of Lupus Butterfly Malar Rash Based on Transfer Learning

Jhonatan Souza*, Tiago Mota de Oliveira*, Claudemir Casa*, André Roberto Ortoncelli*†
* Departamento de Informática, Universidade Federal do Paraná, Curitiba, Brazil
†Coordenação de Engenharia de Software, Universidade Tecnológica Federal do Paraná, Dois Vizinhos, Brazil

***Abstract*—This work presents an approach to the automatic detection of Butterfly Malar Rash (BMR) in images. BMR is a Lupus symptom characterized by a reddish facial rash that appears symmetrically in the cheeks and the back of the nose. The proposed approach is based on Transfer Learning, a popular approach in Deep Learning that consists in the use of pre-trained models as the starting point for computer vision and natural language processing tasks. To perform the experiments, a database was created with images manually collected from the Instagram social network, searching for images with #butterflyrash. We evaluated the proposed approach with eight Convolutional Neural Networks (CNN) architecture. The experimental results are good results, with a precision of up to 0.957.**

***Index Terms*—Lupus diagnosis, Skin lesions, Deep Learning, Computer Vision**

I. Introduction

Lupus Erythematosus (LE) is an inflammatory and autoimmune disease, in which the body develops antibodies against its own cells, which can affect joints, skin, kidneys, blood cells, brain, heart, and lungs. The number of symptoms that can occur makes diagnosis difficult.

There is no cure for Lupus, that more severe cases can cause death, but patients can have a prolonged survival if they receive the proper treatment [1], [2]. In this context, the precise diagnosis of lupus is extremely important.

The diagnosis of Lupus is based on eleven complex criteria, ranging from clinical tests to the individual's report. These criteria were first published in 1971, but have undergone several revisions since then [3].

Based on these criteria, the diagnosis of Lupus can be slow and last for months. Therefore, develop techniques that facilitate and streamline lupus diagnosis is very important.

In this context, this work presents an approach to the automatic detect Butterfly Malar Rash (BMR) - a visual symptom of Lupus, characterized as a reddish facial rash similar to the wings of a butterfly, that appears symmetrically in the cheeks and the back of the nose.

Few studies describe computational methods for automatic detection of facial skin lesions that are symptoms of Lupus. [4], [5]. In [4], unsupervised learning was used to detect BMR in images generated artificially by Generative Adversarial Networks (GANS). In [5], Transfer Learning was explored to detect facial Malar Rash with a model trained to detect Lupus skin rashes, but not specifically BMR.

The face area analyzed is one of the main differences in our approach to the related works. In [5], only the skin lesion area is used. In [4] the entire face area is used. Our method also analyzes a large part of the face area, but unlike [4], we do not use the regions of the eyes and forehead, because the BMR does not appear in those regions.

The proposed approach to BMR detection uses Transfer Learning to transfer information from a neural network that has already been trained, to solve a different task. We use a model pre-trained with the ImageNet database [6] to create a new model trained in our database for BMR detection.

Our experimental database consists of 905 images of BMR. The database images it was manually collected from the Instagram social network, searching for images with #butterflyrash.

We need to produce our own database, because we do not find in the literature, a database that meets our needs.

To evaluate the proposed approach, experiments were performed with eight pre-trained models with eight CNN architectures: i) Resnet-50 [7]; ii) Interception V3 [8]; iii) Inception-Resnet V2 [9]; iv) Densenet [10]; v) VGG-16 [11]; vi) Xception [12]; vii) Mobilenet [13]; and viii) NasNetLarge [14].

In the experiments, we evaluate the precision of the proposed approach. The experimental results are good. The best results were obtained with Densenet-121 architecture, with a precision of 0.957.

The remaining of this article is organized as follows. Section II has related works and foundations. Details about the database and the proposed approach are in Section III. Results are in Section IV. Finally, the Section V concludes the paper.

II. Related Works and Foundations

This Section presents important concepts for understanding the proposed approach. Subsection II-A has concepts of CNN and the details of the architectures used in our approach. The Subsection II-B as Transfer Learning concepts. Subsection II-C has a literature review related to lupus automatic detection methods.

A. Convolutional Neural Networks

CNN is a Deep learning algorithm that, through an input image, can assign weights, which means that the network

Manuscript received December 1, 2012; revised August 26, 2015. Corresponding author: M. Shell (email: http://www.michaelshell.org/contact.html).

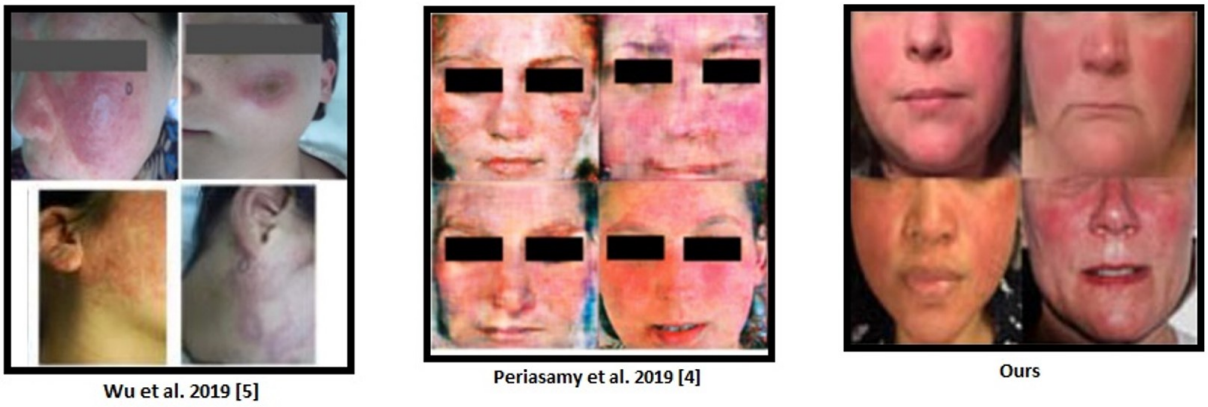


Fig. 1: Examples of the face area used by the lupus skin rash detection methods

learns in order to differentiate one from the other. This of neural network consists of an input layer that will be an image, several convolutional layers, and an output layer as a classifier.

The first use of CNN was in 1988 by Yann LeCun [15] for document recognition and since then CNN has become more complex and has been applied to a wide variety of problems, such as classifying diseases to autonomous cars and bringing great results.

CNN proved to be a great method for working with images. For over the years, CNN's has been bringing excellent results, through its models. Whether for activities such as image classification, detection and recovery [16]–[18].

One of the prominent research fields nowadays is CNN. Several applications of CNN for disease classification can be highlighted: Covid-19 [19], [20], pulmonary nodes [21], skin cancer [22], [23] and Alzheimer's [24].

Our work proposes an approach to BMR detection that uses a CNN model pre-trained to create a new model trained to detect BMR. In our experiments, we explore eight CNN architectures, that was pre-trained with the ImageNet database [6]. Characteristics of the explored CNN architectures as follows:

1) *Resnet-50:* Resnet-50 is a minor variation of ResNet 152, and has 48 layers. Being the convolutional layers: 1 max polling and 1 Average Pool layer. The Resnet architecture uses the concept of residual block, which apply shortcuts between the layers and add the values of initial inputs of the layers and the function ReLU of output [7].
2) *Inception V3:* The Inception V3 architecture has a differential through architectures called inceptions. Those who are extractors of convolutional characteristics, with the function of learning with few parameters. Inceptions modules can facilitate the mapping process between channels and spatial correlations, by factoring out the series of operations by examining them [8].
3) *Inception-Resnet V2:* The Inception-Resnet V2 architecture is a combination of Resnet and Inception, which is capable of using residual connections while maintaining the diversity of scale of the network. In order to improve results [5], [9].
4) *Densenet:* Densenet has a structure that aims to work, by adding connections between the layers. And the output results are added from the inputs to the subsequent layers. And that way the architecture can improve performance with fewer parameters [5], [10].
5) *VGG16:* Developed by Simonyan and Zisserman [11], VGG16 is an architecture that has 16 convolutional layers, and 138 million parameters. With six blocks of various layers, the first being made by a combination of layers of convolution. And the last fully connected block. The first convolution block has two layers, with 64 neurons, 3x3 convolutional filters and 2x2 max polling. And subsequent blocks increase the number of filters per block.
6) *Xception:* Xception is a combination of the Inception structural idea with the concept of depthwise separable convolutions [12], each filter will convolve with an input channel individually. In the common convolution the filter converts with all input channels, adding the results to improve performance during the convolution process by decreasing network operations during training.
7) *Mobilenet:* Proposed by Google in 2017 [13], for mobile applications, it aims to reduce the size and complexity of layers with a focus on efficiency. Like Xception, this CNN structure is based on depthwise separable convolutions to make your architecture lighter.
8) *NasNetLarge:* Unlike other architectures, Nasnet was made through recursive stages called blocks. This structure was designed to learn the ideal set of data of interest. As it is a costly approach when the data set is large, a project for a new search space was proposed [14], which allows the transfer of learning from a small dataset to a large dataset.

B. Transfer learning

Transfer Learning is a Deep Learning technique used to transfer information from a neural network that has already

been trained, for a given activity, to solve a different task. Through this knowledge transfer procedure, it can help reduce training time, improve network accuracy and work with a smaller database for new training [25].

The Transfer Learning method is the use of pre-trained neural networks to make up for the lack of training data sets. With this, trained networks are used to extract characteristics used in fine tuning, a method for adjusting the network parameters. Incrementally adapting pre-trained resources to new data [26].

In order to solve complex problems with little data, and to reduce training time, the Transfer learning technique is used frequently. This method has several uses, such as to solve problems related to image classification. Characteristic vectors are used, which are generated by a deep neural network, trained to recognize characteristics of an already trained database. After the first training, this network can be used as an entrance to a new neural network using a new database.

With the growth of the internet large databases with images are being created for the most varied applications. But there is a great lack of database for various diseases such as lupus.

Transfer Learning has been showing great results by optimizing training time and generating new models with a small amount of data.

C. Lupus Automatic Detection

Lupus is a chronic autoimmune disease that causes the immune system to attack its own tissues, the symptoms of the disease are varied and similar to other diseases causing difficulties in the diagnosis [4].

In addition to BMR, other symptoms are considered in the Lupus diagnosis, such as, photosensitivity, oral ulcers, arthritis, serositis, renal disorder, neurologic disorder, hematologic disorder, immunologic disorder, antinuclear antibody, and discoid rash skin that can appear on different body parts [27], [28].

There are several studies in the literature that explore machine learning-based approaches to the Lupus diagnosis [29], [30], but few studies are related to Malar Rash detection [4], [5] on images.

The [5]'s method, like our approach is based on transfer learning. In [5], has performed studies with a large clinical image dataset of skin diseases (including Malar Rash) from different body parts. In [4] it is demonstrated the use of artificially generated BMR images generated from Generative Adversarial Networks to train a model that differentiates Lupus from its other counter skin diseases using a Neural Network Classifier.

Despite the small amount of work related to facial Lupus Malar Rash detection on images, it is possible highlight recent works related to skin rash detection [22], [23], [31], [32], which is a concept directly related to our work.

III. Proposed Approach

We propose an approach to BMR detection based on Transfer This Section presents the proposed approach. Subsection III-A presents details about the database. Subsection III-B presents the steps of the proposed approach.

A. Database

Since we did not find literature in a public database with the necessary characteristics to perform our experiments, a manual search was necessary to produce our experimental database. The images were manually downloaded from the Instagram social network. We use the #butterflyrash to locate images with BMR.

Our experimental database, it is composed of 905 images of BMR, being 227 images of male and 678 images of female faces. The greater number of female faces is justified by the fact that lupus is more common in this gender [33].

Since our method considers only the face region, we manually segment this area in all images. We do not use the regions of the eyes and forehead, because the BMR does not appear in those regions.

B. Steps of the proposed approach

The proposed approach to detect BMR uses a classifier created with a neural network based on a pre-trained model. The training process consists of five steps. Details of each one of the steps as follow:

1) **Data acquisition**: This step represents the process that we execute to produce our database - presented in Subsection: III-A.
2) **Pre-processing**: All the images in our database it was pre-processed. The pre-processing is done with the following steps:
   a) each one image was adjusted to the a standard size: 224x224. We resize the entire image to this size, because this is the pattern of the ImageNet database.
   b) the face it was manually segmented in the image.
3) **Data augmentation**: data augmentation is applied to generate new images and increase our database [34].
4) **Pre-training**: In this step a model it was trained with the ImageNet database [6]. We performed eight experiments to validate owner approach with each one of the CNN architectures that were presented in the Subsection II-A. The ImageNet is one of the largest image databases, with 1.281.167 images divided into 1.000 classes, and a set of 50.000 images for tests.
5) **Final training**: Global Average Pooling (GAP) was used, as pre-trained networks have many parameters in the last layers. For this reason, a reduction in dimensionality was carried out. And the GAP connected to the last layer of the pre-trained model was applied. Two other 1024-d fully connected layers were connected with the ReLU activation function, and then another layer was connected 512-d by the same activation function. And finally, the last layer with 2 neurons with the softmax activation function. In order to return the probability of each of the two classes, positive or negative for the BMR.

6) **Malar Rash Detection**: the last trained model it was used to BMR in an image.

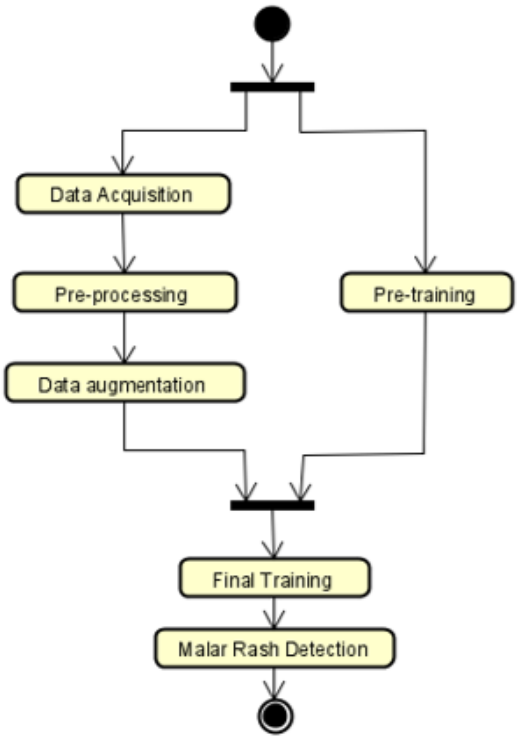


Fig. 2: Activity diagram of the proposed approach

IV. Results and Discussions

This section aims to present the results of the experiments carried out to evaluate the proposed approach. Details of the evaluation metric are in the Subsection IV-A. The experimental results as in the Subsection IV-B. Finally, the Subsection has the analysis of the results.

A. Evaluation Metrics

We use three measures to evaluate experimental results: precision (the fraction of relevant instances among the retrieved instances), recall (the fraction of the total relevant instances that were actually retrieved) and F1-Score (or f-measure, is a harmonic mean between precision and recall). Four parameters it was used to compute these measures:

- Number of True Positives (NTP): numbers of images with BMR in which BMR was detected;
- Number of False Positives (NFP): number of images with BMR in which BMR was not detected;
- Number o False Negatives (NFN): numbers of images without BMR in which BMR was detected;
- Number of True Negative (NTN): numbers of images without BMR in which BMR was not detected;

The precision, recall and F1-Score are computed with the Equations 1, 2, 3.

$$precision = \frac{NTP}{NTP + NFP} \tag{1}$$

$$recall = \frac{NTP}{NTP + NFN} \tag{2}$$

$$F1 - Score = 2 * \frac{precision * recall}{precision + recall} \tag{3}$$

B. Experimental Results

We execute eight experiments instances to evaluate the proposed approach. In each instance, it was used a different CNN architecture to train the pre-trained model. Details of each of the CNN architectures used were in Subsection II-A.

Table I as the recall, precision and F1-score of all experimental instances and also the average result.

Net	Recall	Precision	F1-Score
Resnet50	0.912	0.909	0.91
Inception-Resnet V2	0.826	0.868	0.84
InceptionV3	0.839	0.863	0.85
Densenet-121	**0.941**	**0.957**	**0.94**
Xception	0.826	0.876	0.85
NASNetLarge	0.651	0.717	0.68
VGG16	0.871	0.891	0.88
Mobilenet	0.898	0.893	0.89
Average Result	0.845	0.871	0.85

TABLE I: Experimental results

C. Analysis of results

In the mean, our experiments showed recall, precision and F1-score greater than 0.84. Densenet-121 presented results greater than 0.94 - the best experimental results.

Our results are similar to those of [5] - a recent method for cutaneous lesion detection. On average, our results were higher, which indicates that owner approach got good results.

V. Conclusion

This work presents an approach to automatically detect BMR. The approach is relevant because BMR is one of the symptoms of Lupus, which is a disease that is difficult to diagnose because lupus is a disease that is difficult to diagnose because its diagnosis is based on several criteria.

Our approach combines a strategy to obtain a set of images and a method for BMR detection based on Transfer Learning that is pre-trained with ImageNet database. The pre-trained model is used to create the first layer of the final model, that is trained with our database.

Experiments were carried out with eight models pre-trained with eight CNN architectures. The experimental results it is good, reaching an accuracy of 0.957 with the Densenet-121.

The main difficulty we find to develop this work is the lack of a large database with BMR images. For this reason, we created our database with images from the social network Instagram.

We pretend, in future works, apply our approach to develop a mobile app to streamline the process of diagnosing lupus. The results with the Mobilenet architecture, suggest that the development of this application is viable.

REFERENCES

[1] V. P. Bernardes, L. D. B. Oliveira, and C. Marcon, "Lupus eritematoso sistêmico juvenil: Diagn [ostico de doença crônica e dinâmica familiar," *Barbarói*, pp. 75–90, 2011.

[2] D. J. Wallace, *The lupus book: A guide for patients and their families*. Oxford University Press, 2019.

[3] C. Yu, M. E. Gershwin, and C. Chang, "Diagnostic criteria for systemic lupus erythematosus: a critical review," *Journal of Autoimmunity*, vol. 48, pp. 10–13, 2014.

[4] P. Periasamy and V. L. Byrd, "Generative adversarial networks for lupus diagnostics," in *Practice and Experience in Advanced Research Computing on Rise of the Machines (learning)*, 2019, pp. 1–8.

[5] Z. Wu, S. Zhao, Y. Peng, X. He, X. Zhao, K. Huang, X. Wu, W. Fan, F. Li, M. Chen *et al.*, "Studies on different cnn algorithms for face skin disease classification based on clinical images," *IEEE Access*, vol. 7, pp. 66 505–66 511, 2019.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition*. Ieee, 2009, pp. 248–255.

[7] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *International Conference on Machine Learning*, 2010.

[8] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[9] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *AAAI Conference on Artificial Intelligence*, 2017.

[10] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *IEEE Conference on Computer Vision and Pattern recognition*, 2017, pp. 4700–4708.

[11] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *International Conference on Learning Representations*, 2015.

[12] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1251–1258.

[13] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. A. Mobilenets, "Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[14] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *IEEE Conference on Computer Vision and Pattern recognition*, 2018, pp. 8697–8710.

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[16] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[18] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1915–1929, 2012.

[19] X. Wang, X. Deng, Q. Fu, Q. Zhou, J. Feng, H. Ma, W. Liu, and C. Zheng, "A weakly-supervised framework for covid-19 classification and lesion localization from chest ct," *IEEE Transactions on Medical Imaging*, 2020.

[20] S. Hu, Y. Gao, Z. Niu, Y. Jiang, L. Li, X. Xiao, M. Wang, E. F. Fang, W. Menpes-Smith, J. Xia *et al.*, "Weakly supervised deep learning for covid-19 infection detection and classification from ct images," *IEEE Access*, 2020.

[21] A. E. Jatobá, L. L. Lima, and M. C. Oliveira, "Pulmonary nodule classification with 3d convolutional neural networks," in *Anais do XV Workshop de Visão Computacional*. SBC, 2019, pp. 67–72.

[22] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, pp. 115–118, 2017.

[23] kG Utsch, C. dos Santos, and J. Samatelo, "Convolutional neural network for skin lesion classification," in *Workshop de Visão Computacional*. SBC, 2018, pp. 105–110.

[24] S. Basaia, F. Agosta, L. Wagner, E. Canu, G. Magnani, R. Santangelo, M. Filippi, A. D. N. Initiative *et al.*, "Automated classification of alzheimer's disease and mild cognitive impairment using a single mri and deep neural networks," *NeuroImage: Clinical*, vol. 21, p. 101645, 2019.

[25] R. Ribani and M. Marengoni, "A survey of transfer learning for convolutional neural networks," in *Conference on Graphics, Patterns and Images - Tutorials (SIBGRAPI-T)*. IEEE, 2019, pp. 47–57.

[26] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. Van Der Laak, B. Van Ginneken, and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, 2017.

[27] E. M. Tan, A. S. Cohen, J. F. Fries, A. T. Masi, D. J. Mcshane, N. F. Rothfield, J. G. Schaller, N. Talal, and R. J. Winchester, "The 1982 revised criteria for the classification of systemic lupus erythematosus," *Arthritis & Rheumatism: Official Journal of the American College of Rheumatology*, vol. 25, no. 11, pp. 1271–1277, 1982.

[28] M. C. Hochberg, "Updating the american college of rheumatology revised criteria for the classification of systemic lupus erythematosus," *Arthritis & Rheumatism: Official Journal of the American College of Rheumatology*, vol. 40, no. 9, pp. 1725–1725, 1997.

[29] S. Gomathi and V. Narayani, "A proposed framework using cac algorithm to predict systemic lupus erythematosus (sle)," in *World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave)*, 2016, pp. 1–6.

[30] S. Balderas-Díaz, K. Benghazi, G. Prados, and E. Miró, "Designing configurable and adaptive systems in ehealth," in *Workshop on ICTs for Improving Patients Rehabilitation Research Techniques*. New York, NY, USA: Association for Computing Machinery, 2015, p. 118–121.

[31] T. A. Rimi, N. Sultana, and M. F. A. Foysal, "Derm-nn: Skin diseases detection using convolutional neural network," in *International Conference on Intelligent Computing and Control Systems*. IEEE, 2020, pp. 1205–1209.

[32] J. Velasco, C. Pascion, J. W. Alberio, J. Apuang, J. S. Cruz, M. A. Gomez, B. Molina Jr, L. Tuala, A. Thio-ac, and R. Jorda Jr, "A smartphone-based skin disease classification using mobilenet cnn," *arXiv preprint arXiv:1911.07929*, 2019.

[33] S. Z. Y. Wasef, "Gender differences in systemic lupus erythematosus," *Gender Medicine*, vol. 1, no. 1, pp. 12–17, 2004.

[34] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019.

Automatic Detection of COVID-19 in X-Ray Images Using Fully-Connected Neural Networks

Élisson Carlos de Carvalho*, Raian Campos Malta *, Alessandra Martins Coelho *,
Matheus de Freitas Oliveira Baffa†
*Federal Institute of Education, Science and Technology of Southeast of Minas Gerais (IF SudesteMG)
Rio Pomba, MG, Brazil
†University of São Paulo (USP)
Ribeirão Preto, Brazil
Email: elissoncarvalho314@gmail.com, raianmalta71@gmail.com, alessandra.coelho@ifsudestemg.edu.br,
mfreitas826@gmail.com

***Abstract*—The coronavirus pandemic remains a problem of worldwide interest. The diagnosis of COVID-19 is difficult due to its high rate of occurrence and the limited number of test kits. Medical imaging is already widespread and has been used to quickly provide lung visualization. It's needed some expertise from the radiologist to detect elements in the image that allow differentiating the sick and healthy patterns. Therefore, our goal with this paper is to provide a computer-aided diagnosis tool to help radiologists to accurately diagnose the COVID-19 using X-Ray images. For that, a model based on Fully-Connected Neural Networks was proposed for the detection of patients infected with coronavirus, through the analysis of texture characteristics, such as Haralick and Threshold Adjacency Statistics (TAS) descriptors, extracted from chest X-Ray images. Using 10-Fold Cross-Validation, the proposed method achieved an accuracy of 98.39%, showing itself as an option to aid the disease diagnosis.**

***Index Terms*—computer vision, COVID-19, deep learning, x-ray classification**

I. Introdução

Nos últimos meses o mundo se viu enfrentando uma de suas maiores crises. Em 31 de dezembro de 2019, 27 casos inexplicáveis de pneumonia foram identificados em *Wuhan*, província de *Hubei*, China e associados aos chamados "mercados úmidos", que vendem frutos do mar e carne fresca de uma variedade de animais, incluindo morcegos e pangolins. Desde então, países da América, África, Ásia, Europa e Oceania vêm sendo afetados pelo vírus [6]. A pneumonia encontrada é causada por um vírus identificado como Síndrome Respiratória Aguda Grave CoronaVírus-2 (SARSCoV-2) [18], com a doença associada denominada de coronavírus-2019 ou simplesmente COVID-19. O vírus é transmitido por gotículas de saliva, espirros, acessos de tosse, contato próximo e superfícies contaminadas, podendo sobreviver por até cinco dias em algumas superfícies [3].

No dia 11 de março de 2020, a Organização Mundial da Saúde (OMS) [4] declarou a pandemia do novo coronavírus (COVID-19). Esse vírus, que tem como principal grupo de risco os idosos [7], infectou mais de 10 milhões de pessoas e tirou mais de 500 mil vidas no primeiro semestre de 2020, além de deixar vários países ao redor do mundo em estado de *Lockdown*. Países de primeiro mundo viram seus sistemas de saúde em total colapso, como o caso da Itália, que, em certo ponto da pandemia, priorizou dentre os pacientes, os que teriam mais chances de sobreviver, pois não havia como atender a todos.

Os relatos iniciais da infecção caracterizaram o quadro como uma pneumonia de origem desconhecida, sendo muitos pacientes tratados para pneumonia, porém sem sucesso com a implementação da terapia antibiótica usual. Os primeiros casos apresentaram opacificação mal definida na radiografia de tórax, bilateral e periférica na maioria das vezes e na Tomografia Computadorizada apresentou-se com um padrão em "vidro fosco" e zonas de mosaico. Hoje, sabemos que cerca de 59% dos pacientes apresentam alterações no exame de imagem, no entanto, uma pessoa que possui seu exame de imagem sem alterações não pode ser diagnosticada como negativo para a COVID-19, pois somente o exame de imagem sem alteração não é fator de exclusão [17].

As imagens médicas têm sido utilizadas como forma inicial de diagnóstico do coronavírus. Isso se deve ao fato do alto custo de outras formas de diagnóstico, e à a alta taxa de ocorrência da doença ao redor do planeta. Atualmente, o protocolo padrão para casos de suspeitas da doença e urgência de diagnóstico é a utilização de imagens de Tomografia Computadorizada (*Computed Tomography* – CT) ou de Radiografia do Tórax (Raio-X) [13]. A principal vantagem na utilização deste exame é o seu fácil acesso. Como são métodos utilizados no diagnóstico de grande quantidade de doenças, muitos hospitais e clínicas já possuem aporte para realizá-lo.

Diferenciar entre pacientes doentes ou saudáveis a partir de exames de imagem é um problema comumente abordado na literatura de Visão Computacional aplicada à Saúde [5]. A utilização de inteligência artificial em Sistemas de Auxílio ao Diagnóstico tem se mostrado eficaz, tornando-se uma aliada no diagnóstico preciso de doenças [14].

Diante do exposto, apresentamos neste artigo um método que utiliza uma rede de Aprendizado profundo, treinada a partir de imagens de Raio-X, para detecção da COVID-19.

II. Trabalhos Relacionados

Atualmente o mundo vem passando por uma crise sanitária devido à pandemia causada pela COVID-19. O seu diagnóstico está tipicamente associado aos sintomas de pneumonia, que podem ser revelados por testes de imagem de Raio-X. Visando mais uma ferramenta para auxiliar o diagnóstico médico, vários trabalhos têm sido realizados com o objetivo de desenvolver sistemas de classificação automática, usando principalmente *Deep Learning* [22].

Hemdan et al. [10], por exemplo, desenvolveram o COVIDx-Net, uma estrutura composta por sete arquiteturas diferentes de Redes Neurais Convolutivas (CNN). Cada uma das arquiteturas foram utilizadas para classificar imagens de Raio-X, mostrando um desempenho significativo dos modelos de *Deep Learning*, onde a VGG19 e a DenseNet201 foram as arquiteturas que apresentaram os melhores resultados, com uma acurácia de 90%.

No trabalho desenvolvido por Afshar et al. [2], foi proposta a COVID-CAPS, uma estrutura baseada em *Capsule Network* para identificação do COVID-19 usando imagens de Raio-X, alcançando uma acurácia de 95,7%.

Em Wang et al. [20] foi criado um conjunto de dados composto por 13.975 imagens de Raio-X, de 13.870 casos de pacientes de cinco repositórios de dados distintos. Esses dados foram utilizados para treinar o COVID-Net, um projeto de Rede Neural Convolucional profunda para a detecção de casos COVID-19, alcançando uma acurácia de 93%.

Utilizando a arquitetura ResNet, Abbas et al. [1] realizaram um estudo que se mostrou eficiente, tendo 95,12% de acurácia. Apresentaram soluções robustas para a classificação de casos COVID-19, com a capacidade de lidar com irregularidade de dados e com um número limitado de imagens de treinamento.

Três diferentes modelos baseados em Redes Neurais Convolucionais (ResNet50, InceptionV3 e Inception-ResNetV2) foram propostas para a detecção de pacientes infectados por pneumonia por coronavírus usando Raio-X, em Narin et al. [12]. InceptionV3 e Inception-ResNetV2 alcançaram, respectivamente, 97% e 87% de acurácia. ResNet50 apresentou melhor o desempenho, obtendo 98% de acurácia.

Já em Kasssini et al. [11] foi realizada a comparação das estruturas populares de extração de características, baseada em *Deep Learning*, para a classificação automática do COVID-19. Como componentes do aprendizado, MobileNet, DenseNet, Xception, ResNet, InceptionV3, InceptionResNetV2, VGGNet, NASNet foram escolhidos entre um conjunto de subarquiteturas de Redes Neurais Convolucionais profundas. Os recursos extraídos foram alimentados em vários classificadores de Aprendizado de Máquina, para classificar os assuntos como um caso de COVID-19 ou um caso de controle.

Essa abordagem evitou dados específicos da tarefa, métodos de pré-processamento, para suportar uma melhor capacidade de generalização de dados não vistos.

O extrator de recursos DenseNet121 com classificador de árvore de ensacamento, alcançou o melhor desempenho, com precisão de classificação de 99%. O segundo melhor resultado foi um método híbrido ResNet50, treinado pela LightGBM, com uma precisão de 98%.

III. Materiais e Métodos

Nas Figuras 1 e 2, temos, respectivamente, o Raio-X de uma pessoa com a COVID-19 e de uma pessoa saudável. Ao observá-las, podemos perceber as diferenças entre a imagens. As opacidades na Figura 1 (marcas brancas circulares), associadas a um padrão de vidro fosco, são indicativos da infecção causada pela COVID-19 [19].

Com a utilização de algoritmos de aprendizagem de máquina, busca-se encontrar padrões em uma base de dados contendo imagens de pacientes com e sem a COVID-19. Espera-se, a partir da base de dados usada e de melhorias no algoritmo, obter um diagnóstico cada vez mais preciso.

A linguagem de programação Python foi usada para treinar os modelos de aprendizado de transferência profunda propostos. Todos os experimentos foram realizados no ambiente de programação Anaconda (versão 1.7.2) utilizando o sistema operacional Windows 10 Pro de 64 bits.

O Spyder é uma IDE Python dedicada à computação matemática, integrada com o interpretador IPython, e com pacotes Numpy (álgebra linear), Scipy (processamento de imagens) e Matplotlib (plotagem 2D e 3D). Para o pré-processamento da base de dados utilizou-se a biblioteca OpenCV (versão 4.2.0.34). Já o modelo de Inteligência Artificial foi criado a partir das bibliotecas Tensorflow (versão 2.1.0) e Keras (versão 2.3.1).

Os testes foram feitos em um computador com placa gráfica GTX 960 4GB Windforce e processador Intel core I5 4460 3.4 GHz, com 16 GB de memória RAM.

A. Base de Dados

Neste trabalho, usamos um banco de dados de imagens de Raio-X do tórax para casos positivos de COVID-19, obtidas em Tawsifur et al. [15] e composto por 2905 imagens de tamanho 1024x1024 pixels, divididas em três classes (219 imagens de pacientes com COVID-19, 1.341 imagens de pacientes saudáveis e 1.345 imagens de pacientes com pneumonia viral). Para este trabalho, foram selecionadas apenas as duas primeiras classes, que estão representadas, respectivamente nas Figuras 1 e 2.

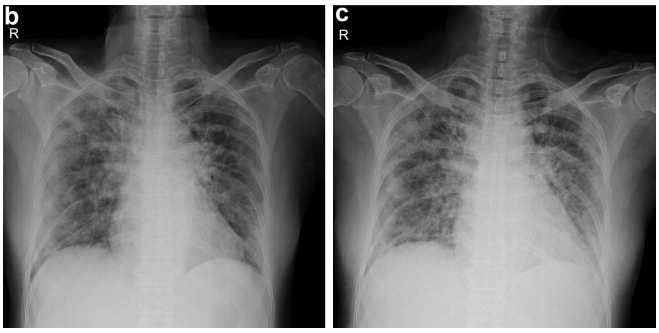


Fig. 1. Representação de imagens de Raio-X de pacientes com COVID-19. Fonte: Tawsifur et al. [15].

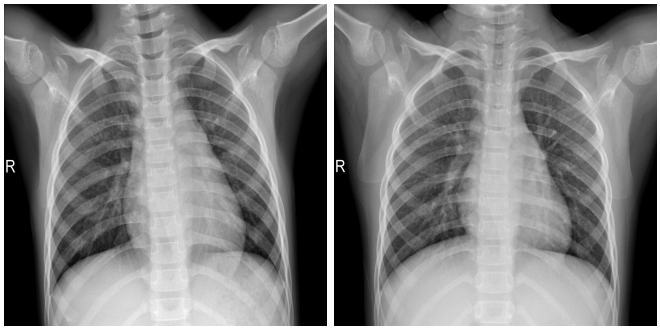


Fig. 2. Representação de imagens de Raio-X de pacientes saudáveis. Fonte: Tawsifur et al. [15].

B. Pré-Processamento da Base de Dados

No pré-processamento da base de dados, as imagens foram carregadas na escala cinza, seguidas da aplicação da técnica de processamento de imagem de equalização de histograma, com o objetivo de melhorar o aspecto da imagem como um todo, reforçando as características texturais importantes para este trabalho.

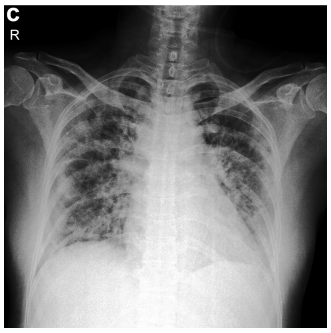


Fig. 3. Representa a mesma imagem da Figura 1, porém agora com o pré-processamento da base de dados. Fonte autor.

A equalização de histogramas é uma operação que melhora o contraste, uniformizando o histograma da imagem de forma automática, redistribuindo os níveis de cinza existentes e mapeando-os para novos níveis. Embora os picos e vales do histograma sejam mantidos, eles são deslocados após a equalização. Esse procedimento (1) faz com que o número de intensidades na imagem resultante seja igual ou menor que na imagem original.

$$Histograma : (rk), k\epsilon[0, L-1h(rk), k\epsilon[0, L-1] \quad (1)$$

C. Extração de Características

Após a etapa de pré-processamento, as imagens foram submetidas à extração de características texturais, que possibilita descrever uma imagem analisando sua textura. Existem várias técnicas para a extração dessas características. Neste trabalho foi utilizada a combinação de dois descritores de textura: o descritor de *Haralick* e o *Threshold Adjacency Statistics* (TAS), gerando um vetor com 67 características.

Haralick descreve uma imagem por meio da técnica de Matriz de Coocorrência de Níveis de Cinza (*Grey-Level Co-occurrence Matrix* - GLCM) [9]. São definidos um conjunto de 14 medidas de características, sendo algumas relacionadas com as características texturais específicas da imagem, como homogeneidade, contraste e a presença de estrutura organizada dentro da imagem, e outras que caracterizam a complexidade e a natureza das transições de tons de cinza que ocorrem na imagem.

O TAS é obtido por meio da aplicação de um limiar à imagem para criar uma imagem binária. Esse limiar é escolhido a partir de um intervalo selecionado para maximizar a diferença visual das imagens [8]. Então, para cada pixel branco, o número de pixels brancos adjacentes é contado. A primeira estatística é então o número de pixels brancos sem vizinhos brancos; a segunda é o número com um vizinho branco e assim sucessivamente até o máximo de oito. Esse procedimento resulta em um histograma do número de pixels brancos adjacentes. A partir disso, o histograma é normalizado, dividindo cada compartimento pelo número total de pixels brancos. Os valores numéricos de cada um dos nove compartimentos do histograma consiste nas estatísticas. Essas podem ser usadas como características em testes de classificação.

D. Metodologia de Classificação

Para classificar as imagens de Raio-X de pacientes infectados ou não com COVID-19, a partir das características descritas anteriormente, foi utilizada uma rede neural totalmente conectada (*Fully-Connected Neural Networks* - FCNN).

A FCNN consiste em uma série de camadas totalmente conectadas, em que cada camada é uma função do $\mathbb{R}^m$ para $\mathbb{R}^n$, onde $\mathbb{R}^{m \times n}$ é o espaço real de uma matriz $m \times n$. Cada dimensão de saída depende de cada dimensão de entrada [16]. Uma camada totalmente conectada é representada conforme apresentada na Figura 4.

A principal vantagem das redes totalmente conectadas é que elas são independentes da estrutura, ou seja, nenhuma suposição especial precisa ser feita sobre a entrada (por exemplo, que a entrada consiste em imagens ou vídeos). [16].

O modelo de FCNN proposto neste trabalho é constituído por seis camadas totalmente conectadas, com dezesseis neurônios cada. Foi definido um *dropout* de 20% após a função de ativação de cada camada. Além disso, a rede é composta por uma camada de saída com um neurônio.

A rede foi treinada por 100 épocas, tendo como parâmetros um *batch size* igual 32 e *optimizer adam*. A função de ativação usada na camada de saída foi o *Sigmoid*. Nas demais camadas, foi utilizada a Unidade Linear Retificada (*Rectified Linear Unit* - ReLU) como função de ativação. Na Figura 5 temos uma ilustração representativa dessa rede.

IV. Experimentos e Resultados

A validação cruzada por *K-Fold* é um procedimento popular para se estimar o desempenho de um algoritmo de classificação ou comparação do desempenho entre dois algoritmos de

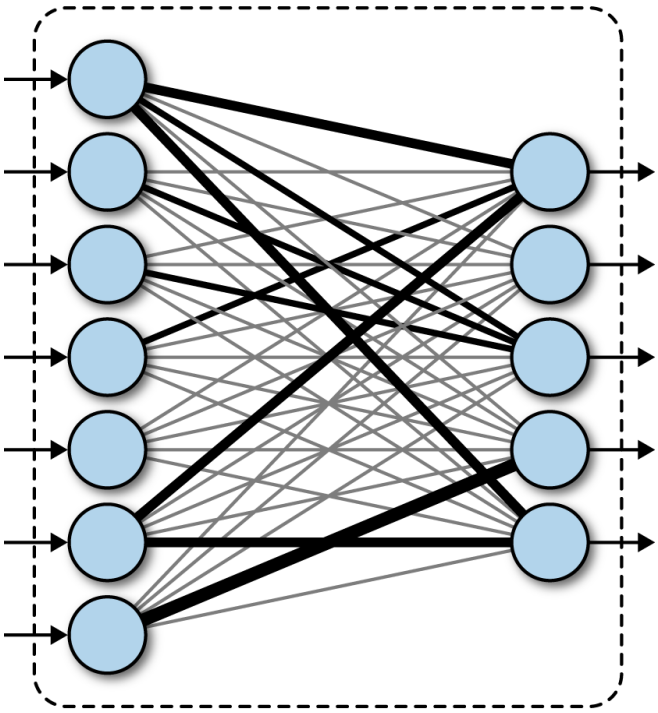


Fig. 4. Representa uma rede neural onde cada dimensão de saída depende de cada dimensão de entrada. Fonte: Ramsundar et al. [16].

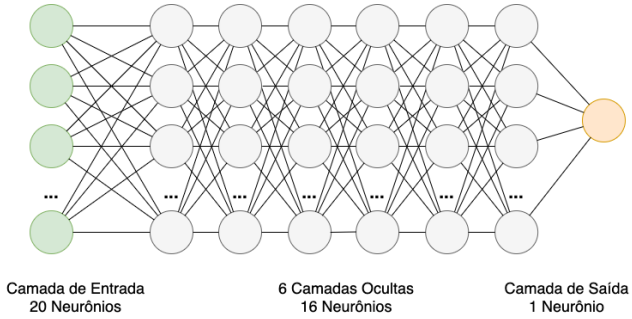


Fig. 5. Representação visual da arquitetura final da rede neural. Fonte: autor.

classificação em um conjunto de dados [21]. Esse procedimento divide aleatoriamente um conjunto de dados em k partes (*folds*), com aproximadamente o mesmo tamanho. Cada parte é usada para testar o modelo induzido das outras k - 1 *folds*, por um algoritmo de classificação. O desempenho da classificação do algoritmo é avaliado pela média das k precisões resultantes da validação cruzada.

Utilizando a validação cruzada *k-fold*, com k = 10, este trabalho avalia o desempenho do modelo proposto, a partir da seguintes métricas de performance: acurácia, sensibilidade e especificidade. Essas métricas podem ser determinadas a partir da contagem correta de exemplares de cada classe, em que Verdadeiro Positivo (VP) é o número de exemplos da classe corretamente reconhecidos; Verdadeiro Negativo (VN) é o número exemplos corretamente reconhecidos que não pertencem à classe; Falso positivo (FP) é o número de exemplos que incorretamente foram atribuídos à classe; e Falso Negativo (FN) é a quantidade de exemplos que não foram reconhecidos como pertencentes à classe.

A Acurácia (2) é a métrica mais importante para os resultados de classificadores de aprendizado profundo. Ela retorna o acerto médio por classe de um classificador.

$$Acuracia = \frac{VP + VN}{VP + FP + FN + VN} \tag{2}$$

A Sensibilidade (3) mede a eficácia do classificador para identificar os verdadeiros positivos, ou seja, ela mede a capacidade do classificador em reconhecer corretamente os volumes.

$$Sensibilidade = \frac{VP}{VP + FN} \tag{3}$$

A Especificidade (4) mede a eficácia do classificador em identificar os verdadeiros negativos.

$$Especificidade = \frac{VN}{VN + FP} \tag{4}$$

Utilizando imagens de Raio-X para treinar uma FCNN e classificar os pacientes em saudáveis ou infectados com COVID-19, o modelo proposto alcançou 98,39% de acurácia, 99,18% de sensibilidade e 93,47% de especificidade. A sensibilidade e especificidade alta mostram que o desbalanceamento na base de dados não impactou na qualidade do método proposto uma vez que não houve uma classe dominante. Na Tabela I são encontrados os valores de todos *k-fold* da validação cruzada. O resultado final é a media desses valores, calculada para cada métrica avaliada.

TABELA I
RESULTADO DO DESEMPENHO OBTIDO PELA FCNN NO MÉTODO 10-*fold* DE VALIDAÇÃO CRUZADA. FONTE: AUTOR.

#	Acurácia (%)	Sensibilidade (%)	Especificidade (%)
0	97,60	97,34	100,00
1	97,60	99,12	81,81
2	98,40	100,00	90,00
3	97,60	98,14	94,11
4	100,00	100,00	100,00
5	97,60	98,14	94,11
6	100,00	100,00	100,00
7	98,38	99,08	93,33
8	97,58	100,00	86,95
9	99,19	100,00	94,44
Média	**98,39**	**99,18**	**93,47**

Na Tabela II, temos um comparativo com outros métodos de classificação propostos na literatura. Como pode ser observado, o modelo proposto neste artigo, foi o que alcançou maior acurácia. Além disso, a sensibilidade alcançada, apesar de não ser a maior, mostra que o método é eficiente na detecção de verdadeiros positivos.

V. CONCLUSÃO

A detecção eficiente de pacientes com COVID-19 é vital para evitar a propagação da doença a outras pessoas e salvar vidas. Neste estudo, propusemos uma abordagem baseada em

TABELA II
COMPARAÇÃO DOS RESULTADOS ENTRE ESTE E OUTROS TRABALHOS. AS ABREVIAÇÕES SÃO: ACURÁCIA (AC), SENSIBILIDADE (SE), ESPECIFICIDADE (ES). FONTE: AUTOR.

Autor	Classificador	Ac (%)	Se (%)	Es (%)
Abbas et al.	ResNet	95.12	97.91	91.87
Afshar et al.	CapsNet	95,7	90	95,8
Hemdan et al.	VGG19	90	-	-
Hemdan et al.	DenseNet201	90	-	-
Hemdan et al.	ResNetV2	70	-	-
Hemdan et al.	InceptionV3	50	-	-
Hemdan et al.	InceptionResNetV2	80	-	-
Hemdan et al.	Xception	80	-	-
Hemdan et al.	MobileNetV2	60	-	-
Narin et al.	InceptionV3	97	100	-
Narin et al.	ResNet50	98	100	-
Narin et al.	InceptionResNetV2	87	90	-
Wang et al.	COVID-Net	93	91	-
Método Proposto	**FCNN**	**98,39**	**99,18**	**93,47**

redes neurais totalmente conectadas e recursos de textura, usando imagens de Raio-X do tórax, obtidas de pacientes com COVID-19 e de pacientes saudáveis, para predizer automaticamente pacientes com COVID-19. Os resultados de desempenho mostram que o modelo de Rede Neural Totalmente Conectada produziu um método de alta eficácia. À luz de nossas descobertas, acredita-se que isto ajudará os médicos a tomar decisões na prática clínica devido ao alto desempenho e ajudar a salvar cada vez mais vidas.

Em trabalhos futuros, este método poderá ser utilizado para diferenciar pacientes com pneumonia viral, de pacientes infectados com COVID-19.

REFERENCES

[1] Asmaa Abbas, Mohammed M Abdelsamea, and Mohamed Medhat Gaber. Classification of covid-19 in chest x-ray images using detrac deep convolutional neural network. *arXiv preprint arXiv:2003.13815*, 2020.

[2] Parnian Afshar, Shahin Heidarian, Farnoosh Naderkhani, Anastasia Oikonomou, Konstantinos N Plataniotis, and Arash Mohammadi. Covid-caps: A capsule network-based framework for identification of covid-19 cases from x-ray images. *arXiv preprint arXiv:2004.02696*, 2020.

[3] Ministério da Saúde. Quanto tempo o vírus sobrevive nas superfícies. Disponível em: https://coronavirus.saude.gov.br/index.php/perguntas-e-respostas. Acessado em: 30 jun. 2020.

[4] Organização Mundial das Nações Unidas (ONU). Organização mundial da saúde declara novo coronavírus uma pandemia. Disponível em: https://news.un.org/pt/story/2020/03/1706881. Acessado em: 25 jun. 2020.

[5] M. de Freitas Oliveira Baffa and L. Grassano Lattari. Convolutional neural networks for static and dynamic breast infrared imaging classification. In *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 174–181, 2018.

[6] Site de notícias G1. países da Ásia, oceania, américa do norte, europa, oriente médio, américa do sul e África são afetados pela covid-19. Disponível em: https://g1.globo.com/ciencia-e-saude/noticia/2020/01/23/numero-de-paises-com-casos-confirmados-de-coronavirus.ghtml. Acessado em: 29 jun. 2020.

[7] Folha de São Paulo. Grupo de risco do novo coronavírus. Disponível em: https://www1.folha.uol.com.br/equilibrioesaude/2020/05/homens-e-idosos-sao-quem-mais-morre-de-covid-19-no-estado-de-sp.shtml. Acessado em: 29 jun. 2020.

[8] Nicholas A Hamilton, Radosav S Pantelic, Kelly Hanson, and Rohan D Teasdale. Fast automated cell phenotype image classification. *BMC bioinformatics*, 8(1):110, 2007.

[9] Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, pages 610–621, 1973.

[10] Ezz El-Din Hemdan, Marwa A Shouman, and Mohamed Esmail Karar. Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images. *arXiv preprint arXiv:2003.11055*, 2020.

[11] Sara Hosseinzadeh Kassani, Peyman Hosseinzadeh Kassasni, Michal J Wesolowski, Kevin A Schneider, and Ralph Deters. Automatic detection of coronavirus disease (covid-19) in x-ray and ct images: A machine learning-based approach. *arXiv preprint arXiv:2004.10641*, 2020.

[12] Ali Narin, Ceren Kaya, and Ziynet Pamuk. Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks. *arXiv preprint arXiv:2003.10849*, 2020.

[13] Caio Vinicius de Oliveira. Coronavírus: uso de tomografia computadorizada na detecção. Disponível em: https://brasilrad.com.br/artigos/coronavirus-uso-de-tomografia-computadorizada-na-deteccao/. Acessado em: 30 jun. 2020.

[14] Omir Antunes Paiva and Luciano M Prevedello. O potencial impacto da inteligência artificial na radiologia. *Radiologia Brasileira*, 50(5):V–VI, 2017.

[15] Tawsifur Rahman, Dr. Muhammad Chowdhury, and Amith Khandakar. Covid-19 radiography database. Disponível em: https://www.kaggle.com/tawsifurrahman/covid19-radiography-database. Acessado em: 25 jun. 2020.

[16] B. Ramsundar and R.B. Zadeh. *TensorFlow for Deep Learning: From Linear Regression to Reinforcement Learning*, chapter 4. O'Reilly Media, 2018.

[17] sanarMed. Diagnóstico do coronavírus. Disponível em: https://www.sanarmed.com/coronavirus-origem-sinais-sintomas-achados-tratamentos. Acessado em: 2 julho. 2020.

[18] saude do viajante. Covid-19 associado a sars-cov-2 - mundial. Disponível em: http://www.saudedoviajante.pr.gov.br/2020/04/94/COVID-19-associado-a-SARS-CoV-2-Mundial.html. Acessado em: 25 jun. 2020.

[19] Fengxiang Song, Nannan Shi, Fei Shan, Zhiyong Zhang, Jie Shen, Hongzhou Lu, Yun Ling, Yebin Jiang, and Yuxin Shi. Emerging 2019 novel coronavirus (2019-ncov) pneumonia. *Radiology*, 295(1):210–217, 2020.

[20] Linda Wang and Alexander Wong. Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images. *arXiv preprint arXiv:2003.09871*, 2020.

[21] Tzu-Tsung Wong. Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation. *Pattern Recognition*, 48(9):2839–2846, 2015.

[22] Rikiya Yamashita, Mizuho Nishio, Richard Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, 9, 06 2018.

Optimizing data augmentation policies for convolutional neural networks based on classification of sickle cells

Matheus Vieira da Silva, Larissa Ferreira Rodrigues, João Fernando Mari
Instituto de Ciências Exatas e Tecnológicas
Universidade Federal de Viçosa - UFV
Caixa Postal 22 - 38.810-000 - Rio Paranaíba - MG - Brasil
Email: {matheus.v.silva, larissa.f.rodrigues, joaof.mari}@ufv.br

***Abstract*—Data augmentation is a key procedure in many image classification tasks, mainly to overcome the problem of small datasets. In this work, we exploit the data augmentation as a hyperparameter optimization approach. We tested our methods to classify erythrocytes to assist the diagnosis of sickle cell anemia. In this study, we proposed a data augmentation approach based on hyperparameter optimization to find the best augmentation policies through the Bayesian optimization algorithm. We also developed a convolutional neural network architecture from scratch and compared it with two classic architectures to classify sickle cell images. Our approach defines the best data augmentation solutions and sends those solutions to be carried out by CNN in the final training. All experiments were validated using a stratified five-fold cross-validation procedure, and our best result achieves 92.54% of accuracy. The results suggest the best augmentation policies defined with optimization allow us to obtain superior results than other strategies such as without data augmentation, several randomly defined image transformations, and only random rotations. As far as we know, our paper is the first to propose optimizing data augmentation policies in biomedical images leading to a better exploration of these strategies in several fields.**

***Keywords*—sickle cell; medical imaging; deep learning; data augmentation; Bayesian optimization.**

I. Introduction

Sickle cell anemia is an inherited disease caused by a single genetic mutation in the hemoglobin resulting in the abnormal hemoglobin S (HbS). Thus, the erythrocyte (red blood cell) assumes an irregular shape becoming a sickle cell. This format reduces the oxygen and blocking the blood vessels, which may cause stroke and other chronic complications. It is estimated that approximately 300,000 children are born with sickle cell anemia each year, making the disease a global health problem [1].

Visual analysis of blood smear is a procedure that can be used to identify sickle cell disease. However, this task is subjective, very time-consuming, and challenging in emerging countries, especially where the incidence of this disease is high [1] [2] [3].

Computer-aided diagnosis has been used for decades to identify patterns in medical images. The improvements in the hardware of computers are allowing us to train even more complex models to identify, classify and quantify anomalies in biomedical images [4] [5].

Recent advances in deep learning techniques, in particular Convolutional Neural Networks (CNNs), demonstrate that this approach learns complex structures from the data itself, without requiring handcrafted feature extraction [6]. Solutions based on CNNs have a low cost and can help healthcare workers diagnose several diseases, such as sickle cell disease. Also, the data and results may be shared and processed around the world [7].

Several studies have been proposed for classifying sickle cell using manually feature extraction [8] [9] [10] [11]. However, solutions based on CNNs are advantageous and allowing automatized the steps of feature extraction and require minimal preprocessing [12].

Although CNN appears to be a promising approach, data augmentation strategies may be necessary to deal with small datasets and overfitting issues. In this way, the choice of best data augmentation strategies is crucial to the classification performance. Usually, this choice is carried manually by testing different strategies based on random transformations until the model gives a "satisfactory" performance. Since the optimal data augmentation strategies are unknown, any definition of a level of satisfaction using this methodology is subjective and time-consuming [13].

In this work, we aim to automate the process of finding an effective data augmentation policy for cell classification tasks. We define each policy as a possible choice of augmentation (e.g., rotations, flips, color adjustments) and the magnitudes for each transformation. Furthermore, the contribution of this paper is an assessment of the data augmentation policies as an optimization problem, where the policies are considered decision variables and the accuracy of the trained model is the objective function to be maximized. We applied a Bayesian search algorithm [14] to identify data augmentation strategies according to the data. These operations are evaluated using three different CNNs architectures: AlexNet [15], LeNet-5 [16], and our custom architecture named Model A. As far as we know, our work is the first to introduce the optimization of data augmentation policies for biomedical image classification. Our results suggest that optimization improving the performance of all CNNs evaluated.

The remaining of this paper is organized as follows: Section II presents the related work. Section III describes the material and methods. Section IV presents and discusses the results.

Section V presents conclusions and future work.

II. Related Work

The state-of-the-art presents multiple systems dedicated to erythrocyte classification, involving machine learning with handcrafted feature extraction to differentiate among the different types of cells [17] [10] [11].

Recently, approaches using deep learning have been proposed. Xu et al. [18] proposed a 10-layer CNN to classify erythrocytes using 7,000 images categorized into five and eight classes. They considered k-fold cross-validation to validate the experiments and obtained for five and eight categories an average accuracy of 89.28% and 87.50%.

Qiu et al. [19] presented a structure to extract regions from the images containing the cells using a Region-based Convolutional Network (RCNN). They performed a multi-label classification using a pre-trained ResNet-50 architecture and a binary classification using Gradient Boosting Classifier. Finally, the method proposed by [19] obtained an overall accuracy of 72.2%.

Alzubaidi et al. [20] utilized a CNN architecture composed of 18 layers, ReLu as the activation function, and batch normalization. The Error-Correcting Output Codes (ECOC) were used to solve the classification problem using the SVM classifier, achieving 92.06% in terms of accuracy.

Most recently, Alzubaidi et al. [21] applied the same domain transfer learning in conjunction with SVM and data augmentation techniques to minimize the overfitting. They evaluated three datasets (main dataset, training with transfer learning, and testing) and developed three architectures with 40, 35, and 29 layers. The result obtained was of 99.98%, which is the best state-of-the-art score reported in the literature. However, for applications in a real-world scenario, it is impracticable to train a CNN with a dataset from the same domain.

The main difference between previous work to our work is the analysis of the data augmentation impact on the performance of a number of CNN architectures. We selected data augmentation policies automatically through a Bayesian search algorithm. We believe that our approach can contribute to identifying sickle cell disease, overcoming the costs of more data acquisition, and avoiding overfitting. Also, our method allows for finding the best augmentation operations to increase accuracy and is suitable to deal with the issue of the lack of training data for CNN architectures.

III. Material and Methods

As presented before, the main goal of this work is to automatically find the best data augmentation policies for training CNNs using the Bayesian optimization strategy. The proposed method was programmed using Python 3.6, the Keras 2.2.4[1] framework with TensorFlow 1.12.0, CUDA version 9.0 and cuDNN 7.1. The data augmentation optimization was drawn from the *deepaugment*[2] library. Also, we used *Numpy*, *OpenCV*, *Scikit-learn*, and *imgaug*[3] libraries. The workflow of the proposed method is summarized in Fig 1.

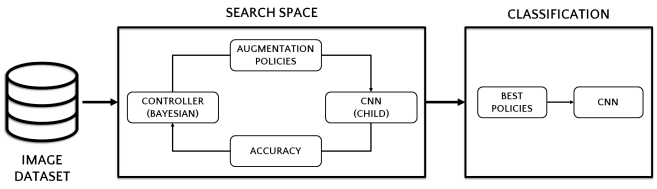


Fig. 1. Steps of the proposed method.

A. Image Dataset

The images used was taken from the *erythrocytesIDB*[4] dataset. It contains 626 images of erythrocytes, each with a single, centered cell in evidence, categorized as healthy (202 images), sickle cell (211 images), and with other deformation (213 images) [22].

When considering the erythrocytes classification based on CNNs, previous studies used images without any preprocessing [20] [21]. However, we performed some preliminary experiments considering the original dataset, and the results demonstrate that training from scratch with original images hurts the results, as the models reached an accuracy of less than 60%. Thus, all images were preprocessed using a simple segmentation procedure proposed by Rodrigues et al. [10]. In this approach, we segment each image using the global Otsu's threshold and morphological operations. Fig. 2 illustrates one image from each class resulting from preprocessing.

Fig. 2. Examples of image instances resulting from preprocessing for each class of *erythrocytesIDB* dataset: (a) healthy; (b) sickle cell; and (c) other deformation.

B. Augmentation policies

The proposal presented in this paper is inspired by [23], which is a framework to search for the best data augmentation policies automatically. These policies are composed of processing functions that will provide a training solution for a child CNN architecture. The term child CNN is the reference to CNN approved in the optimization tests, which uses the accuracy generated in the training step as feedback for the search algorithm.

The augmentation policies were adapted from the *deepaugment* library and are composed of two image processing operations and their respective magnitudes, A and B. The

[1] https://keras.io/
[2] https://pypi.org/project/deepaugment/
[3] https://imgaug.readthedocs.io/en/latest/
[4] Available in: http://erythrocytesidb.uib.es/

magnitude is an optimizable parameter that passed for a discretization process [24], with real values between 0 and 1. In this study, we consider optimizing twenty image processing operations available in the *imgaug* library.

C. CNN Model A

We designed a lightweight convolutional neural network with custom architecture, called Model A. As demonstrated by [25], lightweight models trained from scratch can achieve better results in medical images without the transfer learning technique, especially when the dataset is too small to train a deep network. In this way, when the transfer learning technique is not applied, data augmentation strategies are essential in the training step. Also, our architecture has a lower computational cost when compared to other architectures such as [26] and [27].

Our Model A is composed of six convolutional layers, three pooling layers, and one fully connected layer. Besides, this architecture adopts dropout connections to reduce overfitting and REctified Linear Unit (ReLU) activation to accelerate the training. Fig. 3 illustrates the Model A.

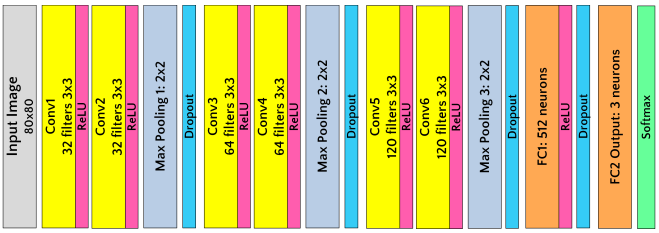


Fig. 3. Model A architecture.

The details of each layer are described below [12].

a) Convolutional layer: The convolutional layers perform the convolution operation in each previous layer to extract relevant features from the images, e.g., color and border. Eq. 1 summarizes the convolution operation.

$$Z_j^l = \sum_{i=1}^{I} W_i^l * A_i^{l-1} + B_j^l \quad (1)$$

Where Z_j^l is the output volume that contains the feature maps, W_i^l is a tensor containing the filters A_i^{l-1}. The Z is the previous layer. Lastly, is added the bias B_j^l and each layer Z has a ReLU activation function.

b) Pooling layer: The pooling layer reduces the size of the feature map. In this study, we apply the maximum-pooling technique. This operation calculates the maximum value of a region of the feature map to improve the generalization and the convergence speed of the model [28]. In our network Model A, we adopted max-pooling of size 2×2, reducing the number of pixels in half.

c) Fully connected layer: The last layer consists of a classic neural network that computes the scalar product between the input vector 1D and the weight vector and adds a bias. The input vector is the result of the 2D feature map converting [16]. Finally, the softmax activation function is applied in the last layer of the network to transform the units into probabilities [15].

d) Dropout: Dropout connections are applied to reduce overfitting In order to reduce overfitting, the dropout method is generally used in literature during training. This connection allows excluding some units and their respective connections avoiding an excessive adaptation of neurons [29].

The training step defines a loss function to calculate the model error and an optimizer for the optimization process that will update the weights using the back-propagation algorithm [30]. In this work, we chose to apply Adam [31] optimizer in conjunction with the Categorical Cross-Entropy (CCE) loss function, defined in Eq. 2.

$$CCE = -\frac{1}{N}\sum_{i=0}^{N}\sum_{j=0}^{J} y_j log(\hat{y}_j) + (1-y_j) log(1-\hat{y}_j) \quad (2)$$

Where $\hat{y}$ is the prediction of the model and the respective label is represented by y. For all three CNN architectures evaluated in this paper, the learning rate was set to 0.001 [32].

D. Classical CNN architectures

We selected two classical architectures for experimental comparison: LeNet-5 and AlexNet, both of which were trained from scratch and initializing the weights randomly.

a) LeNet-5: Was the first case of success of CNNs and was proposed by LeCun et al. [16] for character recognition. It is composed of seven layers: three convolutional layers, two pooling layers with average pooling, and two fully connected layers.

b) AlexNet: Proposed by Krizhevsky et al. [15] and won the ILSVRC 2012 [33]. It is composed of ten layers: five convolutional layers, three pooling layers, and two fully connected layers.

E. Controller

In this study, the controller uses the Bayesian algorithm instead of Reinforcement Learning applied by [23]. The Bayesian approach finds the best possible parameter setup faster than other strategies and presents a lower computational cost [34]. Initially is carried a priori sampling of the augmentation policies. These policies should maximize an objective function O to found the best parameters at each interaction based on a surrogate model built from the objective function O. Updating the magnitudes is performed according to the accuracy feedback obtained from the child CNN.

F. Configuration and Training

All images were randomly partitioned into a training set and testing set with the proportions 80% and 20%, respectively. After, we considered 20% of the training set as a validation set to search for the best augmentation policies. In the two stages, we used the same CNN: search space (with child CNN); and classification.

a) Search space: To find an optimized solution, we defined 300 interactions. In each iteration, the child CNNs are trained with three different samples with a N number of epochs and batch size of 64. The controller uses as feedback the accuracy provided by CNN to search the best possible augmentation policies from the dataset. Accuracy is a metric derived from the confusion matrix used to measure the performance of a model [35], as defined in Eq. 3.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

where TP is the True Positives, TN True Negatives, FP False Positives and FN False Negatives.

b) Classification: This step performs the classification of the cells applying the data augmentation policies according to the optimization defined in the previous step. The training set is used according to the first partitioning, i.e., 80% of the dataset. We adopted a k-fold cross-validation strategy [36], and the dataset was randomly partitioned into five stratified folds which are iteratively selected as training and test sets. Finally, we measured the averaged accuracy of five sets to produce a single estimation.

IV. Results and Discussion

The data augmentation optimization was performed on a machine with a GPU NVIDIA Titan XP with CUDA version 9.0, an Intel processor i5 3.00GHz, and 32 GB RAM. In order to accelerate the experiments, the classification steps were carried out in parallel in a machine with a GPU NVIDIA GeForce GTX 1080 TI under CUDA version 9.0, an Intel processor i5 3.00GHz, and 32 GB RAM. The random processes are locked using a fixed seed value to ensure the reproducibility of the experiments.

All images were resized to 80 $\times$ 80 pixels for Model A; 32 $\times$ 32 pixels for LeNet-5; and 224 $\times$ 224 pixels for AlexNet. Then, we performed the data augmentation policies using Bayesian optimization for each CNN evaluated.

We represent the choice of data augmentation operations as a hyperparameter optimization problem, where we extracted the optimized solution from the dataset for each CNN architecture. Table I presents the augment policies and their respective magnitudes for each CNN found through the Bayesian search optimization approach. It is important to note that for twenty operations available, only ten operations were selected since each policy is composed by two image processing operations and its respective magnitudes, there are five data augmentation policies in this study. During training, each image has 50% chance to be augmented by a policy and one of the five policies is selected randomly to augment the image.

The high computational cost is one of the biggest challenges when working with the optimization approach explored in this study because as the number of CNN parameters increases, the time required for training also increases significantly. However, our approach allows for better results in fewer evaluations than a grid search or random search [37]. The training of LeNet-5 was carried out in two hours and seven minutes, defining in 50 epochs of child CNN. The Model A, it took two hours with six epochs of child CNN. AlexNet, the most complex CNN, needed seven hours and fifty-four minutes for total training, with the number of epochs set at six. We defined the number of child CNN epochs empirically, and 100 epochs for the classification step.

Our results demonstrate that optimized data augmentation operations reduce the overfitting and do not require other regularization techniques. In order to assess the values of loss and accuracy during training and validation, the average values are shown graphically in Fig 4. The charts show each CNN's behavior in the training step, in which training losses and validation losses decrease with each iteration. This behavior suggests that the training did not overfit the data, thus evidencing the importance of cross-validation in our experiments allowing results less subject to randomness.

After defining the best data augmentation policies using Bayesian optimization, we test the trained models generated by each fold. Table II shows the test result obtained for each fold and the average result. These results demonstrate that our Model custom design achieved an average accuracy of 92.54%, and in some folds, obtained an accuracy above 96%, which is very close to the current state of the art [21] that did not consider k-fold cross-validation. Finally, the networks AlexNet and LeNet-5 obtained an average accuracy of 90.00% and 87.93%, respectively.

In addition, we compared our proposed approach with other three strategies: i): without data augmentation; ii) different operations (such as flips, rotations, zoom, whitening transformation, and others.); and iii) random rotations with angles steps of 10°.

To evaluate the impact of each strategy on the accuracy of the CNNs, they were all trained from scratch. To evaluate the impact of these strategies on the accuracy of the CNNs, they were all trained from scratch. As shown in Table III, optimized data augmentation improved the accuracy of all CNN models. Our results demonstrate that only applying several data augmentation strategies without considering Bayesian optimization reduces classification performance. In particular, the accuracy of Model A (with different operations) decreased by 49.21 percentage points compared with optimized data augmentation. It is worth noticing that optimized data augmentation is the best strategy, but in scenarios where only a short time is available for training, only random rotations should be considered as a data augmentation strategy.

V. Conclusion

This paper presents and evaluates an approach to search for the best data augmentation policies using Bayesian optimization. As far we know, our method is the first to introduce optimizing data augmentation policies for biomedical image classification, and the experimental results were very close to the state-of-the-art, reaching an accuracy of 92.54%.

We compared the data augmentation optimized using the Bayesian method with three other training strategies (without data augmentation, different data augmentation operations, and random rotations). Our results point out that Bayesian optimization overcomes the traditional empirically defined data augmentation methods. We also proposed a lightweight CNN architecture we called Model A. Our experimental results

TABLE I. THE BEST DATA AUGMENTATION POLICIES FOUND BY THE BAYESIAN OPTIMIZATION.

	Model A		AlexNet		LeNet-5	
	Augmentation Strategy	**Magnitude**	**Augmentation Strategy**	**Magnitude**	**Augmentation Strategy**	**Magnitude**
Policy A	brighten	0.18	invert	0.352	clouds	0.522
	gamma-contrast	0.665	brighten	0.566	vertical-flip	0.949
Policy B	clouds	0.18	dropout	0.15	gaussian-blur	0.785
	brighten	0.708	translate-y	0.054	rotate	0.928
Policy C	shear	0.027	vertical-flip	0.555	translate-y	0.146
	rotate	0.503	dropout	0.092	dropout	0.379
Policy D	clouds	0.266	emboss	0.857	clouds	0.814
	invert	0.891	rotate	0.626	add-to-hue-and-saturation	0.035
Policy E	brighten	0.726	additive-gaussian-noise	0.047	horizontal-flip	0.389
	gamma-contrast	0.611	dropout	0.571	vertical-flip	0.947

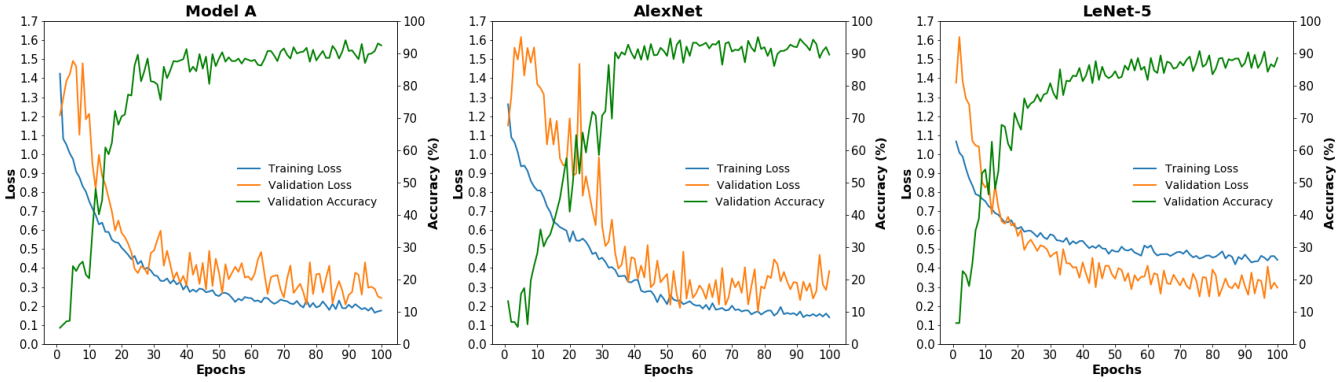


Fig. 4. Charts showing the average evolution of accuracy and loss values for the training and validation set.

TABLE II. 5-FOLD TEST AND AVERAGE ACCURACY FOR EACH CNN MODEL, USING THE BEST DATA AUGMENTATION POLICIES.

Fold	**Model A (%)**	**AlexNet (%)**	**LeNet-5 (%)**
1	94.44	91.27	86.51
2	96.03	90.48	88.89
3	95.24	88.89	89.68
4	91.27	89.68	86.51
5	85.71	89.68	88.10
Average	92.54	90.00	87.93

TABLE III. AVERAGE ACCURACY FOR EACH CNN EVALUATED CONSIDERING DIFFERENT DATA AUGMENTATION STRATEGIES.

	Data Augmentation Strategy			
CNN	**Optimized**	**Without**	**Different Operations**	**Random Rotations**
Model A	**92.54%**	89.68%	43.33%	90.48%
AlexNet	90.00%	87.14%	46.98%	88.10%
LeNet-5	87.93%	87.62%	63.33%	89.68%

demonstrate that combining optimized data augmentation policies and the custom-designed CNN architecture has significantly improved the performance of the sickle cell disease classification. Moreover, the Bayesian search speeds the training process when compared to grid and random search, while it reduces the number of trials.

As future work, we plan to test our proposed method for other biomedical images in order to verify that our findings hold for similar datasets. Moreover, we intend to perform benchmarking with the training technique based on transfer learning and evaluating further optimization algorithms.

ACKNOWLEDGMENTS

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the TITAN Xp GPU used for this research. This study was financed by Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (M. V. da Silva received a scholarship from PIBIC/CNPq). We would like to thanks CAPES and FAPEMIG (Grant number CEX - APQ-02964-17) for the financial support. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

REFERENCES

[1] Gregory J Kato, Frédéric B Piel, Clarice D Reid, Marilyn H Gaston, Kwaku Ohene-Frempong, Lakshmanan Krishnamurti, Wally R Smith, Julie A Panepinto, David J Weatherall, Fernando F Costa, et al. Sickle cell disease. *Nature Reviews Disease Primers*, 4(1):1–22, 2018.

[2] Frédéric B Piel, Martin H Steinberg, and David C Rees. Sickle cell disease. *New England Journal of Medicine*, 376(16):1561–1573, 2017.

[3] Gentil Claudino de Galiza Neto and Maria da Silva Pitombeira. Aspectos moleculares da anemia falciforme. *Jornal Brasileiro de Patologia e Medicina Laboratorial*, 39(1):51–56, 2003.

[4] James S Duncan and Nicholas Ayache. Medical image analysis: Progress over two decades and the challenges ahead. *IEEE transactions on pattern analysis and machine intelligence*, 22(1):85–106, 2000.

[5] Marleen De Bruijne. Machine learning approaches in medical image analysis: From detection to diagnosis, 2016.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[7] Livia Faes, Xiaoxuan Liu, Aditya Kale, Alice Bruynseels, Mohith Shamdas, Gabriella Moraes, Dun Jack Fu, Siegfried K Wagner, Christoph Kern, Joseph RE Ledsam, et al. Deep learning under scrutiny: performance against health care professionals in detecting diseases from medical imaging-systematic review and meta-analysis. 2019.

[8] Athira Sreekumar and Ashok Bhattacharya. Identification of sickle cells from microscopic blood smear image using image processing. *International Journal of Emerging Trends in Science and Technology*, 1(5):783–787, 2014.

[9] Shashi Bala and Amit Doegar. Automatic detection of sickle cell in red blood cell using watershed segmentation. *Int. J. Adv. Res. Comput. and Commun. Eng*, 4(6):488–491, 2015.

[10] Larissa Ferreira Rodrigues, Murilo Coelho Naldi, and João Fernando Mari. Morphological analysis and classification of erythrocytes in microscopy images. In *XII Workshop de Visão Computacional*, Campo Grande, MS, Brazil, 2016. WVC.

[11] Lucas Costa de Faria, Larissa Ferreira Rodrigues, and João Fernando Mari. Cell classification using handcrafted features and bag of visual words. In *2018 Workshop de Visão Computacional (WVC)*, pages 68–73, Nov 2018.

[12] M. A. Ponti, L. S. F. Ribeiro, T. S. Nazare, T. Bui, and J. Collomosse. Everything you wanted to know about deep learning for computer vision but were afraid to ask. In *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, pages 17–41, Oct 2017.

[13] Fábio Perez, Cristina Vasconcelos, Sandra Avila, and Eduardo Valle. Data augmentation for skin lesion analysis. In *OR 2.0 Context-Aware Operating Theaters, Computer Assisted Robotic Endoscopy, Clinical Image-Based Procedures, and Skin Image Analysis*, pages 303–311. Springer, 2018.

[14] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[16] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[17] Manuel Gonzalez-Hidalgo, FA Guerrero-Pena, S Herold-Garcia, Antoni Jaume-i Capó, and Pedro D Marrero-Fernández. Red blood cell cluster separation from digital images for use in sickle cell disease. *IEEE journal of biomedical and health informatics*, 19(4):1514–1525, 2014.

[18] Mengjia Xu, Dimitrios P Papageorgiou, Sabia Z Abidi, Ming Dao, Hong Zhao, and George Em Karniadakis. A deep convolutional neural network for classification of red blood cells in sickle cell anemia. *PLoS computational biology*, 13(10), 2017.

[19] Wei Qiu, Jiaming Guo, Xiang Li, Mengjia Xu, Mo Zhang, Ning Guo, and Quanzheng Li. Multi-label detection and classification of red blood cells in microscopic images. *arXiv preprint arXiv:1910.02672*, 2019.

[20] Laith Alzubaidi, Omran Al-Shamma, Mohammed A. Fadhel, Laith Farhan, and Jinglan Zhang. Classification of red blood cells in sickle cell anemia using deep convolutional neural network. In Ajith Abraham, Aswani Kumar Cherukuri, Patricia Melin, and Niketa Gandhi, editors, *Intelligent Systems Design and Applications*, pages 550–559, Cham, 2020. Springer International Publishing.

[21] Laith Alzubaidi, Mohammed A. Fadhel, Omran Al-Shamma, Jinglan Zhang, and Ye Duan. Deep learning models for classification of red blood cells in microscopy images to aid in sickle cell anemia diagnosis. *Electronics*, 9(3):427, Mar 2020.

[22] Manuel Gonzalez-Hidalgo, FA Guerrero-Pena, S Herold-Garcia, Antoni Jaume-i Capó, and Pedro D Marrero-Fernández. Red blood cell cluster separation from digital images for use in sickle cell disease. *IEEE journal of biomedical and health informatics*, 19(4):1514–1525, 2015.

[23] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 113–123, 2019.

[24] Huan Liu, Farhad Hussain, Chew Lim Tan, and Manoranjan Dash. Discretization: An enabling technique. *Data mining and knowledge discovery*, 6(4):393–423, 2002.

[25] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. In *Advances in Neural Information Processing Systems*, pages 3342–3352, 2019.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[27] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[28] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *International conference on artificial neural networks*, pages 92–101. Springer, 2010.

[29] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[30] Yann LeCun, D Touresky, G Hinton, and T Sejnowski. A theoretical framework for back-propagation. In *Proceedings of the 1988 connectionist models summer school*, volume 1, pages 21–28. CMU, Pittsburgh, Pa: Morgan Kaufmann, 1988.

[31] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[32] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer, 2012.

[33] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[34] Ian Dewancker, Michael McCourt, and Scott Clark. Bayesian optimization for machine learning: A practical guidebook. *arXiv preprint arXiv:1612.04858*, 2016.

[35] Stephen V Stehman. Selecting and interpreting measures of thematic classification accuracy. *Remote sensing of Environment*, 62(1):77–89, 1997.

[36] Michael W Browne. Cross-validation methods. *Journal of mathematical psychology*, 44(1):108–132, 2000.

[37] James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2546–2554. Curran Associates, Inc., 2011.

Evaluating Convolutional Neural Networks for COVID-19 classification in chest X-ray images

Leonardo Gabriel Ferreira Rodrigues, Larissa Ferreira Rodrigues,
Danilo Ferreira da Silva, João Fernando Mari
Instituto de Ciências Exatas e Tecnológicas
Universidade Federal de Viçosa - UFV
Caixa Postal 22 - 38.810-000 - Rio Paranaíba - MG - Brasil
Email: {leonardo.g.rodrigues, larissa.f.rodrigues, danilo.f.silva, joaof.mari}@ufv.br

***Abstract*—Coronavirus Disease 2019 (COVID-19) pandemic rapidly spread globally, impacting the lives of billions of people. The effective screening of infected patients is a critical step to struggle with COVID-19, and treating the patients avoiding this quickly disease spread. The need for automated and scalable methods has increased due to the unavailability of accurate automated toolkits. Recent researches using chest X-ray images suggest they include relevant information about the COVID-19 virus. Hence, applying machine learning techniques combined with radiological imaging promises to identify this disease accurately. It is straightforward to collect these images once it is spreadly shared and analyzed in the world. This paper presents a method for automatic COVID-19 detection using chest X-ray images through four convolutional neural networks, namely: AlexNet, VGG-11, SqueezeNet, and DenseNet-121. This method had been providing accurate diagnostics for positive or negative COVID-19 classification. We validate our experiments using a ten-fold cross-validation procedure over the training and test sets. Our findings include the shallow fine-tuning and data augmentation strategies that can assist in dealing with the low number of positive COVID-19 images publicly available. The accuracy for all CNNs is higher than 97.00%, and the SqueezeNet model achieved the best result with 99.20%.**

***Keywords*—COVID-19; coronavirus; chest X-ray; convolutional neural networks; data augmentation; fine-tuning.**

I. Introduction

Coronavirus Disease 2019 (COVID-19) caused by a novel coronavirus, officially named Severe Acute Respiratory Syndrome Coronavirus 2 (SARS-CoV-2) [1], is a pandemic that first emerged in the Chinese city of Wuhan, and rapidly spread to other countries, affecting Italy, Iran, Spain, Brazil, Russia, India, and United States severely. All countries affected by COVID-19 imposed a nationwide lockdown in an attempt to slow the spread of the virus, causing a profound overall impact on the lives of billions of people from a health, safety, and economic perspective [2] [3]. The COVID-19 can cause illness to the respiratory system leading to inflammation of the lungs and pneumonia [4]. There is no known specific therapeutic drugs or vaccine for COVID-19, and the impact in the healthcare system is also high due to the number of people that needs intensive care unit (ICU) admission and breathing machine for long periods [5].

The most common test technique currently used for COVID-19 diagnosis is the reverse transcription-polymerase chain reaction (RT-PCR). However, considering the difficulties of distributing the kits and collecting the samples and the waiting time for results, auxiliary diagnostics methods are welcome to assist the medical team decision making. In this context, the development of computer-aided diagnosis systems based on machine learning is essential and widely applied in several fields of medicine [6] [7] [8].

Early studies demonstrated that many patients infected with COVID-19 present abnormalities in chest X-ray images [9] [10] [11]. These images can be easily collected, shared, and analyzed around the world. Moreover, the task of COVID-19 identification is not easy, and the specialist reviewing the chest X-ray needs to look for white patches in the lungs, i.e., air sacs filled with pus or water. However, these white patches can also be confused with diseases such as tuberculosis, bronchitis, and other types of pneumonia caused by different viruses or bacteria.

In this study, we aim to explore the identification of COVID-19 using chest X-ray images due to its reduced cost, fast result, and general availability. Our principal goal is to reach the best possible identification rate among COVID-19 and other types of pneumonia. We applied a pure deep learning approach comparing four Convolutional Neural Networks (CNNs): AlexNet, VGG-11, SqueezeNet and DenseNet-121, and we evaluated the performance using a k-fold cross-validation procedure over the training and test sets. Moreover, we carried a confusion matrix analysis to measure accuracy, precision, recall, and F1-score indices.

Furthermore, the chest X-ray image dataset used in this work contains a few positive images on the COVID-19. To deal with this, we applied Shallow Fine-Tuning (SFT) training and data augmentation based on random rotation and shifting to balance the class distribution and the performance of the classification. In addition, our approach is fast and simple producing high performing system. As far as we know, our result is the best obtained for COVID-19 identification in chest X-ray images. We believe that our proposed method can contribute to future researches intended to help healthcare workers to identify COVID-19 and to manage patient's conditions.

The remaining of this paper is organized as follows: Section II surveys related work; Section III describes the material and methods; In Section IV we present and discuss the results obtained. Finally, conclusions and future work are presented in Section V.

II. Related Work

The COVID-19 has been attracting much attention from the image analysis research community due to its severity. In this sense, Narin et al. [12] compared three different CNN architectures (ResNet50, Inception-V3, and InceptionResNetV2) to identify COVID-19 in chest X-ray images. They used a dataset composed of fifty COVID-19 images taken from the open-source GitHub repository shared by Dr. Joseph Cohen [13] and fifty healthy lung images from Kaggle repository "Chest X-Ray Images (Pneumonia)" [14]. The ResNet-50 obtained the best result achieving an accuracy of 98%.

Hemdan et al. [15] proposed a COVIDX-Net composed of seven popular CNN models and used the same dataset considered by [12] and achieved 90% in terms of accuracy. However, only 25 samples of COVID-19 positive and 25 samples of negative images were considered.

Sethy and Behera [16] also considered the same dataset of [15]. Their study states that the ResNet-50 as a feature extractor and Support Vector Machine (SVM) classifier provided the best performance obtained an accuracy of 95.38%.

Wang and Wong [17] proposed a COVID-Net architecture, an open-source CNN created to detect COVID-19 on chest X-ray images. The authors used a dataset created exclusively to support COVID-Net experimentation, which obtained 93.3% accuracy in classifying normal, non-COVID pneumonia, and COVID-19 classes.

Apostolopoulos and Mpesiana [18] adopted different pre-trained network architectures to address the task of classification of COVID-19 in chest X-ray images and achieved 96.78% of accuracy with MobileNet v2 model.

Khan et al. [19] proposed the CoroNet deep CNN with 71 layers, inspired by Xception (Extreme Inception) and trained on the ImageNet dataset [20]. According to the authors, the CoroNet was evaluated using a dataset not publicly available for download and achieved an average accuracy of 89.60% for the COVID-19 identification.

Ozturk et al. [21] proposed an approach for early detection of COVID-19 cases using the DarkNet model as a classifier YOLO object detection system and obtained an accuracy of 98.08% for binary classes and 87.02% for multi-class cases. Ucar and Korkmaz [22] proposed a method based on SqueezeNet [23] architecture with Bayes optimization and achieved 98.30% of accuracy.

Pereira et al. [24] utilized texture descriptors, fusion techniques and a pre-trained Inception-V3 model to identify COVID-19 obtained 89.00% in terms of F1-score. However, the validation methodology used in [22] and [24] is a simple hold-out technique that has a certain probability of building biased sets, which may achieve abnormal accuracy results, mainly in small datasets.

The automated classification of COVID-19 in X-ray images is a hot topic nowadays due to the growing pandemic, and new works are emerging every day. In contrast to the previous works, we explore training based on SFT, data augmentation strategy, and our approach is a promising alternative by delivering a simple and efficient that allows achieved better results with a low computational cost.

III. Material and Methods

The main goal of this paper is to evaluate the performance of different architectures of CNNs to classify COVID-19 in chest X-ray images. More precisely, we find the best possible identification rate among COVID-19 and other types of pneumonia. Fig. 1 illustrates the steps of the methodology adopted here.

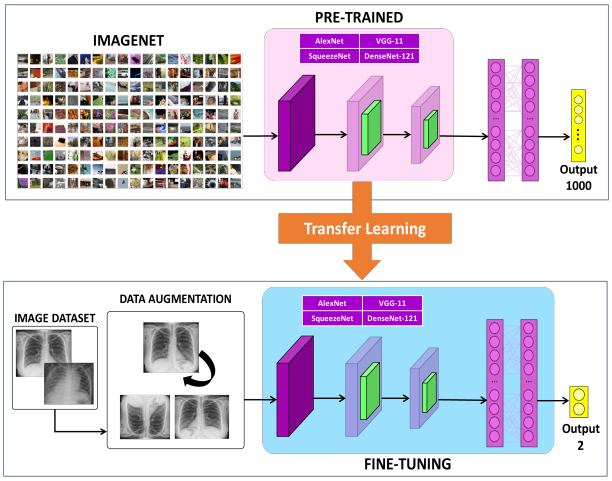


Fig. 1. Steps of proposed method.

A. Image dataset

The images used in this work were obtained from two datasets of chest X-ray images. The first dataset contains 108 images of COVID-19 positive and was taken from the GitHub repository shared by Dr. Joseph Cohen, at the University of Montreal [13] (last accessed April 10, 2020). We selected 299 images of COVID-19 negative, corresponding to 20% of viral pneumonia images selected randomly from the Chest X-Ray Images (Pneumonia) dataset available in Kaggle repository [14]. Note that only about 20% of the viral pneumonia was selected in order to avoid imbalance between the classes or bias the classification performance.

The information about images is summarized in Table I. To illustrate the dataset resulting from the combination of the two datasets previously mentioned, sample images from each class are presented in Fig. 2.

TABLE I. Distribution of the chest X-ray images.

COVID-19	**Samples**	**Source**
Positive	108	GitHub (Dr. Joseph Cohen) [13]
Negative	299	Kaggle (X-ray images of Pneumonia) [14]
Total	**407**	

B. Pre-processing

All images were resized to 224 × 224 pixels based on bilinear interpolation. The resize allows adapting each image for the input of the CNN architectures used in this work.

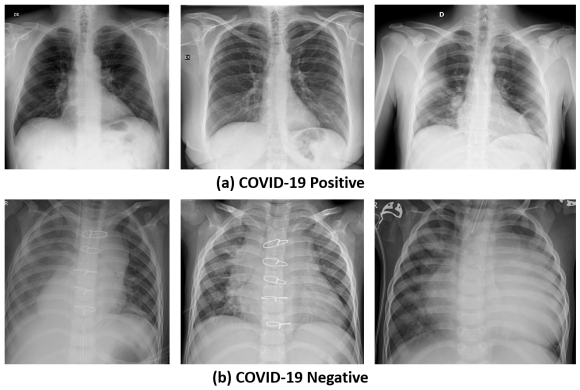


Fig. 2. Examples of image instance for each class.

As one of the main obstacles in this study is the lack of images, we applied data augmentation strategy to increase the training data artificially without introducing labeling costs [25]. All training images were augmented by using vertical and horizontal flips, and rotating each of the original images around its center through randomly chosen angles of between -10° and 10°.

C. Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a multi-stage image classification technique that incorporates spatial context and weight sharing between pixels in order to extract high-level hierarchical representations of the data [26] [27]. Thus, CNN is able to extract features during training. In this work, four CNNs architectures are tested: AlexNet [25], VGG-11 [28], SqueezeNet [23], and DenseNet-121 [29]. All CNNs were selected based on their success in previous image classification tasks.

The AlexNet was the champion of ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 [20]. This CNN consists of five convolutional layers, three max-pooling layers, two fully connected layers with a final softmax layer. In order to reduce overfitting, AlexNet uses dropout connections and REctified Linear Unit (ReLU) activation function [25].

The VGG won the identification and classification tasks in the ILSVRC 2014. In order to reduce the computational cost, this CNN adopted sequences of convolutional filters of size 3×3 [28]. In this paper, we use the VGG-11 architecture with batch normalization, due to its simplicity and robustness. It is important to note that batch normalization is very effective to overcome the challenges of deep training [30].

SqueezeNet is a compact CNN with approximately 50 times fewer parameters than AlexNet model. It is composed of a stand-alone convolution layer followed by eight fire modules and a final convolution layer [23]. Each fire module contains only a filter of size 1×1 inputting into an expanded layer composed by convolutional filters of size 1×1 and 3×3. In this way, the modules are able to perform the same functions of fully connected and dense layers.

DenseNet-121 architecture uses dense blocks to concatenate a number of convolutional layers reducing the number o features through average pooling. In the present study, we use the DenseNet with 121 layers: one initial convolutional layer followed by max-pooling, 116 convolutional layers followed by batch normalization, and ReLU functions, interpolated with three transition blocks, and a last average pooling before the start of fully connected layers [29].

D. Shallow Fine-Tuning

The Shallow Fine-Tuning (SFT) [31] [32] is a training strategy based on the concept of transfer learning and is suitable for small data sets. This approach is used to train deep learning models in which the network is pre-trained for a classification task using a huge dataset such as ImageNet [20].

The weights in all convolutional layers are initialized with the corresponding values from the pre-trained model. These layers are considered more general and retain information about texture, color, and shape. SFT performs fine-tuning only in the last fully connected layer, which is more specialized. Usually, this strategy is the most common allowing weights of the last layers to adapt to the classification problem.

E. Training strategy

The training of the CNNs models is defined as an optimization problem in order to optimize the quality of the prediction. In this work, we considered the objective function as the cross-entropy defined by $\mathcal{L}(W)$. Equation 1 show that $\mathcal{L}(W)$ is computed over a set of training samples X_j considering the tuned weights W, parameters $f(x_j)$, and the known classes y_j, where j represents the classes COVID-19 positive and negative.

$$\mathcal{L}(W) = \frac{1}{n}\sum_{j=1}^{N} \ell(y_j, f(x_j; W)) \tag{1}$$

In this way, to minimize $\mathcal{L}(W)$, we applied the Stochastic Gradient Descent (SGD) [33] optimization algorithm with momentum of 0.9, learning rate of 0.001, and batch size of eight. All CNNs were trained for 30 epochs.

F. Evaluation methodology

Due to the very small number of positive images of COVID-19 available, we have decided not to perform hyperparameter optimization nor early stop strategy. These procedures require a validation set, which further reduces the number of images available for testing the models. For this reason, all CNN models were trained and tested using the stratified k-fold cross-validation method [34]. All chest X-ray images were randomly partitioned into ten folds. Then, the model is trained with $k-1$ folds and tested on the remaining fold. The training and testing procedures are repeated k times, alternating the testing folds. Thus, we guarantee that each image will participate in the training process ($k-1$ times) and will also be part of the test group (1 time). Finally, the results from the k testing sets are averaged to produce a single and trustworthy estimation.

The metrics used to assess the classification performance include accuracy, precision, recall, and F1-score indices. All

indices are based on the number of true positives (TP), true negatives (TN), false positives (FP), and false-negative (FN) classifications obtained from the confusion matrix [35]. Also, we measure the standard deviation in order to assess the confidence of results, where smaller values represent high-reliability.

- Accuracy: is the ratio between the correct classifications and total samples (Eq.2).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

- Precision: is the ratio between TP and the total of positives classification (Eq. 3)

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

- Recall: is the harmonic average of recall and precision (Eq. 4).

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

- F1-Score: is the weighted average of the precision and recall (Eq. 5)

$$F1\text{–}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (5)$$

Also, we used the Receiver Operating Characteristic (ROC), and the Area Under ROC (AUC) as a reliable classification performance measure of all possible classification thresholds.

IV. Results and Discussion

All experiments were programmed using Python (version 3.6) and PyTorch (version 1.4) deep learning framework [36]. This study investigated the performance of four CNNs architectures to classify chest X-ray images on COVID-19 positive and COVID-19 negative (pneumonia) classes.

A. Comparison of architectures

One of the most challenges in training CNNs architectures for classification tasks is the lack of large enough datasets for adjust a large number of model parameters. We adopted SFT fine-tuning training strategy to overcome this problem, instead of training the models from scratch (as described in Section III-D). Thus, the weights in the initial layers of CNN (simpler features) were kept, but the weights of the deeper layers (more specialized features) were adapted to the problem of classifying X-ray images generating greater specialization in the deep layers. It is important to mention that number of positive COVID-19 images in the public repository is very small. To overcome this issue, we applied data augmentation strategies aim to increase the size of the training set.

Table II presents the average classification performance considering the results of accuracy, precision, recall and F1-score indices concerning each CNN evaluated. The chart in Fig. 3 illustrate the variation of each performance values based on the results presented in Table II. Interestingly, SqueezeNet achieved the best performance, followed by AlexNet, DenseNet-121, and VGG-11. Also, SqueezeNet is recognized as having a low computational cost and designed for embedded systems. Therefore, the result suggested that our proposed could be used to evaluate chest X-ray images using mobile devices. Moreover, the standard deviation value obtained for each CNN was small and indicates that our results are reliable.

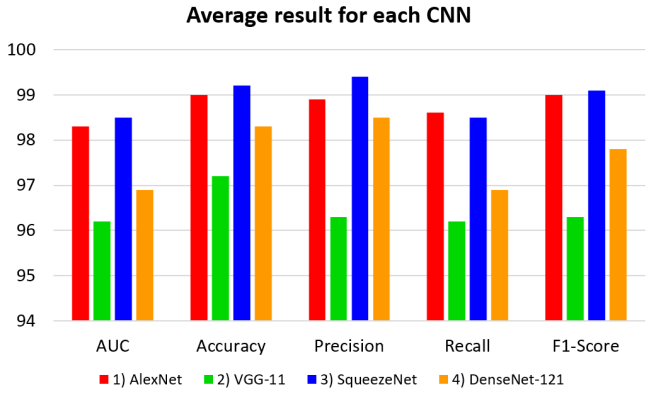


Fig. 3. 10-fold average values of the performance measures for each CNN model. 1) AlexNet; 2) VGG-11; 3) SqueezeNet; and 4) DenseNet-121.

As in each fold the images in the testing sets do not repeat, we consolidate the confusion matrices from each fold by adding the values from each confusion matrix. Therefore, the confusion matrix presented in Table III summarizes results for all ten folds, and presents the prediction for all images in the dataset.

The confusion matrices of each CNN model allow observing several aspects of the classification problem investigated in this work. Note that for all CNNs, the COVID-19 positive and COVID-19 negative is well identified. In particular, AlexNet and SqueezeNet models were able to classify the most positive cases of COVID-19 correctly. The results suggest that models have been able to preserve in the feature maps important information about visual patterns of diagnostic positive. It is important to mention that number of positive COVID-19 images in the public repository is very small. In this study, to compare the performance of different CNN architectures, we focused to obtain reliable and trustfully results.

With the lack of data, we still cannot recommend these methods as a diagnostic aid system, but our results support that the use of CNN models is a promising technique to assist the early diagnosis of COVID-19 in conjugation with other standard tests. However, the number of available images tends to grow as studies advance, and with more accurate researches, it will be possible to understand the capability of CNNs in helping detecting COVID-19.

B. Comparison with literature

The best result achieved in this study in terms of accuracy and F1-score is compared with other state-of-art work in the literature. The best result in our work was obtained with SqueezeNet, trained with SFT using augmented data, which scored 99.20% of accuracy and 99.10% of F1-score (as shown in Table II). The best results reported in the literature are

TABLE II. 10-FOLD AVERAGE VALUES AND STANDARD DEVIATION OF THE PERFORMANCE MEASURES FOR EACH CNN MODEL.

CNN	AUC (%)	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
AlexNet	98.30 ± 0.02	99.00 ± 0.01	98.90 ± 0.02	98.60 ± 0.02	99.00 ± 0.01
VGG-11	96.20 ± 0.04	97.20 ± 0.03	96.30 ± 0.04	98.60 ± 0.04	99.00 ± 0.04
SqueezeNet	98.50 ± 0.02	99.20 ± 0.01	99.40 ± 0.01	98.50 ± 0.02	99.10 ± 0.01
DenseNet-121	96.90 ± 0.03	98.30 ± 0.02	98.50 ± 0.01	96.90 ± 0.03	97.80 ± 0.02

TABLE III. 10-FOLD VALUES OF CONFUSION MATRIX FOR EACH CNN MODEL.

AlexNet

	Positive	Negative
Positive	105	3
Negative	1	298

VGG-11

	Positive	Negative
Positive	102	6
Negative	6	293

SqueezeNet

	Positive	Negative
Positive	105	3
Negative	0	299

DenseNet-121

	Positive	Negative
Positive	102	6
Negative	1	298

presented in Table IV for the same COVID-19 dataset. It can be seen that our best score is upper to the best state-of-the-art technique reported in the literature.

TABLE IV. HIGHEST ACCURACY OF OTHER CLASSIFICATION METHODS USING THE COVID-19 DATASET FROM GITHUB [13].

Method	Accuracy (%)
Narin et al. [12]	98.00
Hemdan et al. [15]	90.00
Sethy and Behera [16]	95.38
Wang and Wong [17]	93.30
Apostopoulos and Mpsiana [18]	96.78
Khan et al. [19]	89.60
Ozturk et al. [21]	98.08
Our work (SqueezeNet + SFT + data aug.)	**99.20**

	F1-Score (%)
Pereira et al. [24]	89.00
Our work (SqueezeNet + SFT + data aug.)	**99.10**

V. CONCLUSION

The results presented in this paper point to a promising using of CNN models to classify COVID-19 cases based on chest X-ray images. We compared the performance of four CNN architectures to classify X-ray images in COVID-19 positive and negative (pneumonia) classes, and our training strategy consists of applying transfer learning with SFT and different data augmentation approaches. Our best result of 99.20% was upper to the highest accuracy score presented in the literature; this result was obtained with SqueezeNet model.

Although there are few COVID-19 positive chest X-ray images, we designed our experiments to minimize the effects of CNN training with small data sets. The training using fine-tuning and data augmentation aim to increase the classification rate, while the stratified k-fold cross validation allows more reliable results than simple hold-out. Now, it is necessary to wait for more images of positive COVID-19 to be available in order to train more reliable models that may confirm the positive perspectives demonstrated by this study.

The presented results open new opportunities towards better machine learning based on deep CNNs for automated detection of COVID-19 and developing of new computer-aided diagnosis applications. Moreover, exciting opportunities and future works raise such as testing other CNN models, evaluating more data augmentation strategies, and applying some hyperparameter optimization and combining classifications techniques.

ACKNOWLEDGMENTS

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the TITAN Xp GPU used for this research. We would like to thanks CAPES and FAPEMIG (Grant number CEX - APQ-02964-17) for the financial support. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

REFERENCES

[1] World Health Organization. Coronavirus disease 2019 (covid-19): situation report, 73. Technical documents, 2020-04-02.

[2] Peng Zhou, Xing-Lou Yang, Xian-Guang Wang, Ben Hu, Lei Zhang, Wei Zhang, Hao-Rui Si, Yan Zhu, Bei Li, Chao-Lin Huang, Hui-Dong Chen, Jing Chen, Yun Luo, Hua Guo, Ren-Di Jiang, Mei-Qin Liu, Ying Chen, Xu-Rui Shen, Xi Wang, Xiao-Shuang Zheng, Kai Zhao, Quan-Jiao Chen, Fei Deng, Lin-Lin Liu, Bing Yan, Fa-Xian Zhan, Yan-Yi Wang, Geng-Fu Xiao, and Zheng-Li Shi. A pneumonia outbreak

associated with a new coronavirus of probable bat origin. *Nature*, 579(7798):270–273, 2020.

[3] Graziano Onder, Giovanni Rezza, and Silvio Brusaferro. Case-Fatality Rate and Characteristics of Patients Dying in Relation to COVID-19 in Italy. *JAMA*, 03 2020.

[4] Wei-jie Guan, Zheng-yi Ni, Yu Hu, Wen-hua Liang, Chun-quan Ou, Jian-xing He, Lei Liu, Hong Shan, Chun-liang Lei, David S.C. Hui, Bin Du, Lan-juan Li, Guang Zeng, Kwok-Yung Yuen, Ru-chong Chen, Chun-li Tang, Tao Wang, Ping-yan Chen, Jie Xiang, Shi-yue Li, Jin-lin Wang, Zi-jing Liang, Yi-xiang Peng, Li Wei, Yong Liu, Ya-hua Hu, Peng Peng, Jian-ming Wang, Ji-yang Liu, Zhong Chen, Gang Li, Zhi-jian Zheng, Shao-qin Qiu, Jie Luo, Chang-jiang Ye, Shao-yong Zhu, and Nan-shan Zhong. Clinical characteristics of coronavirus disease 2019 in china. *New England Journal of Medicine*, 382(18):1708–1720, 2020.

[5] Giacomo Grasselli, Antonio Pesenti, and Maurizio Cecconi. Critical Care Utilization for the COVID-19 Outbreak in Lombardy, Italy: Early Experience and Forecast During an Emergency Response. *JAMA*, 323(16):1545–1546, 04 2020.

[6] Loris Nanni, Stefano Ghidoni, and Sheryl Brahnam. Ensemble of convolutional neural networks for bioimage classification. *Applied Computing and Informatics*, 2018.

[7] Daniel S. Kermany, Michael Goldbaum, Wenjia Cai, Carolina C.S. Valentim, Huiying Liang, Sally L. Baxter, Alex McKeown, Ge Yang, Xiaokang Wu, Fangbing Yan, Justin Dong, Made K. Prasadha, Jacqueline Pei, Magdalene Y.L. Ting, Jie Zhu, Christina Li, Sierra Hewett, Jason Dong, Ian Ziyar, Alexander Shi, Runze Zhang, Lianghong Zheng, Rui Hou, William Shi, Xin Fu, Yaou Duan, Viet A.N. Huu, Cindy Wen, Edward D. Zhang, Charlotte L. Zhang, Oulan Li, Xiaobo Wang, Michael A. Singer, Xiaodong Sun, Jie Xu, Ali Tafreshi, M. Anthony Lewis, Huimin Xia, and Kang Zhang. Identifying medical diagnoses and treatable diseases by image-based deep learning. *Cell*, 172(5):1122 – 1131.e9, 2018.

[8] Larissa Ferreira Rodrigues, Murilo Coelho Naldi, and João Fernando Mari. Comparing convolutional neural networks and preprocessing techniques for hep-2 cell classification in immunofluorescence images. *Computers in Biology and Medicine*, 116:103542, 2020.

[9] Yicheng Fang, Huangqi Zhang, Jicheng Xie, Minjie Lin, Lingjun Ying, Peipei Pang, and Wenbin Ji. Sensitivity of chest ct for covid-19: Comparison to rt-pcr. *Radiology*, 0(0):200432, 0. PMID: 32073353.

[10] Tao Ai, Zhenlu Yang, Hongyan Hou, Chenao Zhan, Chong Chen, Wenzhi Lv, Qian Tao, Ziyong Sun, and Liming Xia. Correlation of chest ct and rt-pcr testing in coronavirus disease 2019 (covid-19) in china: A report of 1014 cases. *Radiology*, 0(0):200642, 0. PMID: 32101510.

[11] Geoffrey D. Rubin, Christopher J. Ryerson, Linda B. Haramati, Nicola Sverzellati, Jeffrey P. Kanne, Suhail Raoof, Neil W. Schluger, Annalisa Volpi, Jae-Joon Yim, Ian B.K. Martin, Deverick J. Anderson, Christina Kong, Talissa Altes, Andrew Bush, Sujal R. Desai, Jonathan Goldin, Jin Mo Goo, Marc Humbert, Yoshikazu Inoue, Hans-Ulrich Kauczor, Fengming Luo, Peter J. Mazzone, Mathias Prokop, Martine Remy-Jardin, Luca Richeldi, Cornelia M. Schaefer-Prokop, Noriyuki Tomiyama, Athol U. Wells, and Ann N. Leung. The role of chest imaging in patient management during the covid-19 pandemic: A multinational consensus statement from the fleischner society. *Chest*, 2020.

[12] Ali Narin, Ceren Kaya, and Ziynet Pamuk. Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks, 2020.

[13] Joseph Paul Cohen, Paul Morrison, and Lan Dao. Covid-19 image data collection. *arXiv*, 2020.

[14] P Mooney. Chest x-ray images (pneumonia). *kaggle, Marzo*, 2018.

[15] Ezz El-Din Hemdan, Marwa A. Shouman, and Mohamed Esmail Karar. Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images, 2020.

[16] Prabira Kumar Sethy and Santi Kumari Behera. Detection of coronavirus disease (covid-19) based on deep features. *Preprints*, 2020030300:2020, 2020.

[17] Linda Wang and Alexander Wong. Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images, 2020.

[18] Ioannis D. Apostolopoulos and Tzani A. Mpesiana. Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks. *Physical and Engineering Sciences in Medicine*, pages 1–6, Apr 2020. PMC7118364[pmcid].

[19] Asif Iqbal Khan, Junaid Latief Shah, and Mohammad Mudasir Bhat. Coronet: A deep neural network for detection and diagnosis of covid-19 from chest x-ray images. *Computer Methods and Programs in Biomedicine*, 196:105581, 2020.

[20] J. Deng, W. Dong, R. Socher, L. J. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.

[21] Tulin Ozturk, Muhammed Talo, Eylul Azra Yildirim, Ulas Baran Baloglu, Ozal Yildirim, and U. [Rajendra Acharya]. Automated detection of covid-19 cases using deep neural networks with x-ray images. *Computers in Biology and Medicine*, 121:103792, 2020.

[22] Ferhat Ucar and Deniz Korkmaz. Covidiagnosis-net: Deep bayes-squeezenet based diagnosis of the coronavirus disease 2019 (covid-19) from x-ray images. *Medical Hypotheses*, 140:109761, 2020.

[23] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and ¡0.5mb model size, 2016.

[24] Rodolfo M. Pereira, Diego Bertolini, Lucas O. Teixeira, Carlos N. Silla, and Yandre M.G. Costa. Covid-19 identification in chest x-ray images on flat and hierarchical classification scenarios. *Computer Methods and Programs in Biomedicine*, 194:105532, 2020.

[25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[26] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[27] M. A. Ponti, L. S. F. Ribeiro, T. S. Nazare, T. Bui, and J. Collomosse. Everything you wanted to know about deep learning for computer vision but were afraid to ask. In *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, pages 17–41, Oct 2017.

[28] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[29] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[30] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[31] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, 35(5):1299–1312, 2016.

[32] Mohammadhassan Izadyyazdanabadi, Evgenii Belykh, Michael Mooney, Nikolay Martirosyan, Jennifer Eschbacher, Peter Nakaji, Mark C. Preul, and Yezhou Yang. Convolutional neural networks: Ensemble modeling, fine-tuning and unsupervised semantic localization for neurosurgical cle images. *Journal of Visual Communication and Image Representation*, 54:10 – 20, 2018.

[33] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov 1998.

[34] Pierre A. Devijver and Josef Kittler. *Pattern Recognition: A Statistical Approach.* Prentice-Hall, 1982.

[35] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2Nd Edition)*. Wiley-Interscience, New York, NY, USA, 2000.

[36] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc., 2019.

Segmentation of fish chromosomes in microscopy images: A new dataset

Rodrigo Júnior Rodrigues*, Rubens Pasa†, Karine Frehner Kavalco†, João Fernando Mari*
Universidade Federal de Viçosa - Campus Rio Paranaíba,
*Instituto de Ciências Exatas e Tecnológicas,
†Laboratório de Bioinformática e Genômica,
Caixa Postal 22 – 38.810-000 – Rio Paranaíba – MG – Brazil
Email: {rodrigo.rodrigues, joaof.mari}@ufv.br

***Abstract*—The chromosome segmentation is the most important step in automatic karyotype assembling. In this work, we presented a brand new chromosome image dataset and proposed methods for segmenting the chromosomes. Chromosome images are usually low quality, especially fish chromosomes. In order to overcome this issue, we tested three filters to reduce noise and improve image quality. After filtering, we applied adaptive threshold segmentation combined with mathematical morphology and supervised classification methods. Support Vector Machine and k-nearest neighbors were applied to discriminate between chromosomes and image background. The proposed method was applied to segment chromosomes in a new dataset. To enable measure the performance of the methods all chromosomes were manually delineated. The results are evaluated considering the Hausdorff distance and normalized sum of distances between segmented and reference images.**

***Index Terms*—Fish karyotype, chromosome segmentation, computer vision, classification, new dataset.**

I. Introduction

The karyotype assembling is an important task in cytogenetics. It is useful in a number of practical and research activities, such as assist the diagnosis of genetic diseases and some types of cancer types [1]. The chromosomes are composed of supercoiled and associated DNA. Human chromosomes can suffer related anomalies to an atomic number of chromosomes or structural abnormality in one or more chromosomes [2]. The human cells contain 46 chromosomes including 22 pairs of chromosome and two sex chromosomes (XY: male and XX: female). Fishes have a variable number of chromosomes and they cannot be previously defined as in humans [3]. The process of chromosomal karyotyping is performed by pairing the chromosomes according to the similarity between them. The chromosomes are classified in one of the four classes according to the location of the centromere: metacentric, submetacentric, subtelocentric, and acrocentric [4] [5].

In the process of segmentation, the images are first converted to binary format. The binary images help to find details about the object shapes [6]. Chromosomes are cellular structures that contain genetic information. When chromosomes are imaged using a microscopy, information about the health of an individual. Since the 1980s, chromosome detection and classification systems have aroused great interest in research. The manual assembling of a karyotype is repetitive, exhausting, time consuming, and subject to error. It can be performed by visual analysis but requires specialized professionals.

The segmentation of chromosome is the most important step in automated analysis of chromosomes [7]. An automated system generally includes the following four steps: (1) image enhancement, (2) segmentation and alignment of the chromosomes, (3) chromosome feature selection, and (4) chromosome classification. The chromosome segmentation is the most important step because their results can affect the performance of the entire system [8].

In this work, we study and compare approaches for segmentation of fish chromosomes in digital images combining filtering operations, segmentation, and morphological operations. Mean filter, median filter, and Non Local Means filter are used to reduce the noise and improve the image quality. Segmentation is performed using adaptive threshold followed by morphological operations.

Supervised classifiers, such as Vector Support Machine (SVM) and k-nearest neighbor (KNN) are applied to discriminate the segmented objects in chromosomes and artifacts (objects that are not chromosomes). The methods were implemented and tested in an image dataset with ground-truth. Finally, we analyze the methods performance considering Hausdorff distance and NSD metrics and compare the implemented approaches.

This paper is organized as follows: This section introduces the subject. Section II shows some related work on chromosome segmentation. Section III describes the new image dataset we created, as well as the proposed methods to automatically segment the chromosomes and the validation methods. In Section IV we present and discuss the results and the conclusion and future works are in Section V.

II. Related Works

Aln W. and Jane Y. [9] developed an algorithm based on an adaptive local kernel (KAFCM) and a classifier of Probabilistic defuzzification to improve segmentation and classification of chromosomes. This is achieved on a window for each pixel and compensate for the intensity of the homogeneity caused during the process of generation of images and by the preparation of the physical chromosome itself. The algorithm was tested on a publicly available dataset and the results were compared

with traditional fuzzy clustering algorithms. The classification results for the proposed method are for defuzzification of standard FCM were compared, and the proposed classification method demonstrated an improved overall.

Monika S. et al. [10] proposed a method to segment and classify chromosomes in healthy patients combining deep-learning and pre-processing methods and crowd-sourcing. The experiments are performed on 400 images taken from healthy patients. For the subset with better images quality, the classification rate is about 95%

Madian et. al. [11] studied the chromosome segmentation considering boundary information. Otsu threshold, morphological operations, and filling holes after binarization were applied. A curvature function was applied to find cut-off points in the object edges. The concavity points at the edges are used to detect chromosomes overlapping zones. The method has been tested on over 350 images with several degrees of overlap and obtained an overall accuracy of 96%.

Karvelis *et al.* [12] present a method for the segmentation of groups of chromosomes that touch each other and chromosomes superimposed on M-FISH images. Initially, the watershed transform is applied and the image is decomposed in regions. Gradient paths are calculated from points of high concavity and used to divide the groups of chromosomes. To validate the method they used a reference dataset composed of 183 M-FISH images. The algorithm resulted in a success rate of 90.6% for the chromosomes that are touched and of 80.4% for the groups of chromosomes that are superimposed.

Rodrigues et al. [13] compared two approaches to segment overlapping chromosomes, one based on morphological skeleton and the other based on restricted Delaunay triangulation. Restricted Delaunay triangulation demonstrates to achieve better results then morphological skeleton.

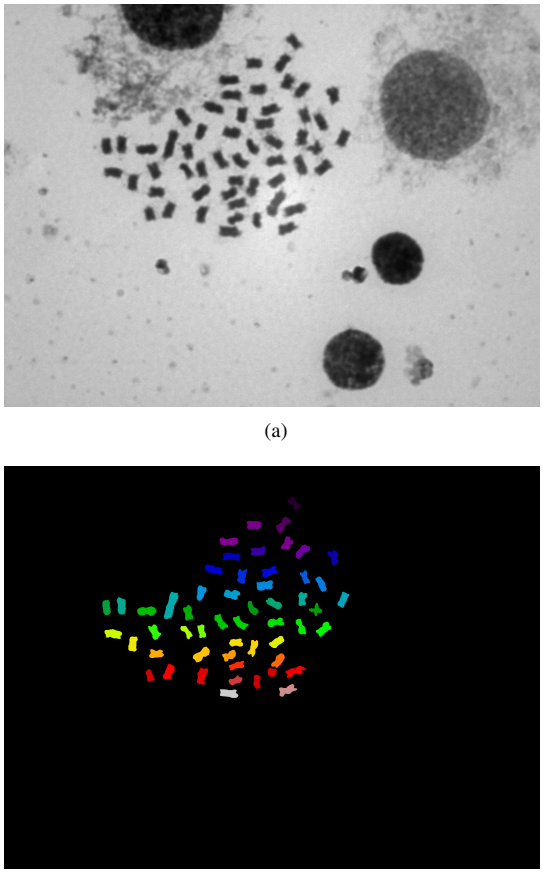
Saiyod and Wayalum [14] developed an approach to compute the skeleton of the chromosomes. With the skeletons it is possible to search for points of intersection. The point of intersection is used to search candidate cut points. The cut-off points are found by calculating the Euclidean distance from the point of intersection to the points of curvature. The nearest four points are the points of interest

III. MATERIAL AND METHODS

A. Dataset

The dataset was constructed in the Bioinformatics and Genomics Laboratory, at the campus of UFV in Rio Paranaíba - Brazil. The images were captured using a microscope Olympus BX 41 (Olympus Inc., Japan) with a 3 MP and a magnification of 1000x using the software Qcapture Pro 6.0 (QImagine, Surrey, BC, Canada). The images were converted to grayscale and for each image, we created a reference image in which each chromosome were manually segmented using the image processing software Gimp. Each image has a size of 1250 x 1250 pixel and have been saved as hdf5 file forming (Figure 1).

[0]Available in https://www.gimp.org



(a)

(b)

Fig. 1: The Chromosome dataset. (a) Original image (b) Labeled image

B. Filtering

Generally, chromosome images have low quality, they are low contrast and it is possible to observe the presence of noise and artifacts. This problem is worse when we are dealing with fish chromosomes because they are smaller than human chromosomes. Thus, the preprocessing step is very important for the segmentation [14]. We tested three filters: (1) median filter with mask size of 3 x 3; (2) average filter with mask size of 5 x 5 [15]; and (3) Non-Local Means filter (NLM) [16] [17] with standard deviation of 0.08 and h of 0.6.

C. Image segmentation in the background and chromosomes

In order to segment the images in pixels belonging to chromosomes and background pixels, we used a local adaptive threshold with a block size of 45 pixels. The block size was chosen empirically. After some experiments, we noticed that very large block sizes tend to generate connected chromosomes.

Mathematical morphology algorithms were applied to improve the quality of the binary image. Fill holes algorithms, based on morphological reconstruction [15] were applied to prevent holes inside the objects that may interfere with the

chromosome classification. A morphological opening operation using a disk-shaped structuring element with radius 2 is applied to break some isthmus and smooth out the object contours. Finally, we removed small objects (less than 120 pixels) which consist of artifacts resulting from the threshold segmentation [15] and removed the objects in the image borders.

D. Classification

A set of five features were selected to classify the connected components in chromosome or artifacts: (1) the area, (2) solidity, (3) eccentricity, (4) equivalent diameter, and (4) mean intensity. We divided our dataset in 80% of the images for training and 20% for testing. The features of all objects in the training set were used for training a Support Vector Machine (SVM) and a K-Nearest Neighbor (KNN) classifier. The dataset was split in an image-wise fashion since the chromosomes on each test image should be presented to the classifier only in the testing step.

After the training, the models were evaluated classifying the objects in the testing set in chromosomes or artifacts (any other segmented object which does not correspond to a chromosome). The metrics used to evaluate the classification were those derived from the confusion matrices: Precision (Equation 1); Recall (Equation 2), and F1-score (Equation 3) [18] [19]. These metrics are used to evaluate the performance of classifiers in the proposed methods.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 - Score = \frac{2 \times (Precision \times Recall)}{Precision + Recall} \tag{3}$$

where TP, FP and FN are *True Positive*, *False Positive* and *False Negative*, respectively.

E. Validation

To verify the efficiency and compare the methods described in this work we used the Hausdorff distance and the normalized sum of distances (NSD) methods. The Hausdorff distance is the largest minimum distance between the object borders in the segmented image I and in the reference image R , according to the Equation 4:

$$Hausdorff(I, R) = \max D(i) : S_i \neq R_i \tag{4}$$

where $D(i)$ is the distance between the pixel i of the object and the border of the reference object. The value 0 indicates a perfect segmentation, however, the index does not have an upper limit [20].

The normalized sum of distances (NSD) between the segmented image I and the reference image R is defined by Equation 5.

$$NSD(I, R) = \frac{\sum_i I_i \neq R_i * D(i)}{\sum_i D(i)} \tag{5}$$

where $D(i)$ is the distance between the pixel i and the border of the reference object. The value 0 indicates a segmentation perfect while 1 indicates that there is no overlap between the segmented cell and to the reference cell.

IV. Results

All images in the dataset described in Section III-A, a total of 97 images, were filtered according to the procedures described in Section III-B. Then the images were segmented in chromosome pixels and background pixels as described in Section III-C. The objects in the segmented images were manually labeled in actual chromosomes and artifacts (all segmented objects that are not chromosomes). These images were split in training and test sets which a proportion of 80 % and 20 %, as described in Section III-D and used to train an SVM and KNN classifiers.

Table I shows the classification results when the images were submitted to the mean filter, Table II is for when the images were submitted to the median filter, and Table III is for the NLM. We can observe the KNN had better accuracy, recall, and f1-score for all filtering strategies. These values where computed over objects in the testing set.

TABLE I: Classification results between SVM and KNN when applying the **mean filter**.

	precision	**recall**	**f1-score**
SVM	0.77	0.82	0.78
KNN	0.79	0.83	0.80

TABLE II: Classification results between SVM and KNN when applying the **median filter**

	precision	**recall**	**f1-score**
SVM	0.78	0.82	0.79
KNN	0.78	0.82	0.79

TABLE III: Classification results between SVM and KNN when applying the **NLM filter**

	precision	**recall**	**f1-score**
SVM	0.78	0.82	0.78
KNN	0.79	0.83	0.80

Tables IV, V, and VI shows the final segmentation results in terms of Hausdorf Distance and NSD. Table IV is for when images where filtered with mean filter, Table V, is for median filter, and VI is for NLM. As we expected, based on

results in Tables I to I, KNN overcomes SVM considering all filtering strategies. Regarding the filtering strategy, the results are very close to each other. Before the classification median and NLM filtering are slightly superior than mean filter in therms of Hausdorf distance. After the classification, the values continues very close to each other, but it is clear that the classification process is essential for a good segmentation. These results were obtained after applying the specific filter process (Section III-B), the segmentation strategy described in Section III-C and, finally, the object classification with the trained models whose results are described in Tables I to III.

TABLE IV: Hausdorff distances, and NSD between the images resulting from the SVM and KNN with **mean filter** and the reference images.

	Hausdorff	**NSD**
No class.	7.03	0.69
SVM	7.38	0.46
KNN	6.50	0.37

TABLE V: Hausdorff distances, and NSD between the images resulting from the SVM and KNN with **median filter** and the reference images.

	Hausdorff	**NSD**
No class.	6.98	0.69
SVM	7.30	0.46
KNN	6.53	0.36

TABLE VI: Hausdorff distances, and NSD between the images resulting from the SVM and KNN with **NLM filter** and the reference images.

	Hausdorff	**NSD**
No class.	6.97	0.69
SVM	7.27	0.46
KNN	6.57	0.36

Figure 2 shows some segmented images using NLM filter and after classification with KNN. The first row shows the original image in grayscale, the second row shows the segmentation before classification, the third row shows the final segmentation after the classification, and the fourth row shows the the image considered as ground-truth.

V. Conclusions

This paper presented a comparison of approaches to segment chromosomes in microscopy images combining filtering techniques, adaptive thresholding, and classification methods.

Experiments were carried out using a newly constructed dataset of fish chromosome images with ground-truth. The images were obtained from the Bioinformatics and Genomics Laboratory of the Federal University of Viçosa in Rio Paranaíba - Brazil. Each image had its chromosomes segmented manually and saved in an h5py dataset. This dataset allows the development of this study, which investigate chromosome segmentation methods in a pragmatic way but also will be useful to a number of future works.

A number of filtering methods are compared (mean filter, median filter, and NLM filter) and tested in conjunction with two supervised classifiers used to improve the segmentation results. It can be seen that the filtering strategies have small effect over the segmentation results, however the object classification has a high impact on the quality of the results. The KNN classifier showed to be better than SVM for this task. Even so, images of fish chromosomes in metaphase state have a very large amount of noise and filtering strategy is still very important.

As future work one can consider testing the using the individually transformed watershed algorithm on chromosomes that touch each other. Another approach using neural networks and deep learning to test the outcome of the segmentation. To apply a method of classification for the chromosomes in metacentric, submetacentric, subtelocentric, and acrocentric for mounting the fish karyotype. And finally, other different metrics to evaluate the quality of segmentation.

Acknowledgment

The authors are grateful to the Bioinformatics and Genomics Laboratory for providing the image dataset. Thanks also to Capes and FAPEMIG. We would like to thanks CNPq for financial support. We would like to thanks CAPES and FAPEMIG (Grant number CEX - APQ-02964-17) for financial support. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

[1] S. Minaee, M. Fotouhi, B. H. Khalaj, A Geometric Approach For Fully Automatic Chromosome Segmentation (2011) 1–8arXiv:1112.4164.

[2] N. Madian, K. B. Jayanthi, Overlapped chromosome segmentation and separation of touching chromosome for automated chromosome classification, Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS (2012) 5392–5395.

[3] O. M. Filho, L. A. C. Bertollo., Análise cromossômica de astyanax scabripinnis rivularis (characiformes, characidae) da região Três Marias MG., Cienc. Cult (1986) 35:855.

[4] A. Levan, K. Fredga, A. A. Sandberg, Nomenclature for centromeric position on chromosomes, Hereditas 52 (2) (1964) 201–220.

[5] R. Manohar, J. Gawande, Watershed and clustering based segmentation of chromosome images, in: Advance Computing Conference (IACC), 2017 IEEE 7th International, IEEE, 2017, pp. 697–700.

[6] D. Somasundaram, V. R. Vijay Kumar, Separation of overlapped chromosomes and pairing of similar chromosomes for karyotyping analysis, Measurement: Journal of the International Measurement Confederation 48 (1) (2014) 274–281. doi:10.1016/j.measurement.2013.11.024.

[7] M. V. Munot, M. Joshi, N. Sharma, G. Ahuja, Automated detection of cut-points for disentangling overlapping chromosomes, in: 2013 IEEE Point-of-Care Healthcare Technologies (PHT), 2013, pp. 120–123.

[8] W. Yan, D. Li, Segmentation algorithms of chromosome images, in: Proceedings of 2013 3rd International Conference on Computer Science and Network Technology, 2013, pp. 1026–1029.

[9] A. W. Dougherty, J. You, A Kernel-based adaptive Fuzzy C-Means algorithm for M-FISH image segmentation, 2017 International Joint Conference on Neural Networks (IJCNN) (2017) 198–205.

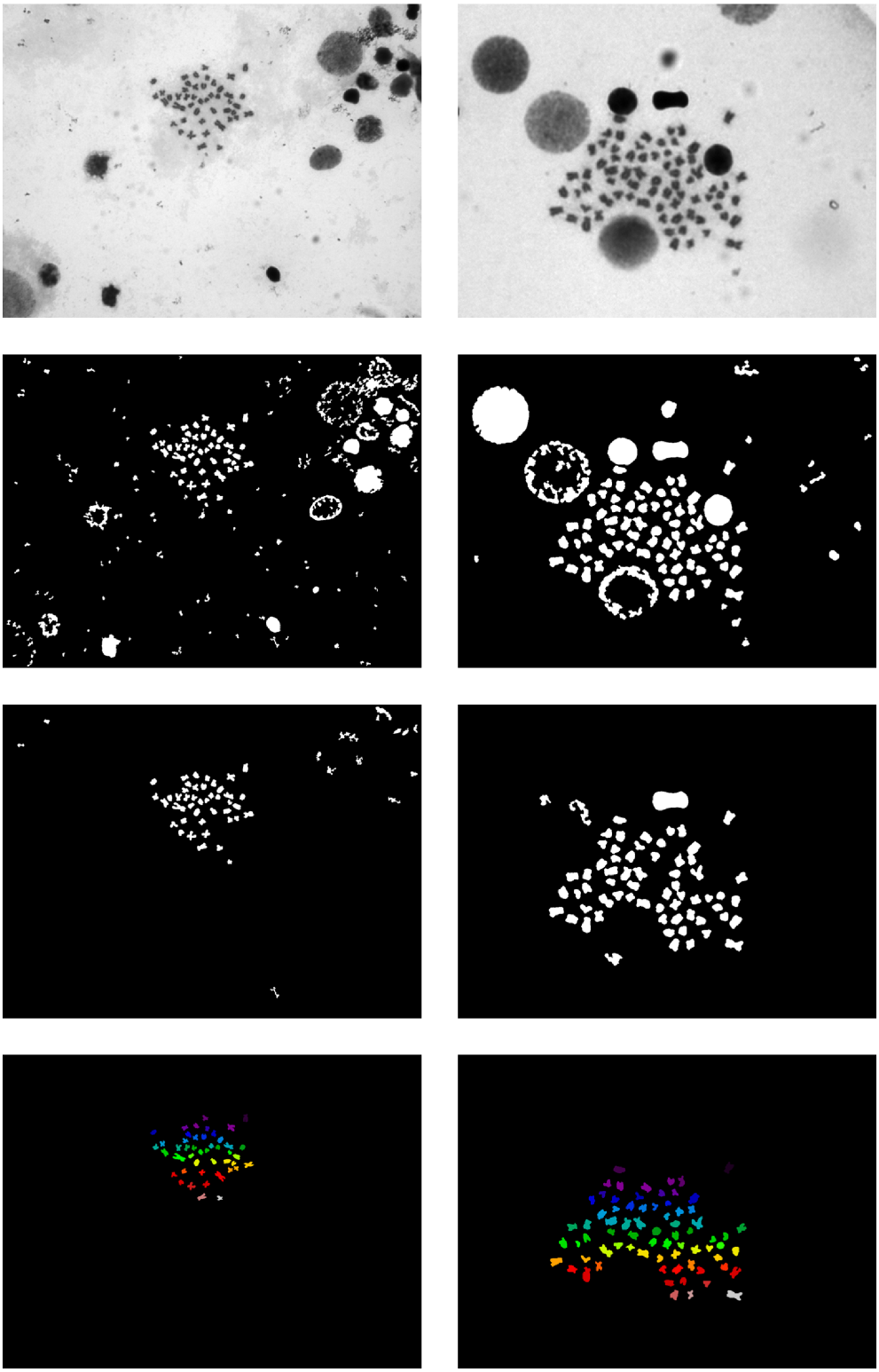


Fig. 2: Segmentation result using NLM filter. The first row shows the original image, the second column the segmented image without classification, the third row shows the final segmented images after classification, and the fourth row shows the reference image.

[10] M. Sharma, O. Saha, A. Sriraman, R. Hebbalaguppe, L. Vig, S. Karande, Crowdsourcing for Chromosome Segmentation and Deep Classification, IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops 2017-July (2017) 786–793. doi:10.1109/CVPRW.2017.109.

[11] N. Madian, K. B. Jayanthi, S. Suresh, Contour based segmentation of chromosomes in g-band metaphase images, in: 2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP), 2015, pp. 943–947.

[12] P. Karvelis, A. Likas, D. I. Fotiadis, Identifying touching and overlapping chromosomes using the watershed transform and gradient paths, Pattern Recognition Letters 31 (16) (2010) 2474–2488.

[13] R. J. Rodrigues, W. F. Gonçalves, J. F. Mari, A comparison between two approaches to segment overlapped chromosomes in microscopy images, in: Anais do XIII Workshop de Visão Computacional, 2017, pp. 118–123.

[14] S. Saiyod, P. Wayalun, A hybrid technique for overlapped chromosome segmentation of g-band matasprеad images automatic, in: 2014 Fourth International Conference on Digital Information and Communication Technology and its Applications (DICTAP), 2014, pp. 400–404.

[15] R. Gonzalez, R. Woods, Digital Image Processing, Pearson/Prentice Hall, 2008.

[16] A. Buades, B. Coll, J.-M. Morel, A non-local algorithm for image denoising, in: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), Vol. 2, IEEE, 2005, pp. 60–65.

[17] J. V. Manjón, J. Carbonell-Caballero, J. J. Lull, G. García-Martí, L. Martí-Bonmatí, M. Robles, Mri denoising using non-local means, Medical Image Analysis 12 (4) (2008) 514 – 523.

[18] J. Fritsch, T. Kuehnl, A. Geiger, A new performance measure and evaluation benchmark for road detection algorithms, in: 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), IEEE, 2013, pp. 1693–1700.

[19] F. Ge, S. Wang, T. Liu, Image-segmentation evaluation from the perspective of salient object extraction, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), Vol. 1, IEEE, 2006, pp. 1146–1153.

[20] L. P. Coelho, A. Shariff, R. F. Murphy, Nuclear segmentation in microscope cell images: A hand-segmented dataset and comparison of algorithms, Proceedings - 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, ISBI 2009 (2009) 518–521.

Domain Adaptation for Robust Face Recognition Using Transfer Kernel Learning

João Renato Ribeiro Manesco
UNESP - São Paulo State University
Bauru, Brazil
joao.r.manesco@unesp.br

Aparecido Nilceu Marana
UNESP - São Paulo State University
Bauru, Brazil
nilceu.marana@unesp.br

***Abstract*—In the last decades, for reasons of safety or convenience, biometric characteristics are increasingly being used to identify individuals who wish to have access to systems or places, and facial features are one of the most used characteristics for this purpose. For biometric identification to be effective, the recognition accuracy rates must be high. However, these rates can be very low depending on the difference (displacement) between the domain of the images stored in the database of the biometric system (source images) and the images used at the moment of identification (target images). In this work, we evaluated the performance of a domain adaptation method called Transfer Kernel Learning (TKL) in the face recognition problem. Results obtained in our experiments on two face datasets, ARFace and FRGC, corroborates that TKL is suitable for domain adaptation and that it is capable of improving significantly the accuracy rates of face recognition, even when considering facial images with occlusions, variations in illumination and complex backgrounds.**

***Index Terms*—biometrics, face recognition, domain adaptation, transfer kernel learning.**

I. Introduction

Recently, either for security or convenience reasons, biometric recognition is gaining popularity among applications that aim to provide access to a particular system or place. Since facial features can be extracted from most people in our society, they end up being one of the most used features for these kinds of applications [1], [2].

Even though facial features are widely used, a few factors can decrease the system's performance in facial recognition tasks. Among these, we can cite the differences among the capture sensors, illumination changes, age disparity, and occlusion of the facial region [3].

An example in which we can see these variations is in mobile banking, where they need to authenticate users with their faces, to open accounts, or authorize transactions, usually from images of distinct origins like IDs and *selfies* [4], [5]. In Figure 1 we can see those differences, wherein the upper row we have images from the Brazilian national ID card, with constant illumination, and without pose variation, whereas in the lower row we have *selfie* images with differences in the capture sensor and variations among illumination and pose.

This disparity between image domains brings up a problem called domain shift, in which the distribution of the source classification train data differs from the test data distribution, decreasing the performance of the classification task [6]. This kind of performance loss is a big problem in biometric identification systems, in which high accuracy is needed for the system to work effectively.

Fig. 1: Examples of face images obtained from four people on the domain characterized by the Brazilian national ID card pictures (upper row) and on the domain characterized by *selfie* pictures (lower row).

A way to deal with the domain shift problem is using domain adaptation techniques, a subarea of transfer learning that uses labeled data from a source domain to improve the classification task in a target domain [7].

In this paper, we approach the face recognition problem using a domain adaptation technique called Transfer Kernel Learning [8] to improve the accuracy of the identification task.

II. Domain Adaptation

Domain adaptation is a subarea of transfer learning that aims to learn from a source data distribution a model with good performance on a distinct target data distribution [9]. Pan and Yang [10] define the following key concepts related to domain adaptation theory.

Definition 1 (Domain)**.** *A domain D is composed of a feature space $\mathcal{F}$ with d dimensions and a marginal probability function $P(x)$, which means that $D = \{\mathcal{F}, P(x)\}$, with $x \in \mathcal{F}$.*

Definition 2 (Task)**.** *Given a domain D, a task $\mathcal{T}$ consists of a set of labels $\mathcal{Y}$ and a classifier $f(x)$, which means that $\mathcal{T} = \{\mathcal{Y}, f(x)\}$, with $y \in \mathcal{Y}$ and $f(x) = P(y|x)$.*

Definition 3 (Domain Adaptation)**.** *Given a source domain $\mathcal{D}_\mathcal{S}$ and a target domain $\mathcal{D}_\mathcal{T}$ and assuming that $\mathcal{D}_\mathcal{S} \neq \mathcal{D}_\mathcal{T}$ regarding their marginal probabilities $P(X^\mathcal{S}) \neq P(X^\mathcal{T})$, and two tasks $\mathcal{T}_\mathcal{S} \approx \mathcal{T}_\mathcal{T}$, with conditional distribution $P(Y^\mathcal{S}|X^\mathcal{S}) \approx P(Y^\mathcal{T}|X^\mathcal{T})$. The goal of the domain adaptation is to improve the prediction $f_\mathcal{T}(\cdot)$ in the target domain $\mathcal{D}_\mathcal{T}$, using the source domain $\mathcal{D}_\mathcal{S}$ data.*

Essentially, the domain adaptation objective is to improve the predictive characteristic of a target domain with a different marginal probability, using data found in the source domain.

A. Transfer Kernel Learning

Transfer Kernel Learning (TKL) [8] is a promising domain adaptation technique in tasks related to visual recognition. This technique aims to use source and target data to learn a domain-invariant kernel that minimizes the domain variance and is used to feed a kernel machine. Its formal problem is described in Problem 1.

Problem 1 (Transfer Kernel Learning [8])**.** *Given a labeled domain $\mathcal{Z} = \{(z_1, y_1), ..., (z_m, y_m)\}$ and an unlabeled target domain $\mathcal{X} = \{x_1, \ldots, x_n\}$, with $\mathcal{F}_\mathcal{Z} = \mathcal{F}_\mathcal{X}$, $\mathcal{Y}_\mathcal{Z} = \mathcal{Y}_\mathcal{X}$, $P(z) \neq P(x)$ and $P(y|z) \neq P(y|x)$, learn a domain-invariant kernel $k(z, x) = \langle \phi(z), \phi(x) \rangle$, such that $P(\phi(z)) \approx P(\phi(x))$. Assume $P(y|\phi(z)) \approx P(y|\phi(x))$ so kernel machines trained on $\mathcal{Z}$ can generalize well on $\mathcal{X}$.*

The TKL method follows the principle that even though metrics like the Maximum Mean Discrepancy can find information about the domain variation, they aren't explored properly, being used only as a penalty to standard learning methods and this will not properly achieve a local minimum in the variation.

The problem is explored by applying standard eigen decomposition on the target kernel matrix K_X and then evaluating the eigensystem on the source data, by using the Mercer Theorem [8], finding the extrapolated eigenvectors of the source domain data by the equation 1

$$\overline{\Phi}_Z \simeq K_{ZX} \Phi_X \Lambda_X^{-1} \tag{1}$$

in which Φ refers to the set of eigenvalues of a particular domain, Λ is set of eigenvectors, and K_{ZX} the cross-domain kernel between Z and X, evaluated using the kernel function k. The eigenvectors are then used in the Nyström Kernel Approximation [11] to find a family of kernels K_Z, extrapolated from a target eigensystem but evaluated on source data.

This family preserves the key structures of the target domain but does not necessarily minimize the domain variance, this is achieved by relaxing the target domain eigenvalues Λ_X to a set of Λ eigenvalues that can be used in a quadratic minimization problem involving $\overline{\Phi}_Z$, K_Z and a damping factor ζ that can be tuned.

After finding the optimized Λ eigenvalues, it is possible to find a domain invariant kernel matrix, $\overline{K}_A$, described in the equation 2.

$$\overline{K_A} = \begin{bmatrix} \overline{\Phi}_Z \Lambda \overline{\Phi}_Z{}^T & \overline{\Phi}_Z \Lambda \overline{\Phi}_X{}^T \\ \overline{\Phi}_X \Lambda \overline{\Phi}_Z{}^T & \overline{\Phi}_X \Lambda \overline{\Phi}_X{}^T \end{bmatrix} \tag{2}$$

By finding the domain invariant kernel matrix, we can use the source data portion $\overline{\Phi}_Z \Lambda \overline{\Phi}_Z{}^T$,or simply K_{AZZ}, to train a kernel machine, like an SVM and evaluate the performance on the target data portion K_{AXZ} found in the kernel matrix as $\overline{\Phi}_X \Lambda \overline{\Phi}_Z{}^T$. We can see the overall procedure of the TKL method in Figure 2.

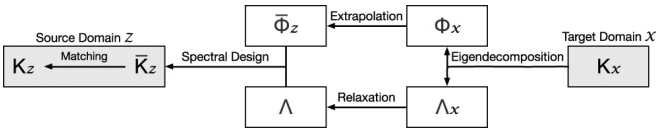


Fig. 2: Overall procedure of the Transfer Kernel Learning (TKL) method [8].

III. Face Recognition

Face recognition is the most common identification method used by humans since it has a high acceptance in society and provides a non-intrusive collaboration with the system, as opposed to iris or fingerprint recognition, in which an individual has to directly interact with the system [3].

A face recognition system can operate in two ways, authentication and identification [1]. An authentication system will match the user face with another face, of who he claims to be, acquired from a face database, and assert if he is that person. On the other hand, an identification system will receive a face as an input and will verify which person he or she is.

Two fundamental phases are required for the proper behavior of a facial recognition system, the detection of the face region, and the feature extraction of the detected faces.

A. Face Detection

Face detection is an essential phase of face recognition, responsible for detecting the face region in an image and enabling proper feature extraction in the posterior phases.

Given the wide range of variations among facial images, face detection is a challenging task, especially in an unconstrained environment, but recently, with the advances in deep learning, some very effective approaches are appearing. In our work, face detection is conducted by using a method named Multi-Task Cascaded Convolutional Neural Network (MTCNN) [12].

The MTCNN method uses a structure of three cascaded neural networks for: (i) detecting the faces in different stages, (ii) filtering the possible face regions, and (iii) refining the final result. It also returns a relationship between face detection and face alignment, returning fiducial points of the eyes and mouth, so proper alignment can be done after the detection.

In the first stage, a CNN P-NET is used to predict the probable face positions. After that a CNN R-NET is used in the second stage to filter the face region, removing the non-candidate faces. In the last stage, the output of the previous

network enters a CNN O-NET and outputs the face region, and the positions of the eyes, nose, and mouth. The whole pipeline of the MTCNN can be seen in Figure 3.

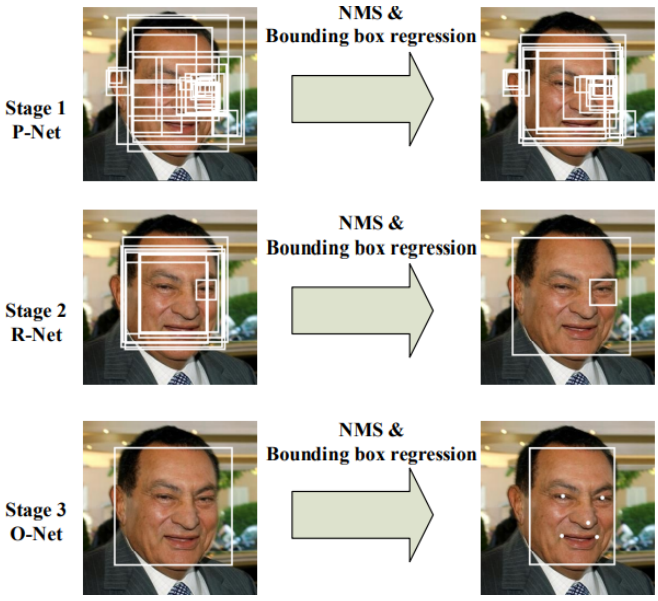


Fig. 3: Pipeline of the Multi-Task Cascaded Convolutional Neural Network face detection [12].

B. Feature Extraction

With the objective of reducing the dimensionality of the features and improve the data representation for the classification task, a Feature Extraction stage is needed, to map each face to a n-dimensional feature space. In our work, the feature extraction is done with a pre-trained convolutional neural network named VGG Face [13], whose architecture can be seen in Figure 4.

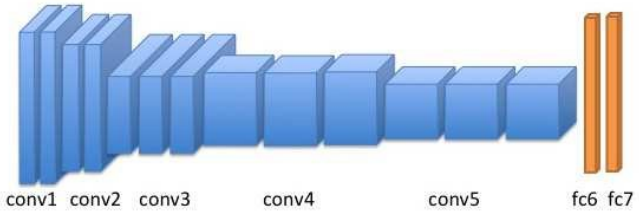


Fig. 4: VGG Face architecture [14].

VGG Face architecture is based on the VGG-16 architecture [15] and was trained in a database with 2.6 million images over a total of 2622 subjects. The input of the network consists of a $224 \times 224 \times 3$ facial image, and the output is a feature vector obtained from the fully connected layer fc7.

IV. Experiments

During all experiments, the face detection was performed using the MTCNN method, described in section III-A, after that, feature extraction was done, by inputting the face regions in the pre-trained VGG Face network described in section III-B. All the faces were normalized per channel using the standard score normalization, which can be seen in equation 3, with the mean and standard deviation values provided by the authors.

$$X_{\text{norm}} = \frac{X - \mu}{\sigma} \tag{3}$$

After feature extraction, the data was divided into different domains, according to their respective database. In all cases, the tests were carried out in the identification variation of face recognition, with three classifier instances, one K-Nearest Neighbors, with $k = 1$, a regular SVM, and the TKL method.

The parameters were also the same among all tests, with the damping factor $\zeta = 10.0$, the SVM regularization parameter was set to 1.1 and the kernel used in both the SVM and the TKL was the Gaussian kernel with $\sigma = 1.0$. All the results were compared through the accuracy metric.

A. Databases

Two databases were used for evaluation in our work, ARFace [16], and the Face Recognition Grand Challenge database [17].

1) ARFace: The ARFace [16] is a database which contains face images from 126 subjects with 26 images each, all the images are obtained in a constrained background with different contexts. In Figure 5 it is possible to see the different contexts available in the database, involving variations in facial expression, illumination, ocular region occlusion, and mouth region occlusion.

Fig. 5: Context differences in the ARFace database, involving variations in facial expression, illumination, ocular region occlusion, and mouth region occlusion [16], [18].

For the domain adaptation task, the following domains were proposed for analysis:

- **N:** Faces with neutral and other expression variations;
- **O:** Faces with occlusion in the ocular region;
- **C:** Faces with occlusion in the mouth region;
- **I:** Faces with illumination variations.

B. *Face Recognition Grand Challenge*

The Face Recognition Grand Challenge (FRGC) [17] is a database proposed to advance and develop research in face recognition.

The database contains colored facial (RGB) images collected on different seasons for three years, captured in a constrained or unconstrained setting. It also provides three dimensional face image data for 3D face recognition tasks. In Figure 6 we can see the differences between the two settings, in the first two columns we have images obtained in a constrained environment with face expression variations, while in the last three columns, we have images obtained in unconstrained environments, with differences in the background complexity and the illumination intensity.

Fig. 6: Examples of images of the FRGC database [17].

For the domain adaptation task, the constrained images were used as the source domain and the unconstrained images were used as the target domain.

V. Results

In Table I we can see the accuracy rates obtained on the three classification tasks for the ARFace database in the identification face recognition. As aforementioned, **N** refers to facial images in neutral or with expression variations, **I** refers to facial images with changes on illumination, **O** refers to facial images with occlusion in the ocular region, and **C** refers to facial images with occlusion in the mouth region. In the notation **X**→**Y**, **X** represents the source domain and **Y** represents the target domain. In our experiments, the neutral setting **N** was always used as the source domain, while the other settings were used as target domains.

Method	N→O	N→C	N→I
1-NN	67.29%	95.89%	99.62%
SVM	64.81%	94.52%	**100%**
TKL	**89.73%**	**97.71%**	99.76%

TABLE I: Accuracy rates obtained on ARFace database considering the identification task (**N** refers to faces with neutral or with expression variations, **I** refers to faces with changes on illumination, **O** refers to faces with occlusion in the ocular region, and **C** refers to faces with occlusion in the mouth region).

As we can see, the TKL method performed very well in all settings. Particularly, when classifying faces with occlusion in the ocular region, **N**→**O**, the domain adaptation provided by TKL greatly improved the classification results. Another important result that must be noted is that the feature vector obtained from the fully connected layer fc7 of VGG-Face showed to be robust when dealing with changes in illumination, given that the accuracy rates on the domain adaptation **N**→**I** were very high and they did not change that much for the three compared methods (SVM obtained 100% of accuracy rate, TKL obtained 99.76% and 1-NN obtained 99.62%).

The results obtained on the ARFace dataset also tell us about the importance of the ocular region for face recognition, since when this area is occluded, the accuracy rates drop significantly for the three assessed methods. In this case, **N**→**O**, the domain adaptation provided by TKL was of paramount importance to overcome this problem.

Regarding the experiments carried out on the FRGC dataset, Figure 7 shows the results obtained. In these experiments, the constrained images were used as the source domain, while the unconstrained images were used as the target domain.

We can see in Figure 7 that the TKL method provided, also for this more challenging dataset and difficult settings, significant gain in the accuracy rates, showing its suitability for facial recognition tasks. While TKL obtained an accuracy rate of 82.63%, the second best result, reached by SVM, was 76.35%, that is a 6.28% lower result.

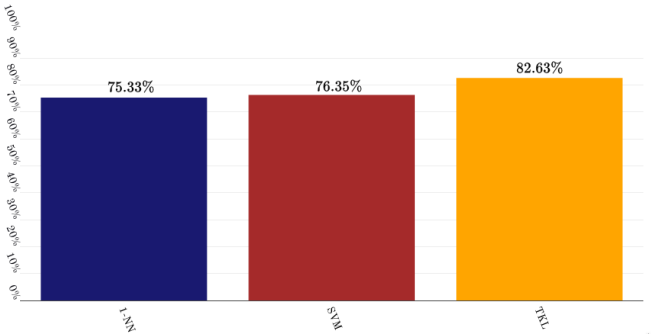


Fig. 7: Face recognition accuracy rates obtained by 1-NN, SVM and TKL methods on the FRGC database.

Since the purpose of this paper is to verify the effectiveness of domain adaptation methods in the facial identification task, this paper focused on a domain adaptation protocol for its experiments, therefore it would be unfair to compare the results with methods that follow different protocols and approach a different recognition task. That being the case, the importance of domain adaptation tasks for face recognition can be verified and even different state of the art methods could benefit from using them.

VI. Conclusions

In this work we evaluated the performance of a domain adaptation method called Transfer Kernel Learning (TKL) in the face recognition problem. Results obtained in experiments

carried out on two face datasets, ARFace and FRGC, corroborate the results found in literature that TKL is a powerful method for domain adaptation. Besides, the results showed that TKL is capable of improving the accuracy rates of face recognition, even when considering challenging scenarios, with face images presenting occlusions, variations in illumination and complex backgrounds.

VII. Acknowledgments

This paper is a result of the ongoing research of the Scientific Initiation named Robust Face Recognition Based on Domain Adaptation and has the financial support of FAPESP, process nº: 2019/15357-8.

References

[1] A. K. Jain and S. Z. Li, *Handbook of face recognition*. Springer, 2011.
[2] C. Ding and D. Tao, "A comprehensive survey on pose-invariant face recognition," *ACM Transactions on intelligent systems and technology (TIST)*, vol. 7, no. 3, p. 37, 2016.
[3] A. K. Jain, A. A. Ross, and K. Nandakumar, *Introduction to biometrics*. Springer Science & Business Media, 2011.
[4] G. Folego, M. A. Angeloni, J. A. Stuchi, A. Godoy, and A. Rocha, "Cross-domain face verification: Matching id document and self-portrait photographs," *arXiv preprint arXiv:1611.05755*, 2016.
[5] J. S. Oliveira, G. B. Souza, A. R. Rocha, F. E. Deus, and A. N. Marana, "Cross-domain deep face matching for real banking security systems," in *2020 Seventh International Conference on eDemocracy & eGovernment (ICEDEG)*. IEEE, 2020, pp. 21–28.
[6] W. M. Kouw and M. Loog, "An introduction to domain adaptation and transfer learning," *arXiv preprint arXiv:1812.11806*, 2018.
[7] G. Csurka, "Domain adaptation for visual applications: A comprehensive survey," 2017.
[8] M. Long, J. Wang, J. Sun, and S. Y. Philip, "Domain invariant transfer kernel learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 6, pp. 1519–1532, 2015.
[9] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, "Visual domain adaptation: A survey of recent advances," *IEEE signal processing magazine*, vol. 32, no. 3, pp. 53–69, 2015.
[10] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct 2010.
[11] C. K. Williams and M. Seeger, "Using the nyström method to speed up kernel machines," in *Advances in neural information processing systems*, 2001, pp. 682–688.
[12] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, Oct 2016.
[13] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proceedings of the British Machine Vision Conference (BMVC)*. BMVA Press, September 2015, pp. 41.1–41.12.
[14] M. Nakada, H. Wang, and D. Terzopoulos, "Acfr: Active face recognition using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 35–40.
[15] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[16] A. Martinez and R. Benavente, "The AR Face Database," Tech. Rep., June 1998.
[17] P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, and W. Worek, "Preliminary face recognition grand challenge results," in *7th International Conference on Automatic Face and Gesture Recognition (FGR06)*. IEEE, 2006, pp. 15–24.
[18] J. Zhou and B. Zhang, "Collaborative representation using non-negative samples for image classification," *Sensors*, vol. 19, no. 11, p. 2609, 2019.

A New Method for Gait Recognition Using 2D Poses

Daniel Ricardo dos Santos Jangua
UNESP - São Paulo State University
Bauru, Brazil
daniel.jangua@unesp.br

Aparecido Nilceu Marana
UNESP - São Paulo State University
Bauru, Brazil
nilceu.marana@unesp.br

***Abstract*—Over the last decades, biometrics has become an important way for human identification in many areas, since it can avoid frauds and increase the security of individuals in society. Nowadays, most popular biometric systems are based on fingerprint and face features. Despite the great development observed in Biometrics, an important challenge lasts, which is the automatic people identification in low-resolution videos captured in unconstrained scenarios, at a distance, in a covert and non-invasive way, with little or none subject cooperation. In these cases, gait biometrics can be the only choice. The goal of this work is to propose a new method for gait recognition using information extracted from 2D poses estimated over video sequences. For 2D pose estimation, our method uses OpenPose, an open-source robust pose estimator, capable of real-time multi-person detection and pose estimation with high accuracy and a good computational performance. In order to assess the new proposed method, we used two public gait datasets, CASIA Gait Dataset-A and CASIA Gait Dataset-B. Both datasets have videos of a number of people walking in different directions and conditions. In our new method, the classification is carried out by a 1-NN classifier. The best results were obtained by using the chi-square distance function, which obtained 95.00% of rank-1 recognition rate on CASIA Gait Dataset-A and 94.22% of rank-1 recognition rate on CASIA Gait Dataset-B, which are comparable to state-of-the-art results.**

***Index Terms*—biometric, gait recognition, pose estimation, human identification.**

I. Introduction

Over the last decades, Biometrics, that consists in the statistical study of physical or behavioral characteristics [1], has become an important tool for human identification in many areas since it can avoid frauds and increase the security of individuals in society. The most usual biometrics systems that have been deployed are based on fingerprint or face traits, which are biological characteristics, harder to imitate than the behavioral characteristic like voice and gait.

However, despite the great development observed in Biometrics, an important challenge lasts, which is the automatic people identification in low-resolution videos captured in unconstrained scenarios, at a distance, in a covert and non-invasive way, with little or none subject cooperation. In these cases, gait biometric characteristics can be the only choice.

Gait can be defined as motor behaviors composed by repetitive and integrated movements of the human body that form a pattern of corporal movements that repeat in each cycle [2]. Researches conducted in the last decades show that each individual has a special and distinct way of walking [3]. In that context, gait recognition gains relevance due to its advantages in comparison with classical biometrics methods: (i) it can be executed at a distance; (ii) it presents a good classification performance even with low-resolution images; (iii) it does not depend on subject's cooperation; (iv) occlusions do not interfere so much on performance [4].

In gait analysis there are two different approaches for characteristics representation. The first one is based on the silhouette analysis, working mostly with static aspects. The second one uses a spatio-temporal model. Despite demanding more computational effort, the model-based methods present higher reliability and better classification performance because they work with dynamic aspects of gait [2]. For a model-based approach, it is necessary a robust pose estimator that can be utilized as a part o feature extraction, capable of estimate the position of the individual's skeleton joints in video with a good reliability. For that, one can use the OpenPose [5] algorithm that utilizes Part Affinity Fields (PAFs) to learn how to associate parts with individuals that are detected within an image.

The goal of our work is to propose a new method for modeling the human gait over the frames of a video, analyzing, at each frame of the video, the angles and distances of the individual's body parts to the neck position and building a signal that represents how each body part behave during the gait cycles. For this analysis, we utilize the body parts extracted from 2D human pose estimated by OpenPose [5]. Experiments were made utilizing the public datasets CASIA Gait Dataset-A [6] composed by 20 subjects with 12 video sequences each (4 sequences for each camera position: frontal, lateral and oblique) and CASIA Gait Dataset-B [7] composed by 124 subjects walking in three different conditions (normal, wearing a coat and carrying a bag) with 11 view angles each. The results obtained showed that the proposed method is promising since they approached state-of-the-art results.

The rest of this paper is organized as follows: in Section II some related works are briefly presented. Section III discusses Human Pose Estimation, focusing on OpenPose. Section IV gives a brief introduction to gait. Section V describes the proposed approach. Section VI shows the carried out experiments in detail and Section VII draws some conclusions obtained from the results.

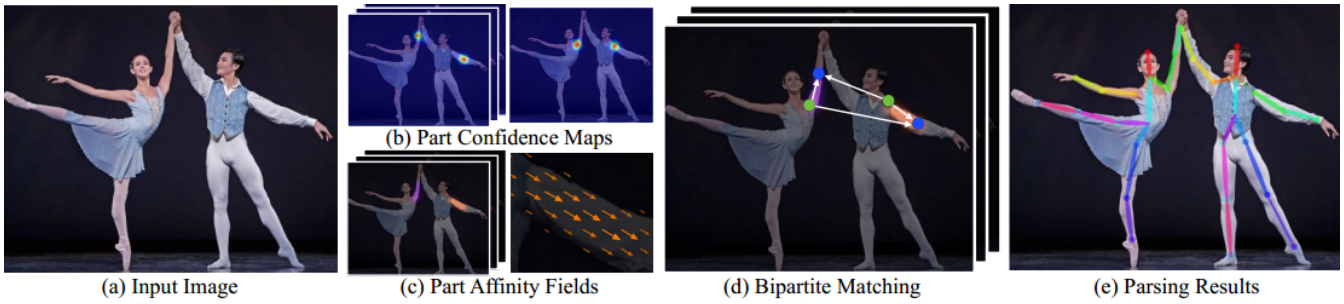


Fig. 1: Pipeline of the OpenPose method presented in [5]. The method uses the Part Confidence Maps (b) to detect the joints of human bodies in the input image and associate them using the Part Affinity Fields (c) by Bipartite Matching (d) forming poses of each individual in the image (e).

II. Related Works

In this section, some works that are related to the current proposition are briefly presented. All of them are focused on gait recognition and present results obtained on public available gait datasets.

In Wang *et al.* [8], the authors utilize a method based on silhouette analysis. The silhouette of a walking person is segmented from the video frames by a background subtraction procedure. Then, the changes of the detected silhouettes over time are represented using an associated sequence of complex vector configuration which is analyzed using the *Procrustes* shape analysis method in order to obtain a mean shape that describes all the gait sequence. On CASIA Gait Dataset-A this method achieved 90% of rank-1 accuracy in the best case and 88.75% in the worst.

In Yu *et al.* [9], a dynamic time warping (DTW) based contour similarity measure is proposed to be used in gait recognition based on silhouette analysis aiming to reduce the effect of noise on classification. On CASIA Gait Dataset-B this method achieved 83.5% of rank-1 accuracy.

In Chen *et al.* [10], the authors propose a dynamic gait representation scheme called frame difference energy image (FDEI) to work with human silhouettes even when they are incomplete. A gait cycle is divided into clusters. The FDEI of each frame is constructed using the dominant energy image (DEI) that represents a cluster. The FDEI representation can preserve the kinetic and static gait information of each frame even on incomplete silhouettes. On CASIA Gait Dataset-B the method achieved 91.1% of rank-1 accuracy.

In Liu *et al.* [11], the authors propose the use of a memory mechanism inspired by the mechanism of brain sequence processing. The 2D position of human joints are extracted using the migratory articulated human detection. Then, this information is used as input for the memory-based gait recognition (MGR) network which achieves the process of memory and identification of the gait sequence. On CASIA Gait Dataset-A this method achieved 95% of rank-1 accuracy in the best case and 85% in the worst.

In De Lima and Schwartz [12], a model-based approach is used to extract the position of the subject's joints from each video frame utilizing a pose estimation algorithm. Then, this information is transformed into signals and movement histograms to be used as feature descriptors and the subject is classified using a 1-NN classifier, with Euclidean distance. On CASIA Gait Dataset-A this method achieved 97.5% of rank-1 accuracy in the best case and 92.5% in the worst. On CASIA Gait Dataset-B this method achieved 98% of rank-1 accuracy.

III. Human Pose Estimation

Human pose estimation can be described as the detection of joint points in the human body in a given image [13]. With this information it is possible to find the human limbs by connecting the joint points and, after, calculating different features from the person's limbs movement over time when the pose estimation is applied on all frames that compose a video. In our work the OpenPose [5] method was utilized to detect 2D poses.

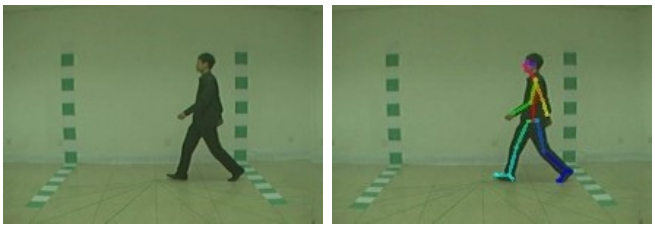
A. OpenPose

OpenPose, proposed in [5], is a real-time method for multi-person 2D pose detection on images capable of performing detection with high accuracy and good computational performance. It is the first open-source method for real-time 2D pose detection that includes body, feet, hands and face key-points. Unlike the most common approaches that detects each subject in the input image and estimate their poses individually, OpenPose takes a bottom-up approach that treats the image globally, detecting all body parts in the input image and associating them forming each individual's pose. Figure 1 presents the pipeline of the OpenPose method.

This method gains accuracy and performance using an approach named Part Affinity Fields (PAF) that maps the position and orientation of body parts present in the image domain using 2D vectors set along with the Part Confidence Maps that represent the probability of the existence of a body part in a given pixel. By using this encoded global information it is possible to adopt a greedy approach of detection and association that allows to reduce the computational complexity without losing the confidence of the results [5].

In the pose detection process, the PAFs are iteratively improved together with the confidence maps through two

interconnected convolutional neural networks (CNN), one for the PAF and other for the confidence maps. Then, the parts are associated based on the most likely matches, forming the poses [5]. Figure 2 shows an example of 2D pose estimation using OpenPose.



(a) Input image (b) Output image

Fig. 2: Example of 2D pose estimation by using the OpenPose method. (a) Frame image of a video from the CASIA Gait Dataset-B; (b) 2D pose (each colored line represents a limb part of the individual in the image).

IV. Gait Recognition

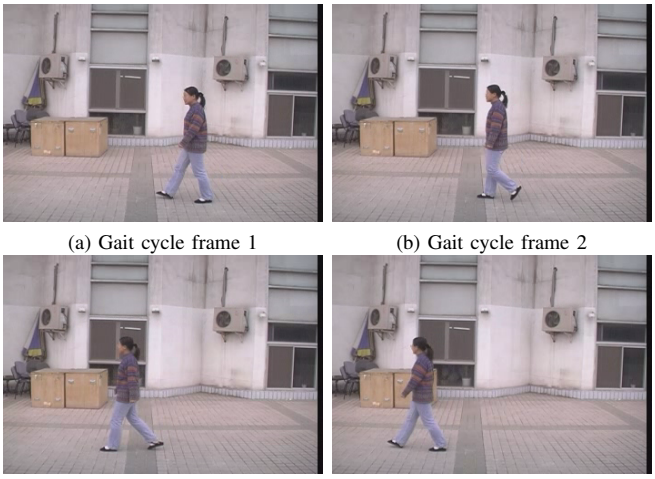
Early studies in Medicine and Psychology have shown that human gait has some components that could be used to identify an individual [4] and indicated that every human being has a unique muscular and skeletal structure, indicating that human gait recognition is feasible. According to [4], gait has some unique properties that other biometric approaches do not have: (i) it can be captured far away and at low resolution, (ii) it can be done with simple instrumentation (*e.g.* a camera or an accelerometer), (iii) it does not need the subject cooperation, (iv) it is hard to impersonate; and (v) it works well even with partial occlusion of parts of the body.

Many studies show that gait is a periodic movement that repeats a pattern into a cycle. According to [14], a gait cycle is the time interval between successive instances of initial foot-to-floor contact and each leg has two periods, a stance phase, when the foot is in contact with the floor and a swing phase when the foot is off the ground moving forward to the next step. Figure 3 shows a gait cycle resumed in four frames from a CASIA Gair Dataset-A [6] video sequence. Inside each gait cycle the superior and inferior member of the human body realize a movement similar to a pendulum, varying its angulation in relation to the horizontal (or vertical) axis forming a pattern of angle variation.

The main hypothesis of our work is based on the results obtained by the works presented in Section II, mainly in [12], which shows that with the information of how the body members behave during the gait cycle, it is possible to determine a gait signature based on the angular variation of each limb part and this signature would keep sufficient spatio-temporal information about the gait for performing biometric identification.

V. Proposed Method

This work proposes a human identification method based on gait recognition. First, OpenPose [5] is utilized to extract the



(a) Gait cycle frame 1 (b) Gait cycle frame 2

(c) Gait cycle frame 3 (d) Gait cycle frame 4

Fig. 3: Example of the gait cycle in a video sequence taken from CASIA Gait Dataset-A. The cycle begins and ends when the right heel touches the ground.

2D poses of individuals in all frames of the input video. After, for each frame, the coordinates of all joint points are utilized to calculate the angulation of each limb part in relation to the horizontal axis and the distance between the line defined by the two points representing the joints of a given limb part and the point that represents the neck. After that, these two information are utilized to build, for each limb part, over all video frames, two histograms (one for the angles and other for the distances) that are used as the gait feature vector. Finally, these feature vectors are used by a 1-NN classifier, with a predefined distance function, in order to assign the identity to the individual whose 2D poses were estimated in the input video. Figure 4 shows the block diagram of our method.

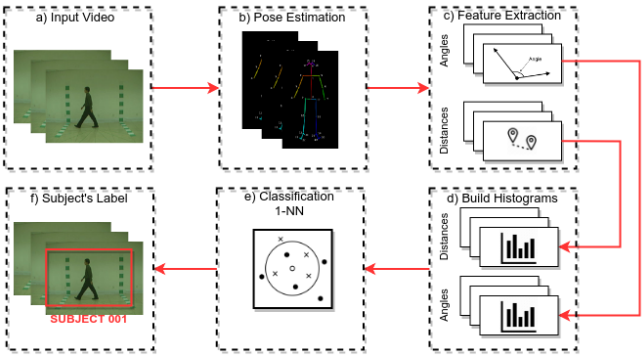


Fig. 4: Block diagram of the proposed method for gait recognition using 2D poses.

A. Pose Estimation

The pose estimation is the first step of the proposed method. In this step, the OpenPose [5] algorithm extracts, in each frame in the input video, the joints points of the person walking. The

output of this algorithm is a JSON file that contains the vertical and horizontal coordinates of each keypoint that composes the detected skeleton in each video frame. In OpenPose, it is possible to choose which skeleton type will be used in the pose estimation process. In our work, we used the BODY_25 format, which has 25 keypoints. Figure 5 shows the output of the BODY_25 format.

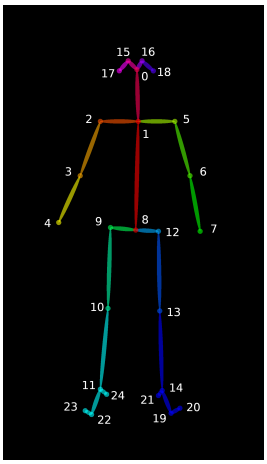


Fig. 5: Pose output format of BODY_25 [5].

B. Feature Extraction

After the pose estimation, we select the most important keypoints to be used in the feature extraction step, that is, the keypoints that encode more gait information: 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13 and 14. These keypoints define the left and right arms, the forearms, the legs and the thighs, according to the BODY_25 format (Figure 5).

For each limb part, we build a sequence formed by the angles of that limb part in relation to the horizontal axis in each frame of the video sequence. For this calculation, given two keypoints $P_1 = (x_{P1}, y_{P1})$ and $P_2 = (x_{P2}, y_{P2})$ that form a limb part, we consider the member as a 2D vector $w = (x_1, y_1)$ where $(x_1, y_1) = (x_{P1} - x_{P2}, y_{P1} - y_{P2})$ and find the angle φ between it and the vector $(x_2, y_2) = (1, 0)$ utilizing the Equation 1.

$$\varphi = \arccos \frac{x_1 * x_2 + y_1 * y_2}{\sqrt{x_1^2 + y_1^2} * \sqrt{x_2^2 + y_2^2}} \quad (1)$$

Analogously to the angle sequence, the distance sequence for each limb part is formed by the distance d between the straight line defined by the two limb part's keypoints and the keypoint that represents the neck (keypoint 1 in Figure 5), in each frame. Considering the vector $v = P_{neck} - P_2$, in which P_{neck} is the neck point and P_2 is a keypoint that forms the member in question, we can use equations 2 and 3 to do this calculation:

$$Proj_w v = \left(\frac{v \cdot w}{\|w\|^2} \right) * w \quad (2)$$

$$d = \|v - Proj_w v\| \quad (3)$$

C. Gait Histograms

With the sequences of angles and distances for each limb part in the video sequence, we build two histograms: one histogram for angles and other for distances. For both histograms, we use 16 bins, a parameter value found empirically. As our method considers eight limb parts, we have eight angle histograms and eight distance histograms, with 16 bins each.

The angle histograms are defined in the interval $[0, \pi]$, because the possible angle vary between 0 and π. The distance histograms are built applying the base 2 logarithmic function (log_2) in the distances, so the distance histograms are defined in the interval $[0, log_2(max\_dist)]$, in which max_dist is the longest calculated distance. The use of the log_2 function improves the performance of the method, as it maps the distances so that the difference between shortest distances (most recurring) is accentuated and the largest are grouped.

Finally, we concatenate all angle histograms forming one 1-dimensional angle feature vector and do the same for the distance histograms. So, at the end, our gait descriptor is composed of two histograms (angles and distances) of 128 bins each.

D. Classification

For the classification process we use a 1-NN classifier. In order to decide which distance function should be used, we assessed two distance functions, the Euclidean and the chi-square. Results of these tests are presented in Section VI.

Given a distance function, we calculate the distance d_1 between the angle histograms of the probe (query) and gallery (database) videos. Then, we calculate the distance d_2 between the distance histograms of the probe and gallery videos. The final distance between the probe and the gallery videos is $d_d = (d_1 + d_2)/2$. As both histograms, angles and distances, are normalized, there is no need to normalize the distances d_1 and d_2.

VI. Experimental Results

In order to assess the proposed new method for gait recognition, we carried out experiments on two gait datasets, CASIA Gait Dataset-A [6] and CASIA Gait Dataset-B [7].

The CASIA Gait Dataset-A, created on 2001, includes 20 subjects, each one with 12 video sequences, 4 sequences for each of the three directions: 90, 45 and 0 degrees to the camera position, that represents the lateral, oblique and frontal view of the person walking, respectively.

The CASIA Gait Dataset-B, created in 2005, has 124 individuals walking in three different conditions: normal, wearing a coat and carrying a bag. Figure 6 shows an example of these variations. For each walking sequence there are 11 view angles varying from 0 to 180 degrees.

In our first experiment, utilizing the CASIA Gait Dataset-A, we applied our method of gait recognition with the Euclidean and chi-square distance functions and compared their performances using the Cumulative Matching Characteristic (CMC) curve using the mean accuracy obtained for the three different directions (totaling 240 walking sequences, 80 for

(a) Normal walking (b) Walking in a coat (c) Carrying a bag

Fig. 6: Example of the walking condition variation on CASIA Gait Dataset-B [7] video sequences.

each direction). Figure 7 shows the CMC curves obtained in this experiment. One can observe that the chi-square distance function obtained a better performance. This result corroborates other studies that indicate that chi-square function is a good metric for histogram comparison [15].

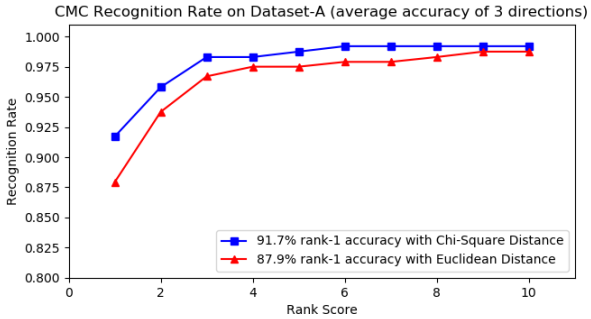


Fig. 7: CMC curves obtained by using the Euclidean and chi-square distance functions on CASIA Gait Dataset-A. Chi-square function obtained a better result.

TABLE I: Rank-1 Accuracy - CASIA Gait Dataset-A

Method	Lateral	Oblique	Frontal
Wang et al. [8]	**88.75%**	87.50%	90.00%
Liu et al. [11]	85.00%	87.50%	95.00%
De Lima and Schwartz [12]	**92.50%**	**96.25%**	**97.50%**
Our method (Euclidean)	80.00%	87.50%	**96.25%**
Our method (Chi-square)	87.50%	**92.50%**	95.00%

Table I shows the rank-1 accuracy values obtained by three methods presented in Section II ([4], [11], [12]), including two recent methods that can be considered state-of-the-art ([11] and [12]), and also by our method (using both distance functions) for each position of the camera in the CASIA Gait Dataset-A. One can observe that all methods showed better results when the person is in the front position to the camera. This should be because in this angle there is more information about the gait signature, mainly because there is no limb occlusions. One can also observe that our method, with the chi-square distance function, was superior to the method by Wang et al. [8] and was competitive with the state-of-the-art methods proposed by Liu at al. [11] and De lima and Schwartz [12].

In another set of tests, we used the CASIA Gait Dataset-B, that is significantly bigger than the CASIA Gait Dataset-A and has walking sequences that presents variation on clothing and carrying conditions. For the first test with this dataset, we utilized only the walking sequences in the lateral direction (90 degrees to the camera position) and in normal walking condition (totaling 744 walking sequences), and calculated CMC curves for our method using both distance functions. The result of this test is presented in Figure 8. It is possible to notice that, again, the chi-square distance function showed better results than Euclidean distance function.

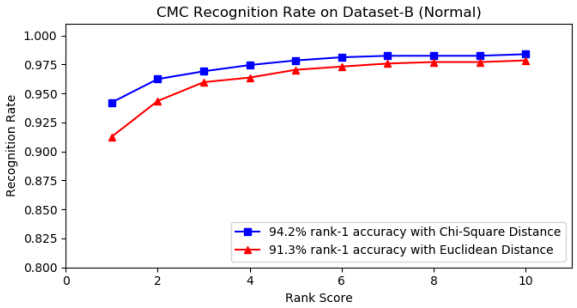


Fig. 8: CMC curve obtained by using the Euclidean and chi-square distance functions on CASIA Gait Dataset-B utilizing only the lateral view position and normal walking sequences. Chi-square function obtained a better result.

TABLE II: Rank-1 Accuracy - CASIA Gait Dataset-B (Normal)

Method	Lateral
Yu et al. [9]	83.50%
Chen et al. [10]	91.10%
De Lima and Schwartz [12]	**98.00%**
Our method (Euclidean)	91.26%
Our method (Chi-square)	**94.22%**

Table II shows the rank-1 accuracy values obtained by three methods presented in Section II ([9], [10], [12]), including one very recent method that can be considered state-of-the-art ([12]), and also by our method (using both distance functions) for the lateral position of the camera in the CASIA Gait Dataset-B. Again, although our method did not get the highest rank-1 accuracy rate, it obtained (with both distance functions) higher results than the methods proposed by Yu et al. [9] and Chen at al. [10]. In general, the tests with CASIA Gait Dataset-B showed better results considering the lateral view, this probably happens because: (i) the walking sequences in CASIA Gait Dataset-B are captured in an indoor environment and CASIA Gait Dataset-A are captured in an outdoor environment, (ii) the walking sequences in CASIA Gait Dataset-A have alternated directions in each sequence and in CASIA Gait Dataset-B all walking sequences are right-to-left, (iii) the CASIA Gait Dataset-B is significantly bigger than the CASIA Gait Dataset-A.

The second test utilizing the CASIA Gait Dataset-B was carried out with the goal of analyzing the influence of clothing

in gait recognition. We used the video sequences in which the individuals walk in the lateral direction, in normal conditions and wearing a coat. The rank-1 accuracy rates obtained by our method (with Euclidean and chi-square distance functions) and by De Lima and Schwartz's method [12] are presented in Table III. One can observe that in this case, the three results were inferior to the results presented in Table II, indicating that variations in clothing may interfere in the gait recognition.

TABLE III: Rank-1 Accuracy - CASIA Gait Dataset-B (Normal+Wearing a Coat)

Method	Lateral
De Lima and Schwartz [12]	**95.16%**
Our method (Euclidean)	86.29%
Our method (Chi-square)	**89.72%**

We observe that our method share some ideas with the method proposed by De Lima and Schwartz [12], that obtained the best results in all carried out experiments. Both methods utilize 2D poses and histograms as gait descriptors, however in the best results the method by De Lima and Schwartz [12] utilizes two histograms for each keypoint of the detected skeleton (one histogram for the horizontal coordinate and other for the vertical coordinate) totaling 24 histograms with 85 bins each (that results in a 2040-dimensional feature vector), while our method utilizes two histograms for each limb part totaling 16 histograms with 16 bins each (that results in two 128-dimensional feature vector - one for distances and other for angles). As the number of limb parts are lower, our method leads to a significant reduction in the dimensionality of the feature vectors and, consequently, improves the computational performance, while keeping comparable accuracy rates.

VII. Conclusions and future work

The results obtained by our method are preliminary and still have room for improvements. They indicate that the angular variation of the limbs in gait sequence combined with the distance to the neck point can encode sufficient information about the gait signature to obtain good results in gait recognition. The main advantage of our method is that compared with the method proposed by De Lima and Schwartz [12], for example, it presents a better computational performance because it defines a more compact gait signature information.

In all conducted experiments that confronted the Euclidean and chi-square distance functions, we could observe that the use of chi-square distance function improved the method's accuracy. This probably happens because chi-square distance function seems to suit better for histogram comparisons. As future work, we intend to assess other distance functions, such as Bhattacharyya and Intersections [15], since the choice of a good distance functions matters. We also intend to investigate the best weights to be used for calculating the average distance between the angle and distance histograms.

From our experimental results we can also infer the importance of the pose estimation step to the method's performance, since we use its output coordinates to map the individual's movements during the gait cycle. A pose estimator with less detection errors is of paramount importance for the robustness of the method. For future work, we intend to assess other algorithms for 2D pose estimations, such as PifPaf [16], and other skeleton formats.

For future work, we also intend to focus on improvements of the pose estimation step, mainly in error handling and noise attenuation that, according to the tests, seems to have the higher impact in the gait recognition performance, mostly when there are variations in clothing conditions, for instance.

VIII. Acknowledgments

This article is a result of the research of the scientific initiation conducted with the support of the Institutional Program for Scientific and Technological Initiation Scholarships (PIBIC) sponsored by National Council for Scientific and Technological Development (CNPq).

References

[1] A. K. Jain, A. A. Ross, and K. Nandakumar, *Introduction to biometrics*. Springer Science & Business Media, 2011.

[2] M. Arantes and A. Gonzaga, "Human gait recognition using extraction and fusion of global motion features," *Multimedia Tools and Applications*, pp. 655–675, 2011. [Online]. Available: https://doi.org/10.1007/s11042-010-0587-y

[3] M. S. Nixon and J. N. Carter, "Automatic recognition by gait," *Proceedings of the IEEE*, vol. 94, no. 11, pp. 2013–2024, 2006.

[4] C. Wan, W. Li, and V. V. Phoha, "A survey on gait recognition," *ACM Digital Library*, 2018. [Online]. Available: https://dl.acm.org/doi/10.1145/3230633

[5] Z. Cao, G. Hidalgo, T. Simon, S. Wei, and Y. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *CoRR*, vol. abs/1812.08008, 2018. [Online]. Available: http://arxiv.org/abs/1812.08008

[6] Liang Wang, Tieniu Tan, Huazhong Ning, and Weiming Hu, "Silhoutte analysis based gait recognition for human identification," *IEEE trans Pattern Analysis and Machine Intelligence(PAMI)*, vol. 25, no. 12, pp. 1505–1518, 2003.

[7] Shiqi Yu, Daoliang Tan, and Tieniu Tan, "A framework for evaluating the effect of view angle, clothing and carrying condition on gait recognition," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 4, Aug 2006, pp. 441–444.

[8] Wang L., Tan T., Hu W., and Ning H., "Automatic gait recognition based on statistical shape analysis," *IEEE transactions on pattern analysis and machine intelligence*, vol. 12, no. 9, pp. 1120–1131, 2003.

[9] Yu S., Tan D., Huang K., and Tan T., "Reducing the effect of noise on human contour in gait recognition," *Internat. Conf. on Biometrics*, 2007.

[10] Chen C., Liang J., Zhao H., Hu H., and Tian J., "Frame difference energy image for gait recognition with incomplete silhouettes," *Pattern Recognition Letters*, 2009.

[11] Liu D., Ye M., Li X., Zhang F., and Lin L., "Memory-based gait recognition," *BMVC*, 2016.

[12] V. C. de Lima and R. Schwartz, "Gait recognition using pose estimation and signal processing," *Iberoamerican on Pattern Recognition - CIARP*, 2019.

[13] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 1653–1660.

[14] Mark S. Nixon, Tieniu N. Tan, and Rama Chellappa, *Human Identification Based on Gait*. Springer Science & Business Media, 2006.

[15] P. A. Marín-Reyes, J. Lorenzo-Navarro, and M. Castrillón-Santana, "Comparative study of histogram distance measures for re-identification," *arXiv preprint arXiv:1611.08134*, 2016.

[16] S. Kreiss, L. Bertoni, and A. Alahi, "Pifpaf: Composite fields for human pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 977–11 986.

Using CNNs for Quality Assessment of No-Reference and Full-Reference Compressed-Video Frames

1st Renato R. da Silva
School of Computer Science
Federal University of Uberlândia
Uberlândia, Brazil
renato.rsufu@gmail.com

2nd Luiz F. A. Brito
School of Computer Science
Federal University of Uberlândia
Uberlândia, Brazil
luiz.brito@ufu.br

3rd Marcelo K. Albertini
School of Computer Science
Federal University of Uberlândia
Uberlândia, Brazil
albertini@ufu.br

4th Marcelo Z. do Nascimento
School of Computer Science
Federal University of Uberlândia
Uberlândia, Brazil
marcelo.nascimento@ufu.br

5th André R. Backes
School of Computer Science
Federal University of Uberlândia
Uberlândia, Brazil
arbackes@yahoo.com.br

***Abstract*—For videos to be streamed, they have to be coded and sent to users as signals that are decoded back to be reproduced. This coding-decoding process may result in distortion that can bring differences in the quality perception of the content, consequently, influencing user experience. The approach proposed by Bosse et al. [1] suggests an Image Quality Assessment (IQA) method using an automated process. They use image datasets pre-labeled with quality scores to perform a Convolutional Neural Network (CNN) training. Then, based on the CNN models, they are able to perform predictions of image quality using both Full-Reference (FR) and No-Reference (NR) evaluation. In this paper, we explore these methods exposing the CNN quality prediction to images extracted from actual videos. Various quality compression levels were applied to them as well as two different video codecs. We also evaluated how their models perform while predicting human visual perception of quality in scenarios where there is no human pre-evaluation, observing its behavior along with metrics such as SSIM and PSNR. We observe that FR model is able to better infer human perception of quality for compressed videos. Differently, NR model does not show the same behaviour for most of the evaluated videos.**

***Index Terms*—Convolutional Neural Network, Digital Video Streaming, Quality Analysis.**

I. Introduction

The consume of digital contents is increasingly becoming part of our everyday lives, especially regarding digital video. Video streaming is widely transmitted nowadays not only for high definition television, but also for video chats, conferences and internet streaming [1].

For video contents to be transmitted to users, some important processes must be done. First, the video content has to be coded and transformed in signals that will be sent as packages to the final user [2]. Next, this signals are decoded and remounted as video content to be reproduced on the user side. The problem is that this coding-decoding process may result in distortions which may lead to differences in the quality perception. Therefore, the received video, when reproduced to the audience, may not show the exact quality as the original source file [2].

Depending on the context of video reproduction, quality assessment can be a crucial aspect. Usually, the quality of a video is related to the quality of the images or frames that compose it. The human perception of the quality of a visual content can be hard to quantify as it is a subjective matter and may vary from person to person. Thus, being able to assess quality is an important task but, definitely not a trivial one [3]. One of the ways to perform Image Quality Assessment (IQA) is to make a classification depending on the amount of information from the original reference image present in the distorted one [1]. When the access to the full reference image is available, the IQA approach is called Full-Reference (FR), and, when it is not available, the IQA is called No-Reference (NR) approach [1].

Some state-of-the-art techniques evaluate quality of images and videos purely based on human opinion. Basically, various samples of images with different levels and types of compression are shown to human subjects that, based on their visual perception, classify the samples with scores of quality [4].

Recently, other approaches have been proposed to perform IQA using automated methods. For example, in the work of Bosse et al. [1], the authors use datasets of images already classified according to their quality to train a Convolutional Neural Network (CNN). Classification was performed with traditional human review and used as training and validation samples to the CNN. Then, the CNN models were used to perform predictions of Image Quality Assessment.

Convolutional neural networks, in recent years, have shown great relevance among the traditional approaches related to

computer vision. This technique has been widely used due to the quality and amount of data available, and the computing power that has been growing significantly through the years. Furthermore, CNNs allow researchers to provide joint learning of resources and regression based on raw input data with very little manual interference needed [5].

These networks receive labeled samples as inputs. As these samples pass through the network layers by the epochs, features are extracted and the network learns, more generally, which features best represent each label [6]. Different types of layers can be used to build the network structure. Some of the most commonly used are the convolutional layer, the pooling layer and the fully connected layer. The convolutional layer is responsible for applying convolutions using activation filter masks responsible for extracting the features of the image samples. The use of this type of layer is the reason for the name "convolutional neural network". The filters are initially defined in a random way and have their values adjusted gradually at each iteration of the samples in the neural network [6]. The pooling layer is responsible for receiving samples and, based on some parameters, producing smaller samples which occupy less disk space. This fact is important since neural networks usually demand a large amount of input samples. Besides this advantage, this layer is intended to generate more robust features by reducing the sensitivity of the network to distortions. This way, a greater variety of images can be associated with the generated features, thus enhancing the classification [6]. Finally, the fully connected layer is responsible for performing regression and weight adjustments. The samples used as inputs to the neural network are initially divided into training and validation sets. Then, the validation set is compared with the training set in order to identify necessary weight adjustments for next iterations [6].

In their work, Bosse et al. [1] use TID2013 [7] and LIVE [8] datasets of images already classified according to their quality. The quality labels previously defined by human subjects are used as classes to train a CNN using 3000 epochs, 10 convolutional layers, 5 pooling layers, as well as 2 fully connected layers for regression. After the training process, the CNN models are used to perform predictions of Image Quality Assessment. Also, it is worth mentioning that, although the aim of their work was to propose methods for assessment of image quality in video streaming, all images used in the training process were single pictures and not extractions of compressed videos. Thus, we can consider that compression methods covered by the training process only exploit spatial redundancies to reduce the size of pictures. Another approach could also exploit temporal redundancies when considering a sequence of pictures as frames that compose the video. Besides, the tests provided by their original article only states the use of images as tests to the CNN, in order to obtain quality prediction and compare that result to the quality evaluation provided by the human subjects in the dataset.

In this paper, we explore the methods created by Bosse et al. [1] exposing the CNN quality prediction to frames extracted from real compressed videos. Two different video codecs using various quality compression levels were applied to them. We also evaluate how their methods perform while predicting human visual perception of quality in scenarios where there is no human pre-evaluation, observing its behaviour along with metrics such as SSIM and PSNR.

The remainder of this paper is organised as follows. In Section II we describe the process of acquisition, compression and extraction of frames used in the test process, as well as the basic application of these test samples in the CNN using models previously trained. Section III describes the experiments and we report the results achieved by this work. Conclusions and future work are presented in Section IV.

II. Methods

In this section we introduce our methods used to investigate whether the CNN models proposed in [1] infer correctly the perceptual quality of actual compressed videos. First, in Section II-A we present the raw videos used for compression. Then, in Sections II-B and II-C we describe the steps to compress and extract frames from the raw videos. Finally, in Section II-D we detail the process of evaluation using the inference models. A flow chart of all the steps can be observed in Figure 1. Also, the overall configurations of our methods for this paper is presented at Table I.

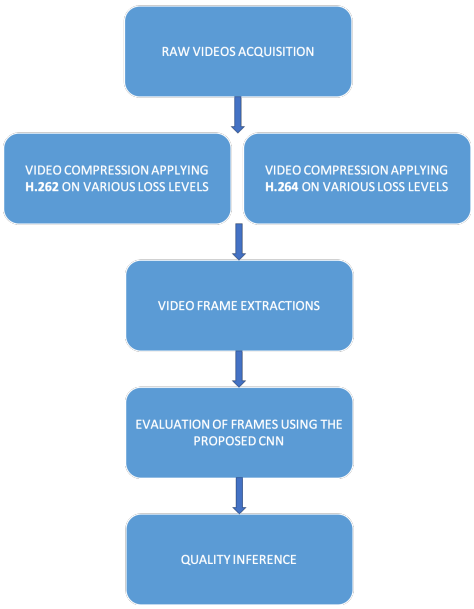


Fig. 1. Flow chart showing the main steps of this project.

A. Video Acquisition

First, we collected videos stored in raw formats that represent diverse scenarios. This diversity is important because different characteristics can introduce different challenges for video codecs and IQA models. For this paper, we obtained four 720p videos with 50 frames per second and no chroma sub sampling. We chose these videos because their quality and sub sampling level allowed us to do a more detailed evaluation.

As they have high quality we could explore more freely more levels of compressions. Also, these videos have no copyright restrictions and are available on https://media.xiph.org/video/derf/ [9].

In Figure 2, we present the preview for the four videos we used to compress and evaluate the CNN models proposed in [1]. Figure 2a, `ducks`, shows a frame extracted from the corresponding video that contains ducks swimming in a river. This scene has slow movements and the colours have low frequencies. The most challenging part for codecs to handle is the wave movements created by the ducks. Figure 2b, `house`, shows the landscape of a house surrounded by vegetation. This scene has elements with low frequency – the house – and high frequency – the vegetation. Trees have borders with irregular shapes, which will probably affect negatively the compression in these regions. Figure 2c, `park`, shows people running with distinct clothing in a park. The main characteristic of this scene is the fast movement of the objects. While camera follows people, trees appear and disappear in the foreground and background, creating a not straightforward time dependency among objects. Finally, Figure 2d, `town`, shows the aerial view of a town. This scene is very detailed and presents very high frequencies, a difficult scenario for video codecs.

B. Video Compression

We compressed the raw videos collected in the previous step using two video codecs provided by the FFmpeg software [10], namely, h.264 [11] and h.265 [12]. These two video codecs reduce the size of raw videos by exploiting spatial and temporal redundancies. First, they convert the image colour space to YCbCr and apply chroma sub sampling to reduce the size of each frame by half without perceptual degradation. This is due to our vision system that do not distinguish subtle changes of colours. Then, they apply prediction techniques to infer whole blocks of pixels by using data of other blocks previously processed. Some techniques predict blocks using only data contained in the same frame, this is called spatial prediction [13]. Frames, such as those containing blue skies, can have well defined patterns and algorithms can use knowledge gathered previously to predict next blocks. Spatial prediction techniques are also employed by image codecs such as JPEG [13] and JPEG2000 [14], which are used in the work of Bosse et al. [1]. Other techniques use data from other frames in order to predict next blocks, this is called temporal prediction [12]. For example, in a movie that contains a ball bouncing on the floor, blocks in the next frame can be predicted by observing the displacement of objects compared to their location in previous frames. Therefore, compression of videos can have different results when compared with compression of single images due to the addition of temporal prediction techniques.

Often, h.264 and h.265 codecs are used for lossy compression but can also be used for lossless compression. After the prediction step, these codecs use the Discrete Cosine Transform (DCT) to obtain coefficients in frequency domain and, then, they apply a quantization matrix to reduce data. The factor of this quantization matrix controls the compression behaviour. If the quantization factor is zero, then, all prediction errors are stored without reduction and, at decoding phase, they are used to reconstruct the video with no loss. If the quantization factor is greater than zero, then, it is a lossy compression and, the higher the quantization factor is, the smaller will be the size and the overall quality of the resulting video. Therefore, lossy compression increases the pixel-to-pixel error and can decrease perceptual quality when the quantization factor is high.

The FFmpeg software provides implementation of many video codecs and can control the bit rate of compressed videos using input parameters. In this paper, we chose h.264 and h.265 codecs because FFmpeg has a uniform parameter, the Constant Rate Factor `CRF`, that controls the compression level of these codecs. The `CRF` parameter for h.264 and h.265 varies from 0 to 51, where 0 means the compression is lossless and 51 means the compression has the highest loss. The default value for `CRF` is 23 and the documentation says that, in order to keep visually lossless quality, one should use `CRF` values near 17 or 18. In this paper, we vary `CRF` from 1 to 51 using h.264 and h.265 for every video described in Section II-A. Below is an example of the command line we used to compress the video `ducks.y4m` to `ducks_h264_1.mp4` using the codec h.264 with `CRF` equals to 1.

```
$ ffmpeg -i ducks.y4m \
  -vcodec libx264 -crf 1 ducks_h264_1.mp4
```

C. Frame Extraction

We used the FFmpeg software to extract 10 frames from each compressed video. First, we queried the duration of the videos using the `ffprobe` command, a program included in the FFmpeg installation. Below is an example of the command line we used to query the duration of the video `ducks_h264_1.mp4`.

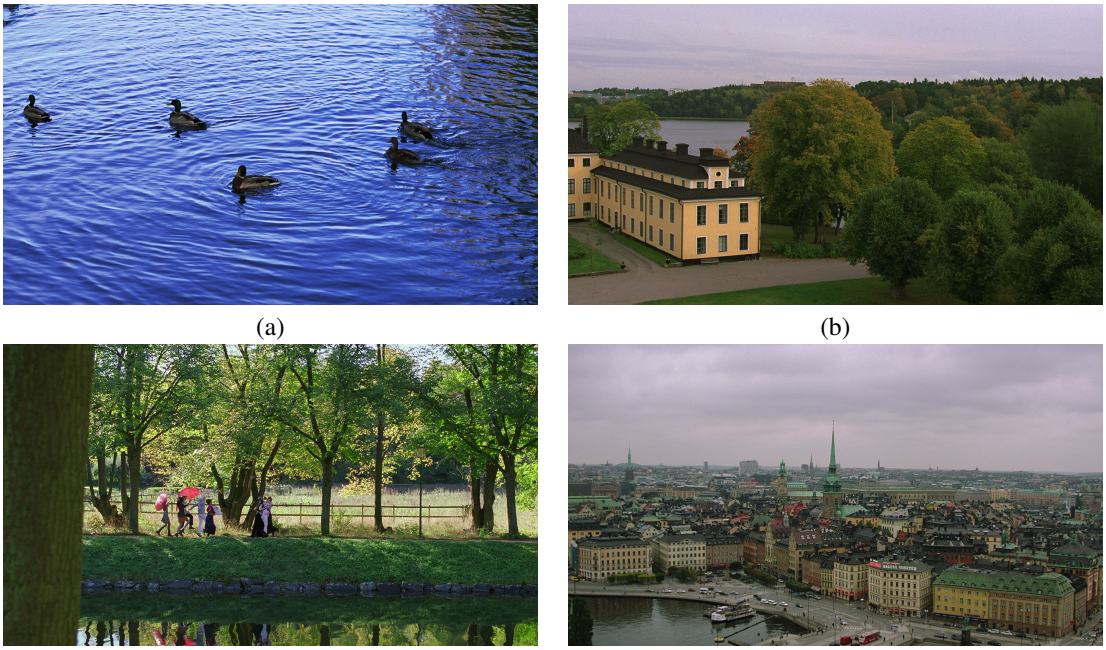
```
$ duration=$(ffprobe -i ducks_h264_1.mp4 \
    -show_entries format=duration \
    -v quiet -of csv="p=0")
```

Then, we generated 10 time stamps at random ranging from 0 to the duration of each video. Finally, we extracted the next frame after the generated time stamps in each compressed video. Note that the time stamps generated for a particular reference video were used to extract frames from all corresponding compressed videos. Below is an example of the command line we used to extract the first frame after the second 5 of the compressed video `ducks_h264_1.mp4` as the image `ducks_h264_1_5.bmp`.

```
$ ffmpeg -i ducks_h264_1.mp4 \
    -ss 5 -vframes 1 ducks_h264_1_5.bmp
```

D. Perceptual Quality Inference

After extraction, we evaluated the compressed video frames using the CNN models proposed by Bosse et al. in [1]. In their



(a) (b) (c) (d)

Fig. 2. Preview of the videos obtained for compression, frame extraction, and model evaluation.

work, the authors built models from two datasets, TID2013 [7] and LIVE [8], using two different approaches, FR and NR, in which they compare CNNs with two different pooling layers applying standard mean or weighted mean. In this paper, we will explore only the FR and NR models built from TID2013 dataset using weighted mean variant of pooling layer.

The original authors have made their code and models available on https://github.com/dmaniry/deepIQA. In order to evaluate a compressed frame using the FR approach, the reference frame needs to be passed to their program. Differently, for NR approach, only the compressed frame needs to be passed as input. The output of all executions were stored in a Comma Separated Values (CSV) file to run the analysis described in Section III. Below is an example of the command line we used to execute the CNN model `fr_tid_weighted.model` to predict the perceptual quality of the compressed frame `ducks_h264_1_5.bmp` using the reference `ducks_reference_5.bmp` for the FR approach.

```
result=$(python evaluate.py \
--model fr_tid_weighted.model \
--top weighted \
ducks_h264_1_5.bmp ducks_reference_5.bmp)
```

Additionally, we also computed quality measurements of the compressed videos using the ImageMagick software [15]. This software is often used to automate image edition but it also provides the ability to compute quality measurements of compressed images given the reference picture. In this paper, we computed Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) for every compressed frame using the corresponding frame of the reference video. Below is an example of the command line we used to compute the PSNR of the compressed frame `ducks_h264_1_5.bmp` regarding its reference frame `ducks_reference_5.bmp`. The residuals are stored in the output image `residuals.png` and can be ignored.

TABLE I
EVALUATED PARAMETERS

Data	Value
Compression Codecs	H.264 and H.265
Quality Loss levels (CRF)	0, 1, 6, 11, 16, 21, 26, 31, 36, 41, 46, 51
Assessment approaches	FR and NR
Videos	4
Frames Extracted per Video	500
CNN Training model	TID2013 weighted
Total of Frame Samples	2000

```
psnr=$(magick compare -metric psnr \
ducks_h264_1_5.bmp ducks_reference_5.bmp \
residuals.png 2>&1)
```

All scripts for data acquisition, video compression, frame extraction and evaluation are public available in our repository on https://bitbucket.org/luizcoro/seminario-multimidia-2019/.

III. RESULTS

In this section, we present and discuss our results. We gathered outputs of the CNN models proposed by Bosse et al. in [1] along with the quality measures PSNR and SSIM. PSNR is based on pixel-to-pixel error of the compressed frames and SSIM is a measure that treats structural components differently in order to achieve a quality closer to our visual perception.

Therefore, the results of the CNN models should present more similarities with SSIM than PSNR.

Instead of presenting the results for each extracted frame, we opted to present in terms of mean and standard deviation. This aggregation also generalize the experiments and improve the readability of graphs. Additionally, we inverted the output v from the CNN models, to analyse quality level instead of quality loss level, also to be able to compare them with the other measures in this same behavior. It was accomplished by using the function $v_{inv} = 100 - v$. The 100 element in the function represents the highest possible output value v from the model and originally meant the highest quality loss, while 0 was the lowest quality loss. After the inversion, at the graphs, 0 represents the lowest quality level, while 100, the highest quality level.

In Figure 3, we present the results of the models FR and NR, and the measures PSNR and SSIM, varying values of `CRF`. We noted that the FR model describes more accurately the perceptual quality of the compressed videos than PSNR. In our experiments, PSNR had approximately a constant decreasing behaviour for all videos as the `CRF` increased and, therefore, does not corresponds well to our visual perception. Differently, the FR model presents a non-linear descending curve showing that the perceptual quality of compressed videos does not decrease at the same rate. This behaviour seems more natural to our visual perception since small degradations sometimes are not captured by our visual system. Furthermore, when the video begins to present distortions, as `CRF` values increases, our visual system begins to perceive the decreasing of quality more clearly, which agrees with the FR model.

Its important to notice that the FR value presents a larger standard deviation as `CRF` increases, especially for `park` and `house` videos. These videos present more details and movement, which affect negatively the compression. Differently, PNSR presents larger standard deviation for smaller values of `CRF`, thus corroborating its inability to correctly quantify the perceptual quality of the video.

Another important aspect is that the results of FR model have more similarities with SSIM than PSNR, even though they do not agree precisely. This is expected as the SSIM measure captures more characteristics of our visual system than PSNR does.

In contrast, the NR model did not exhibit an accurate description of our visual perception in most videos. This is due to the absence of information about reference frames, which compromises the ability of the method to infer the perceptual quality. For example, in the video `ducks`, the NR model obtained very low fluctuation as the `CRF` value increased. We believe that this behaviour happened because the frames were very similar. As we compress the video, it is expected its perceptual quality to diminish. Yet, it can be observed that the curves representing `park` and `town` videos, in Figure 3b, showed an unexpected oscillation (as also a large standard deviation), with exception of the video `town` compressed with h.265. However, in `house`, the NR model was very close to the FR model and described the perceptual quality of the compressed video more accurately. Such result may be due to the presence of reference in the training process. Therefore, there are cases in which the NR model can be used to infer the human perception of quality in compressed videos. Further investigation is needed to point the cases that the usage of NR model is appropriate.

According to FFmpeg documentation, `CRF` values around 17-18 were expected to generate compression without quality loss perceivable by our visual system. However, as our results show, this threshold appears to be around 25-26 when using the FR model, for the videos presented in this paper. Therefore, more aggressive compressions can be used, saving space and, consequently, improving transmission.

In this paper, we do not show the sequences of videos to compare with the results. We suggest running the scripts publicly available in our repository on https://bitbucket.org/luizcoro/seminario-multimidia-2019/ to have access and reproduce the compressed videos.

IV. Conclusions

In this work we evaluated the results of the methods created by Bosse et al. in [1], using various frames as test samples extracted from several compressed videos. For this work, we used four raw videos. We generated different quality levels of compression for each video. We also utilized h.264 and h.265 codec compression in order to explore the effects of the loss levels in the result of the automatic evaluation.

In terms of NR assessment of images, it can be noticed that the results is sometimes equivocated, as the methods suggest that visual perception alternates sometimes between lower and higher values, even though the quality in our tests only decreases. Also, as shown in Figure 3b, the algorithm predicted a kind of uniform level of quality despite the constant decreasing of compression quality. We believe that it is due to the fact that that video has very similar frames.

In contrast, results also demonstrated that the proposed methods of Image Quality Assessment using Deep CNN has a great effectiveness in most cases when using FR approach. Despite the CNN models have only been trained with single pictures, exploiting only spatial redundancies, FR method was able to infer perceptual quality on compressed video frames. Indicating that the approach covered in this paper can be considered a feasible solution for IQA of video frames specially in FR approach. As future work, we propose to investigate further the cases in which the usage of NR model is appropriate.

Acknowledgment

André R. Backes gratefully acknowledges the financial support of CNPq (National Council for Scientific and Technological Development, Brazil) (Grant #301715/2018-1). Marcelo Z. do Nascimento gratefully acknowledges the financial support of CNPq (National Council for Scientific and Technological Development, Brazil) (Grant #304848/2018-2). Luiz F. A. Brito gratefully acknowledges the financial support of FAPEMIG (Foundation to the Support of Research in Minas

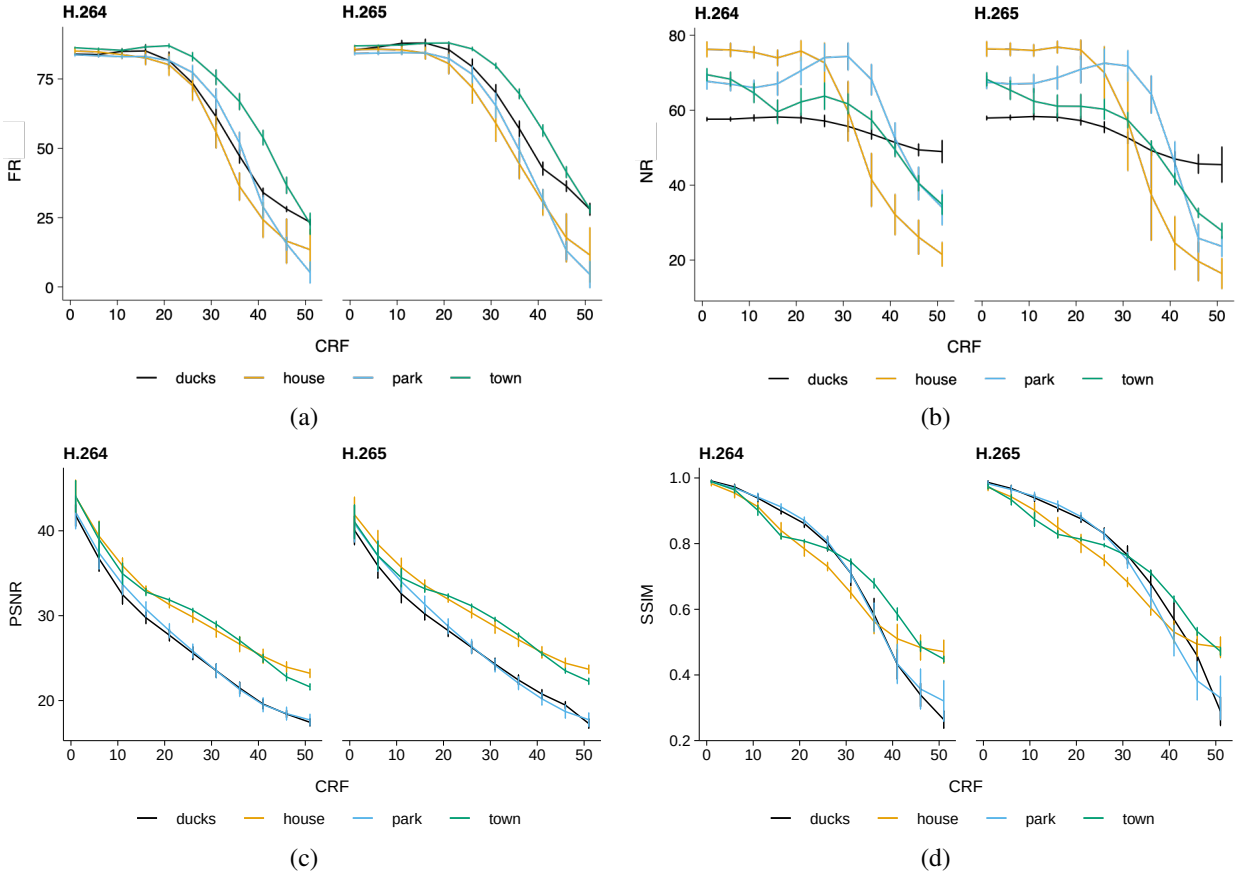


Fig. 3. Results for each video using h.264 and h.265 video codecs. In x axis, we vary the values of `CRF` while, in y axis, we present the results for the FR model (a), NR model (b), PSNR(c) and SSIM (d). Each line describes the different videos that we previously presented in Figure 2.

Gerais). This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) - Finance Code 001.

REFERENCES

[1] S. Bosse, D. Maniry, K.-R. Müller, T. Wiegand, and W. Samek, "Deep neural networks for no-reference and full-reference image quality assessment," *IEEE Trans. Image Processing*, vol. 27, no. 1, pp. 206–219, 2018.

[2] Z.-N. Li, M. S. Drew, and J. Liu, *Fundamentals of Multimedia*, ser. Texts in Computer Science. Springer, 2014.

[3] Z. Wang, A. C. Bovik, and L. Lu, "Why is image quality assessment so difficult?" in *ICASSP*. IEEE, 2002, pp. 3313–3316. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7874

[4] T. K. Tan, R. Weerakkody, M. Mrak, N. Ramzan, V. Baroncini, J.-R. Ohm, and G. J. Sullivan, "Video quality evaluation methodology and verification testing of hevc compression performance," *IEEE Trans. Circuits Syst. Video Techn*, vol. 26, no. 1, pp. 76–90, 2016.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Apr. 10 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[6] L. Kang, P. Ye, Y. Li, and D. S. Doermann, "Convolutional neural networks for no-reference image quality assessment," in *CVPR*. IEEE Computer Society, 2014, pp. 1733–1740. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6909096

[7] N. Ponomarenko, L. Jin, O. Ieremeiev, V. Lukin, K. Egiazarian, J. Astola, B. Vozel, K. Chehdi, M. Carli, F. Battisti, and C.-C. J. Kuo, "Image database tid2013: Peculiarities, results and perspectives," *Signal Processing: Image Communication*, vol. 30, pp. 57 – 77, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0923596514001490

[8] H. R. Sheikh, M. F. Sabir, and A. C. Bovik, "A statistical evaluation of recent full reference image quality assessment algorithms," *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3440–3451, Nov 2006.

[9] Xiph.Org Foundation, "Non-profit corporation dedicated to protecting the foundations of internet multimedia from control by private interests," https://www.xiph.org/, 1994–2019.

[10] FFmpeg Software, "Complete, cross-platform solution to record, convert and stream audio and video," https://ffmpeg.org/, 2000–2019.

[11] A. Mittal, A. K. Moorthy, and A. C. Bovik, "Visually Lossless H.264 Compression of Natural Videos," *The Computer Journal*, vol. 56, no. 5, pp. 617–627, 07 2012. [Online]. Available: https://doi.org/10.1093/comjnl/bxs105

[12] V. Sze and M. Budagavi, "High throughput cabac entropy coding in hevc," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1778–1791, Dec 2012.

[13] W. B. Pennebaker and J. L. Mitchell, *JPEG: Still image data compression standard*. Springer Science & Business Media, 1992.

[14] D. Taubman and M. Marcellin, *JPEG2000 image compression fundamentals, standards and practice: image compression fundamentals, standards and practice*. Springer Science & Business Media, 2012, vol. 642.

[15] ImageMagick Software, "Free software delivered as a ready-to-run binary distribution or as source code that you may use, copy, modify, and distribute in both open and proprietary applications," https://imagemagick.org/index.php, 1987–2019.

A Thorough Evaluation of Kernel Order in CNN Based Traffic Signs Recognition

Lucas Armand Souza Assis de Oliveira[1], Guilherme Lucio Abelha Mota[1], Vitor da Silva Vidal[1]
[1]Post-Graduation Program in Computational Sciences, Rio de Janeiro State University, Brazil

***Abstract*—Convolutional Neural Network is an important deep learning architecture for computer vision. Alongside with its variations, it brought image analysis applications to a new performance level. However, despite its undoubted quality, the evaluation of the performance presented in the literature is mostly restricted to accuracy measurements. So, considering the stochastic characteristics of neural networks training and the impact of the architectures configuration, research is still needed to affirm if such architectures reached the optimal configuration for their focused problems. Statistical significance is a powerful tool for a more accurate experimental evaluation of stochastic processes. This paper is dedicated to perform a thorough evaluation of kernel order influence over convolutional neural networks in the context of traffic signs recognition. Experiments for distinct kernels sizes were performed using the most well accepted database, the so-called German Traffic Sign Recognition Benchmark.**

***Keywords*—autonomous vehicles, CNNs, kernel size, statistical evaluation**

I. Introduction

Autonomous vehicles is a major trend that will change the paradigm of goods and people transport [1]. Currently, one of the main technological challenges related to autonomous vehicles is the correct perception of external environment [2]. Computer vision is a powerful tool that allows the autonomous system to "understand" the world around. One of the most popular machine learning techniques is Neural Networks [3]. Recently, deep networks have experimented a fast evolvolution and specialization in complex problem-specific designs. In the case of image classification, a particularly effective architecture is the Convolutional Neural Network (CNN) [4].

A problem of particular interest for the autonomous vehicles application in the area of computer vision is classifying traffic signs. We reviewed a wide bibliography on this topic (section II), and found a variety of solutions and architectures proposed for this application. However, very few papers seek to present general conclusions about the best configuration of the proposed neural networks architectures (e.g. amount of filters, number of layers and filter order) for the specific application. Even those who focus on doing multiple tests varying particular architecture have little or no statistical analysis to substantiate the results. For instance, [5] compares the performance of five distinct architectures for traffic signs detection without tuning networks configuration. Authors, indeed, combine any features extractors with distinct base architectures, nevertheless, all components are treated as closed modules without revealing the examination of such "black-boxes". Another example is [6], which employs genetic algorithms just in the context of obtaining the optimal learning ratio and number of epochs to be used for training a CNN.

One exception is [7], that analyses the outcomes of distinct kernel sizes in a CNN model in the context of traffic sign recognition. However, little or no discussion is made about the character intrinsically stochastic of the learning process and the randomness of the neural network parameters initialization. As usual, comparisons are restricted to the model's accuracy and lack the statistical significance evaluation of results.

This work proposed a review of Sichkar and Kolyubin paper [7] in the direction of finding the best kernel size in a traffic signs recognition application using a CNN architecture. It aims at validating the results presented by the original paper, while proposing a more in-depth statistical analysis underlain by a broader set of experimental results. The objective of this work is not to advance the state of the art in terms of classification accuracy in the benchmark used, but to point out the need for more robust statistical analysis to reach general conclusions. The results presented here invalidate the conclusions in their original article. This may be explained by an inadequate experimental design.

This paper is organized as follows. In Section 2, a detailed theoretical review of the application of neural networks to the problem of interest is presented. In Section 3, the techniques used and the way the results were produced will be presented. In Section 4, experimental results are evaluated. And finally, section 5 analyses conclusions and perspectives for future work.

II. Previous Works

Traffic sign detection, tracking and recognition are important issues concerning autonomous and assisted driving, signaling inventory and quality control. This section brings a review of traffic sign detection and recognition approaches as well the most used public benchmarks. Similarly to other computer vision applications, a number of recent researches in these fields are based on deep learning architectures.

In therms of public databases, the most remarkable one is The German Traffic Sign Recognition Benchmark (GTSRB) [8] which contains 51,839 images and 43 classes. GTSRB has a compatriot dedicated to sign detection [9], the German traffic sign detection benchmark (GTSDB), containing 900 images with the corresponding signaling bounding boxes annotations. A more recent database, the The European traffic sign dataset (ETSD) assembles several European public available datasets: from Belgium, the KUL Belgium Traffic Signs dataset [10];

from Croatia, the MASTIF datasets [11]; from France, the Stereopolis dataset [12]; from Germany, the above mentioned GTSRB [8]; from Netherlands, the RUG Traffic Sign Image-Database [13]; and from Sweden, the Swedish Traffic Signs Dataset [14]. ETSD amounts 82,476 images of 164 classes.

A. Detection

An example of direct detection traffic sign detection is presented in [15]. This approach relies on the Single Shot Detector (SSD) architecture [16], basically, a feed forward CNN, which produces predictions on the position and class of target objects. The predicted bounding box position is then submitted to 2D Pose Prediction which fits the box to the quadrilateral which best adjusts to the traffic sing. The method ends up with a boundary corner estimation process that produces based on the sign class shape an accurate boundary for such occurrence. The presented experiments, in terms of SSD architecture adaptation, is limited to reducing computational complexity in order to accomplish processing time requirements for the application, permitting a low-power mobile platform to reach 7 FPS. Song et al. [17] proposed an efficient CNN which remarkably minimize the redundancy, downsize the parameters set and speed up the networks. So, it reduces its computational cost, achieving 833 ms per frame on a 2048×2048 px image.

An region proposal approach is presented in [18]. The proposed deep detection network is composed of four modules. Firstly, CNN layers that compute features. In parallel, the so-called attention network, which makes a rough detection, is a color segmentation module, exploiting intrinsic properties of signs. The third module employs a fully convolutional network to produce the final regions proposals. The last module is an improved Fast Region-based Convolutional Network (Fast R-CNN), functioning as a detector (classifier and regressor) and synthesizing information from the remaining modules. In the experiments, the method is compared with other approaches, without concerning optimizing the internal architecture. In the most successful experiments using a GPU equipped computer, produced a 7.8 FPS for input frames of 1024×800 px.

A combination of image analysis and pattern recognition techniques for traffic sign detection dedicated to mobile systems is presented in [19]. The method is based on complementary interest regions extraction approaches relying on color and shape which follow a preprocessing stage which enhance traffic sign regions and fade background. The candidate regions provided by the interest region detectors are then classified as either traffic sign or background by a Support Vector Machine (SVM) using Histograms of Oriented Gradient (HOG) features. Regions claimed as signs are then filtered in order to eliminate false positives.

An adaptive color method for sign detection method based on adaptive color threshold is presented [20]. First an adaptive segmentation threshold is calculated using the cumulative distribution function of the image histogram. Afterwards, an approximate maximum and minimum normalization method is used to suppress the interference of high brightness and background areas. Results are submitted to a shape symmetry detection algorithm based on statistical hypothesis testing. The experimental evaluation on the GTSDB obtained an accuracy which exceeded 94%.

A method for detection and classification of traffic sign is presented in [21]. Roughly speaking, the method can be split up into color based ROIs segmentation and shape classification. While K-means and an area-based filter are exploited for ROIs extraction, shape classification extract pyramids of HOGs which are discriminated by a SVM.

B. Traffic Signs Recognition

A number of scientific studies in the literature are dedicated to traffic signs recognition. Their performance comparison is easier when they use the GTSRB, the widest spreading traffic signs recognition benchmark. [22] present and evaluate the use of Spatial Transform Network (STN) and CNN. The most successful assemblage was STN-CNN-STN-CNN-STN-CNN consisting of more than 14 million of parameters which achieve an accuracy 99.71%. The deep learning architecture that won the contest in the IJCNN 2011 [23] is presented in [24]. It consists of a committee of 25 CNNs, encompassing approximately 38.5 million of parameters and achieving 99.46% accuracy. Each one of the 25 CNNs parameters are initialized randomly, five well-known image enhancement techniques are presented to the input of five specialized CNNs. Outputs of each CNN relative to each class are democratically averaged producing the outcome of the so-called Multi-Column Deep Neural Network. The use of Multiscale-CNNs was proposed in [25], concerning on a two stages CNNs in which the output of the first stage is also presented, after an additional pooling, to the fully connected layer, conveying a multi-scale feature representation. Authors present some variations of architectures, the most successful consisting in receiving only a gray level image as input which obtained 99.17% accuracy on GTSRB while having 1,437,791 parameters to be trained.

A traffic sign recognition approach based on a combination of complementary and discriminant feature sets containing HOG, Gabor features and Compound local binary pattern is proposed in [26]. The method used a extreme learning machine (ELM) network as classifier. The results of the experimental work concerning the GTSRB reached 99.10% of accuracy. A similar approach using SVM [27] achieved 97.04%. An approach based on robust traffic sign image descriptor, consisting on a variant of HOG, and sparse classifiers is presented in [28]. The method provided 98.17% of accuracy on GTSRB.

III. Methodology

As previously introduced, in this work will be made a review of [7], so we implement the same architecture of CNN, but we plan our experiments to enable a more thorough statistical analysis of the outcomes.

A. Convolutional neural network architecture

The herein presented CNN is composed of a convolutional layer, a layer of dimensionality reduction (pooling), one hidden layer and the output layer. A $3 \times 3 \times 3$ version of the standard

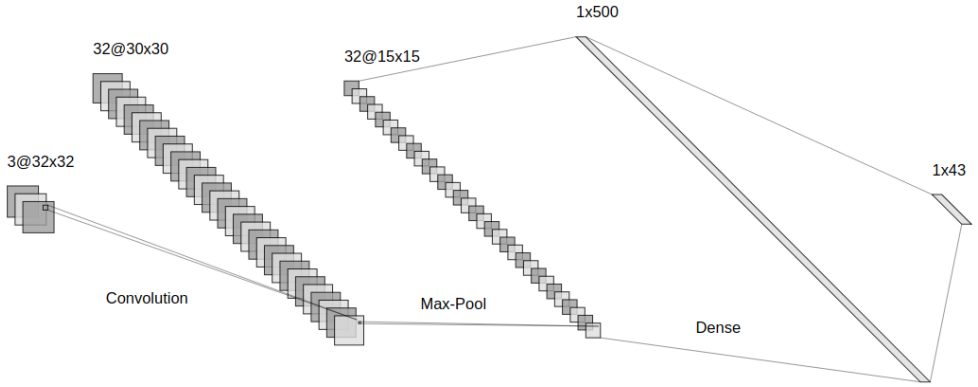


Fig. 1. CNN architecture example with a 3x3 kernels' size.

TABLE I
CNN SPECIFICATIONS

Parameters	Description
Weights Initialization	HE Normal
Weights Update	Policy Adam
Activation Function	ReLU
Pooling	2 x 2 Max
Loss Function	Negative log-likelihood
Cost Function	Average of Loss Functions
Stride for Convolution Layer	1
Stride for Pooling Layer	2

convolutional neural network applied for the problem of traffic signs recognition is presented in Fig. 1. The architecture receives a 32×32 RGB input image which is submitted to by 32 $N \times N \times 3$ filters, where distinct values for N are to be evaluated in the experiments. Filtered maps are, then, processed by a rectified linear unity activation function followed by a 2×2 max pooling. Remaining maps are fully connected to a hidden layer with 500 neurons which in turn are connected to the 43 neurons on the output layer, accordingly to the number of classes in the dataset. Table I presents the functions and some other specific characteristics used in such CNN implementation.

B. Statistical analysis

In order to compare multiple models in machine learning Pizarro [29] proposes two approaches: Parametric Analysis and non-parametric analysis. However, as [30] points out, parametric analyzes (e.g. ANOVA) is based on assumptions that the samples are drawn from normal distributions and, in general, there is no guarantee for normality of classification accuracy distributions across a set of problems. Therefore, in this work, a non-parametric analysis of the accuracy of the models will be made.

Dietterich [31] proposes tests based on 5×2 cross validation as a strategy that counterbalances the need for multiple runs, while avoiding overlapping test sets for each round (which inflates the hypothesis of independence between runs). Otherwise [29] proposes thirty rounds of execution with re-shredding of the data, however with multiple executions every time to deal with outliers.

The nonparametric approach consists of transforming each round of execution, in relatively ranked results. So the best result (highest accuracy and / or lowest error rate) is "the first", that is, receive rank 1. Similarly, rank two, three, four, etc. and so on to the other results are assigned for each of the thirty repetitions.

Initially we tested the hypothesis that all the algorithms were equivalent and that the difference between results in each round is due to nothing more than luck. If this is true, no algorithm should perform better than another consistently, that is, if this hypothesis is true, even if in some cycle, one of the algorithms is better than another, in general, the average rank of all of them must be the same. For this we use the Friedman's [32] test.

In Equation 1, be r_{ij} the rank of the j-th of k algorithms on the i-th of N data sets. The Friedman test compares the average ranks of algorithms $\frac{1}{n}\sum r_{ij}$, about the null hypothesis, which states that all the algorithms are equivalent and so their average ranks should be equal. Being k the number of models and n the total number of rounds of execution of the models.

$$\chi_F^2 = \frac{12n}{k(k+1)} \left[\sum_{j=1}^{k} \left(\frac{1}{n} \sum_{i=1}^{n} r_{ij} \right)^2 \right] - 3n(k+1) \quad (1)$$

If it is possible to falsify the hypothesis that all algorithms have an equivalent accuracy, the question arises which algorithm is better and how they can be compared with each other. In this problem [30] proposes the use of tests that avoid two-by-two comparisons, for this purpose we use the Nemenyi Test [33] to calculate the difference between the averages of rank and compare them with a "critical difference" (CD). The CD tests if there is statistical significance to affirm that there is a difference between the accuracy of the methods. Equation 2 presents the calculation of the critical difference by the Nemenyi method:

$$CD_F = z_{adj} \sqrt{\frac{nk(k+1)}{6}}, \quad (2)$$

where n and k are the same as in Eq. 1 and the value of z_{adj} is obtained from the table of the Normal Distribution [34] and will be a function of Type I error rate that will be tolerated by researches.

IV. Experiments

A. Experiments design

The dataset used for training and evaluating the CNN performance in this work is [35] which is the same employed in [7]. Broadly speaking, it is a pre-processed derivation of the GTSRB [8], with insertion of artificially generated data to balance the number of available elements of all classes. In the following experiments, the dataset was divided between training, validation and test data, being respectively 86,989, 4,410 and 12,630, as in the reference article. However, differently from [7], each network configuration was trained from the scratch for 30 times.

The purpose of the experiments is evaluating the influence of a specific CNN parameter value (in this case, the kernel size of the convolutional layer) on the accuracy of the network. The choice of accuracy as an analysis parameter was to allow comparisons with the reference article [7]. The analyzed models have kernel sizes 3×3, 5×5, 9×9, 13×13, 15×15, 19×19, 23×23, 25×25 and 31×31. As the database images are RGB 32×32 px, these kernel sizes were varied from the smallest possible 3×3 to close to the maximum possible 31×31 and are in accordance with the reference article.

B. Results and Analysis

Fig. 2 shows the results of the thirty executions' accuracy for each model. The boxplot show the mean, standard deviation, minimum and maximum acuracy.

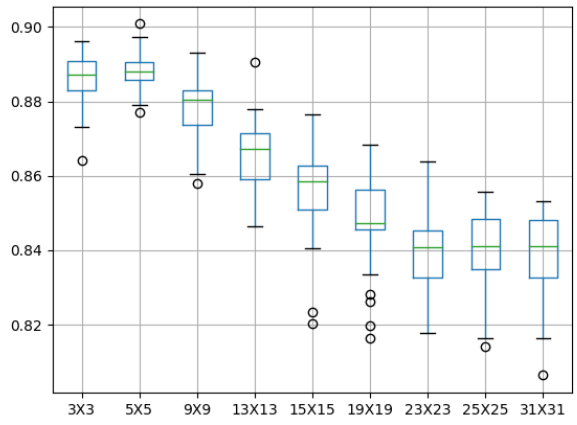


Fig. 2. BoxPlot of Model's Accuracy.

Is already possible to notice that there is a great variation between the results of the models. Models with better kernel size obtain better results, especially models with kernel size 3×3 and 5×5, with the latter having the highest mean accuracy.

TABLE II
Average accuracy table

Average Accuracy				
3X3	**5X5**	**9X9**	**13X13**	**15X15**
0.886685	0.888390	0.878694	0.866030	0.856490

Average Accuracy			
19X19	**23X23**	**25X25**	**31X31**
0.847786	0.840878	0.839751	0.839692

Table II shows the average accuracy in each model. Comparing these results with those presented in the reference paper [7] it is possible to notice that, with the exception of the accuracy for 3x3 and 5x5, the average accuracy presented by Sichkar and Kolyubin paper fit inside of the distance of two standard deviations from the mean accuracy obtained in our experiments.

To perform the non-parametric test of the null-hypothesis that all the models are equivalent, we must rewrite the results in terms of the relative rank they obtained in each round [36]. In this way, the best result (the most accurate) receives rank 1, the second highest accuracy receives rank 2 and so on until all methods are ranked in each round for all rounds. For all of the thirty rounds, each method will receive a rank between 1 and 9.

Fig. 3 presents the histogram of the ranking results as discussed previously.

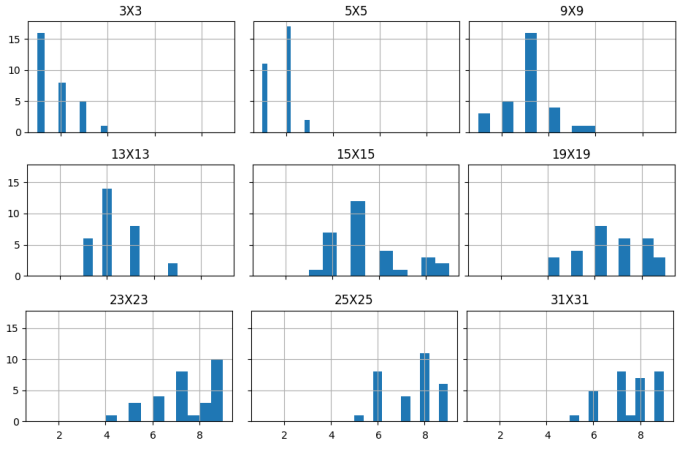


Fig. 3. Histogram of Rank of Models.

It is possible to notice that some methods were ranked with non-integer values. This occurs when two methods achieved exactly the same result in one run, so they were given the average between the ranks (e.g., instead of both being classified as seventh place, or both being classified as eighth place, the two received rank "7.5" and the other methods are classified regardless of what happened).

When comparing the results of the ranks' histograms with the boxplot, it is noteworthy that the model with a 3×3 kernel seems to have better rank results than the 5×5 model, even though the second has a higher average and a smaller variance

TABLE III
AVERAGE RANK

Average Rank				
3X3	**5X5**	**9X9**	**13X13**	**15X15**
1.7	1.7	2.3	4.27	5.47

Average Rank			
19X19	**23X23**	**25X25**	**31X31**
6.57	7.35	7.43	7.55

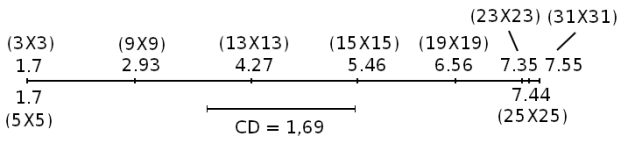


Fig. 4. Average Ranks dispose in a ruler, cooperation with CD.

around the average. This can be explained because the rankings depend not only on the accuracy of the model in each round, but also on how this accuracy is compared to the other models in the same round. So, when we represent the results in terms of rank, part of the correlation between the accuracy of the models becomes explicit, which is not possible to notice when we look only at the accuracy distribution of each model individually.

Table III shows the average rank of each model.

From the ranks averages we can have a good understanding of how the models perform in relation to others.

To test the null hypothesis (what explains the variation between the data is luck) we will use the chi-square [34] (Table A4). Table of the Chi-Square Distribution for p-value of 0.99 (with 8 grades of freedom) we have $\chi^2_{0.99} = 20.09$. Equation 1 presents the chi-square estimation for our problem. Calculating for $n = 30$ and $k = 9$ we get $\chi^2_F = 187.49$, that is, the result of this calculation shows us that the null hypothesis can be rejected.

At that point we initiate the post hoc analysis. Equation 2 presents the calculation of the critical difference by Nemenyi's test [33]. Using values proposed by [34], assuming the per comparison Type I error rate (α_{PC}) of 0.05, we will use a $z_{adj} = 2.39$. The Equation 2 result $CD = 50.7$. If we normalize the CD by the number of replications we can directly compare the value with the average rank of each model [30]. So our normalized critical difference is $CD/n = CD_n = 1.69$.

The idea behind Neyemin's test is that when performing multiple independent 2-by-2 tests the probability that at last one of them, by chance, results in a false positive increases exponentially with the number of models. The critical difference is a factor that already considers the number of models to be compared and, instead of conducting all paired t-tests (e.g., in our case $9 \times 8/2 = 36$ comparisons), we can compare all the differences between ranks models with CD to determine if the difference between models' results has statistical significance.

Fig. 4 presents the critical difference as a distance, placing all average ranks in a "ruler" for comparison. If the size of the difference between the average rank of two models is greater than CD, then the hypothesis that they are equivalent can be rejected, otherwise, there is no statistical significance in the difference between the results, so this test says nothing about these models.

The image shows a spatial perspective about the relationship between the average rank of models and the calculated CD.

Finally, a Table IV with the final results of the comparisons between the models. We can see that the statistical analysis indicates which are the best models with 3×3, 5×5 and 9×9 kernel sizes and that we cannot show that there is a statistical difference between them. Our conclusions differ from the reference article [7] since our results indicates smaller kernels provides a greater accuracy for a CNN with this architecture.

V. CONCLUSION

This paper has presented the use of standard convolutional neural networks for the problem of traffic signs recognition. The architecture recieves a 32×32 RGB input image which is convolved by 32 $N \times N \times 3$ filters. Filtered maps are, then, submitted to a rectified linear unity activation function followed by a 2×2 max pooling. Remaining maps are fully connected to a hidden layer with 500 neurons which in turn are connected to the 43 neurons on the output layer, accordingly to the number of classes in the dataset, which was derived from the German traffic sign recognition benchmark. Nine distinct values for the N parameters were evaluated, each of them was trained from the scratch for 30 times.

The statistical analysis herein presented indicates that the best results where provided by convolutional layers of 3×3, 5×5 and 9×9 which did not produced significant statistic difference. This conclusion is somehow different to the one presented by Sichkar and Kolyubin in [7] which pointed out 9×9 and 19×19 as the ones which produced the best accuracies. The reason for that discrepancy is probably due to the stochastic characteristic of the network training that was not so carefully taken into consideration in [7].

Future work could explore statistic analysis with multiple CNN architectures and multiple data sets of traffic sign. At same time, focus in more robust indices to determine the quality of neural networks than accuracy (ROC, AUC, ...).

REFERENCES

[1] M. G. Speranza, "Trends in transportation and logistics," *European Journal of Operational Research*, vol. 264, no. 3, pp. 830 – 836, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221716306713

[2] F. Favarò, S. O. Eurich, and N. Nader, "Autonomous vehicles' disengagements: Trends, triggers, and regulatory limitations." *Accident; analysis and prevention*, vol. 110, pp. 136–148, 2018.

[3] O. Abiodun, A. Jantan, O. Omolara, K. Dada, N. Mohamed, and H. Arshed, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, p. e00938, 11 2018.

[4] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, *Object Recognition with Gradient-Based Learning*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 319–345.

[5] Q. Tang and K. Jo, "Analysis of various traffic sign detectors based on deep convolution network," in *2019 IEEE/SICE International Symposium on System Integration (SII)*, Jan 2019, pp. 507–511.

TABLE IV
FINAL COOPERATION BETWEEN MODELS (P-VALUE X)

Kernel's Size	Models not significantly different	Model significantly worse accuracy	Model significantly better accuracy
3X3	5x5 e 9x9	13x13,15x15,19x19,23x23,25x25 e 31x31	-
5X5	3x3 e 9x9	13x13,15x15,19x19,23x23,25x25 e 31x31	-
9X9	3x3, 5x5 e 13x13	15x15,19x19,23x23,25x25 e 31x31	-
13X13	9x9 e 15x15	19x19, 23x23, 25x25 e 31x31	3x3 e 5x5
15X15	13x13 e 19x19	23x23,25x25 e 31x31	3x3, 5x5 e 9x9
19X19	15x15, 23x23, 25x25 e 31x31	-	3x3, 5x5, 9x9 e 13x13
23X23	19x19, 25x25 e 31x31	-	3x3, 5x5, 9x9, 13x13 e 15x15
25X25	19x19, 23x23 e 31x31	-	3x3, 5x5, 9x9, 13x13 e 15x15
31X31	19x19, 23x23 e 25x25	-	3x3, 5x5, 9x9, 13x13 e 15x15

[6] A. Jain, A. Mishra, A. Shukla, and R. Tiwari, "A novel genetically optimized convolutional neural network for traffic sign recognition: A new benchmark on belgium and chinese traffic sign datasets," *Neural Processing Letters*, vol. 50, no. 3, pp. 3019–3043, 02 2019.

[7] V. Sichkar and S. Kolyubin, "Effect of various dimension convolutional layer filters on traffic sign classification accuracy," *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, vol. 19, pp. 546–552, 06 2019.

[8] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The german traffic sign recognition benchmark: A multi-class classification competition," in *The 2011 International Joint Conference on Neural Networks, IJCNN 2011, San Jose, California, USA, July 31 - August 5, 2011*. IEEE, 2011, pp. 1453–1460.

[9] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The german traffic sign detection benchmark," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*. IEEE, aug 2013.

[10] R. Timofte, K. Zimmermann, and L. Van Gool, "Multi-view traffic sign detection, recognition, and 3d localisation," *Mach. Vision Appl.*, vol. 25, no. 3, p. 633–647, Apr. 2014. [Online]. Available: https://doi.org/10.1007/s00138-011-0391-3

[11] S. Šegvić, K. Brkić, Z. Kalafatić, V. Stanisavljević, M. Ševrović, D. Budimir, and I. Dadić, "A computer vision assisted geoinformation inventory for traffic infrastructure," in *13th International IEEE Conference on Intelligent Transportation Systems*, Sep. 2010, pp. 66–73.

[12] N. Paparoditis, J.-P. Papelard, B. Cannelle, A. Devaux, B. Soheilian, N. David, and E. Houzay, "Stereopolis ii: A multi-purpose and multi-sensor 3d mobile mapping system for street visualisation and 3d metrology," *Revue française de photogrammétrie et de télédétection*, vol. 200, no. 1, pp. 69–79, 2012.

[13] C. Grigorescu and N. Petkov, "Distance sets for shape filters and shape recognition," *IEEE Transactions on Image Processing*, vol. 12, no. 10, pp. 1274–1286, Oct 2003.

[14] F. Larsson and M. Felsberg, "Using fourier descriptors and spatial models for traffic sign recognition," in *Scandinavian conference on image analysis*. Springer, 2011, pp. 238–249.

[15] H. S. Lee and K. Kim, "Simultaneous traffic sign detection and boundary estimation using convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 5, pp. 1652–1663, may 2018.

[16] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 21–37.

[17] S. Song, Z. Que, J. Hou, S. Du, and Y. Song, "An efficient convolutional neural network for small traffic sign detection," *Journal of Systems Architecture*, jan 2019.

[18] T. Yang, X. Long, A. K. Sangaiah, Z. Zheng, and C. Tong, "Deep detection network for real-life traffic sign in vehicular networks," *Computer Networks*, vol. 136, pp. 95–104, may 2018.

[20] X. Xu, J. Jin, S. Zhang, L. Zhang, S. Pu, and Z. Chen, "Smart data driven traffic sign detection method based on adaptive color threshold and shape symmetry," *Future Generation Computer Systems*, vol. 94, pp. 381–391, may 2019.

[19] S. Salti, A. Petrelli, F. Tombari, N. Fioraio, and L. D. Stefano, "Traffic sign detection via interest region extraction," *Pattern Recognition*, vol. 48, no. 4, pp. 1039–1049, apr 2015.

[21] H. Li, F. Sun, L. Liu, and L. Wang, "A novel traffic sign detection method via color segmentation and robust shape matching," *Neurocomputing*, vol. 169, pp. 77–88, dec 2015.

[22] Á. Arcos-García, J. A. Álvarez-García, and L. M. Soria-Morillo, "Deep neural network for traffic sign recognition systems: An analysis of spatial transformers and stochastic optimisation methods," *Neural Networks*, vol. 99, pp. 158–165, mar 2018.

[23] A. A. Minai, "2011 international joint conference on neural networks (ijcnn 2011) [conference reports]," *IEEE Computational Intelligence Magazine*, vol. 7, no. 1, pp. 13–15, Feb 2012.

[24] D. Cireşan, U. Meier, J. Masci, and J. Schmidhuber, "Multi-column deep neural network for traffic sign classification," *Neural Networks*, vol. 32, pp. 333–338, aug 2012.

[25] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale convolutional networks," in *The 2011 International Joint Conference on Neural Networks*, 2011, pp. 2809–2813.

[26] S. Aziz, E. A. Mohamed, and F. Youssef, "Traffic sign recognition based on multi-feature fusion and ELM classifier," *Procedia Computer Science*, vol. 127, pp. 146–153, 2018.

[27] S. Kaplan Berkaya, H. Gunduz, O. Ozsen, C. Akinlar, and S. Gunal, "On circular traffic sign detection and recognition," *Expert Systems with Applications*, vol. 48, 12 2015.

[28] P. H. Kassani and A. B. J. Teoh, "A new sparse model for traffic sign classification using soft histogram of oriented gradients," *Applied Soft Computing*, vol. 52, pp. 231–246, mar 2017.

[29] J. Pizarro, E. Guerrero, and P. Galindo, "Multiple comparison procedures applied to model selection," *Neurocomputing*, vol. 48, pp. 155–173, 10 2002.

[30] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[31] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

[32] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937. [Online]. Available: http://www.jstor.org/stable/2279372

[33] P. Nemenyi, *Distribution-free Multiple Comparisons*. Princeton University, 1963. [Online]. Available: https://books.google.com.br/books?id=nhDMtgAACAAJ

[34] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, 4th ed. Chapman & Hall/CRC, 2007.

[35] K. Sichkar V. N. (2019) Traffic signs preprocessed data. [Online]. Available: https://www.kaggle.com/valentynsichkar/traffic-signs-preprocessed

[36] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *The Annals of Mathematical Statistics*, vol. 11, no. 1, pp. 86–92, 1940. [Online]. Available: http://www.jstor.org/stable/2235971

Neonatal Pain Assessment From Facial Expression Using Deep Neural Networks

Lucas Fontes Buzuti*, Tatiany M. Heideirich†, Marina C. M. Barros†, Ruth Guinsburg†, Carlos Eduardo Thomaz*
*Departamento de Engenharia Elétrica, Centro Universitário FEI, São Bernardo do Campo-SP, Brasil
†Departamento de Pediatria, Universidade Federal de São Paulo, São Paulo-SP, Brasil

***Abstract*—Currently, neonatal pain assessment varies among health professionals, leading to late intervention and flimsy treatment of pain in several occasions. Therefore, it is essential to understand the deficiencies of the current pattern of pain assessment tools in order to develop new ones, less subjective and susceptible to external variable influences. The aim of this paper is to investigate neonatal pain assessment using two models of Deep Learning: Neonatal Convolutional Neural Network trained end-to-end and ResNet trained using Transfer Learning. We used for training two distinct databases (COPE and Unifesp) and our results showed that the use of multi-racial databases might improve the evaluation of automatic models of neonatal pain assessment.**

***Index Terms*—Automated pain recognition, deep learning, facial expression, neonatal pain assessment, pattern recognition**

I. Introdução

A definição de dor, pela Associação Internacional para o Estudo da Dor (IASP), é "uma experiência sensorial e emocional desagradável associada a danos teciduais reais, potenciais ou descrita em termos de tais danos". A capacidade de comunicação verbal da dor, ou simplesmente o ato de apontar (escala visual analógica) não se aplica para o neonato. Por várias décadas, os pediatras acreditavam que os neonatos não sentiam ou não se lembravam da dor, uma vez que suas capacidades eram limitadas devido à ausência de substrato neurológico para percepção da mesma [1]. Tal crença foi refutada por diversos estudos científicos [2] [3].

Estudos relatam que experiências dolorosas repetidas vividas pelos neonatos estão associadas a alterações que podem prejudicar a curto e longo prazos suas vidas, sendo esses: alterações na sensibilidade e percepção da dor [4] [5] [6] [7], funcionamento do sistema de resposta ao estresse (altos níveis de cortisol) [8] [9] [10] [5], entre outros. Fortes evidências em relação à exposição extensa à dor durante o período inicial da vida estão associadas a alterações estruturais e funcionais do cérebro. As alterações que ocorrem são: alterações na substância branca cerebral e na substância cinzenta subcortical [7] [11] [5], atraso no desenvolvimento corticoespinhal [6] [5], alterações no número de conexões sinápticas e neuróglia (são células não neuronais do sistema nervoso central que proporcionam suporte e nutrição aos neurônios) e na alteração do grau de ramificação capilar que aumenta o suprimento de sangue e oxigênio [12] [13]. Tais alterações podem resultar em uma variedade de alterações comportamentais, de desenvolvimento e de aprendizagem [8] [14] [15].

Anand [16] relatou que os neonatos sentiam dor e, em geral, esta não era reconhecida e, por tanto, subtratada, por isso, recomendou o uso de analgésicos, que deveriam ser prescritos de acordo com os cuidados que cada neonato necessitasse. Trabalhos relataram que o uso excessivo de medicamentos analgésicos, tais como morfina e fentanil, poderiam causar efeitos colaterais. Zwicker et al. [17] relataram estes efeitos após observar o aumento de 10 vezes do uso de morfina (um agente comumente usado para o tratamento da dor neonatal), visto que está associado ao deficit do crescimento cerebelar no período neonatal, com um quadro de resultados piores para o desenvolvimento neurológico no período da primeira infância. Diversas revisões descrevem o fentanil como um analgésico extremamente potente e listam diversos efeitos colaterais tais como, neuroexcitação e depressão respiratória, para o uso de altas doses [15] [18].

Segundo Hummel et al. [19] e Simons et al. [20], em média, quatorze procedimentos dolorosos por dia são realizados em bebês na Unidade de Terapia Intensiva Neonatal (UTIN). Métodos de avaliação da dor comumente utilizados na pediatria, como a autoavaliação e o uso de escala visual analógica, com símbolos ou números para indicar diferentes níveis de dor, são considerados o padrão-ouro. Entretanto, estes métodos não são aplicáveis na neonatologia, visto que requerem uma capacidade de comunicação complexa, ainda não presente nos recém-nascidos. Os métodos atuais para avaliar a dor nessa população vulnerável dependem da atuação de profissionais bem qualificados, que observam as múltiplas repostas comportamentais e fisiológicas aliadas. No entanto, há uma dificuldade na utilização dessas escalas, relatada por Heiderich [21], pois, uma vez que os pacientes neonatais são pré-verbais e se encontrarem em diferentes fases do desenvolvimento cognitivo, ainda existem muitas dúvidas quanto à interpretação e à avaliação das respostas à dor neste paciente.

Poucos estudos como [22] e [21] foram realizados para analisar e avaliar a dor neonatal usando tecnologias de Visão Computacional e Aprendizado de Máquina. Por outro lado, uma variedade rica de métodos foi proposto para avaliar dor de adultos [23] [24] [25] [26] [27]. Os motivos destacados por Zamzmi [1] sobre a falta de estudos para o reconhecimento da dor neonatal, principalmente usando Aprendizado Profundo, se referem aos números limitados de bancos de imagens neonatais e à crença de que os algoritmos projetados para avaliar dor em adultos teriam desempenhos semelhantes para

os neonatos, o que não acontece na prática. Apenas dois trabalhos que utilizam a tecnologias de Aprendizado Profundo são [28] e [29], que aplicaram o Aprendizado por Transferência em algumas arquiteturas de rede neurais, tal como a ResNet [30], para classificar a dor neonatal.

Este artigo tem por objetivo investigar dois modelos de Aprendizado Profundo (DL), Neonatal Convolutional Neural Network (N-CNN) [29] e ResNet50 [29], em bases de imagens distintas, para avaliação facial automática da dor neonatal.

II. Metodologia

A. RetinaFace

A localização automática da face é uma etapa pré-requisitada na análise de imagens faciais para muitas aplicações, como atributo facial [31] e reconhecimento de identidade facial [32]. Uma definição estreita de localização de face pode se referir à detecção de face tradicional [33], que visa estimar as caixas delimitadoras de face sem nenhuma escala e posição anterior. Este artigo fez uso do algoritmo RetinaFace proposto por Deng et al. [32], visto que utiliza de uma definição mais ampla de localização de face, assim encontrando faces em qualquer posição, incluindo: detecção de face, alinhamento de face, análise de face em pixel e regressão de correspondência densa em 3D. Esse tipo de localização densa da face fornece informações precisas da posição facial para todas as escalas diferentes.

No treinamento da RetinaFace [32] os autores utilizaram o erro de multi-tarefas (multi-task loss), no intuito de minimizar o erro da caixa de âncora i. Tal erro é definido sendo:

$$\begin{aligned} L =& L_{cls}(p_i, p_i^*) + \lambda_1 p_i^* L_{box}(t_i, t_i^*) + \\ & \lambda_2 p_i^* L_{pts}(l_i, l_i^*) + \lambda_3 p_i^* L_{pixel}. \end{aligned} \tag{1}$$

$L_{cls}(p_i, p_i^*)$ erro da classificação, em que p_i é a probabilidade prevista de i ser uma face, p_i^* será 1 para i positivo e 0 para i negativo, e L_{cls} é a função softmax para classes binárias (face/não face). $L_{box}(t_i, t_i^*)$ erro de regressão da caixa de face, onde $t_i = \{t_x, t_y, t_w, t_h\}_i$ e $t_i^* = \{t_x^*, t_y^*, t_w^*, t_h^*\}_i$ representando as coordenadas da caixa prevista e a caixa do ground-truth associado ao i positivo. $L_{pts}(l_i, l_i^*)$ erro de regressão dos pontos faciais, uma vez que $l_i = \{l_{x1}, l_{y1}, \ldots, l_{x5}, l_{y5}\}_i$ e $l_i^* = \{l_{x1}^*, l_{y1}^*, \ldots, l_{x5}^*, l_{y5}^*\}_i$ representam os cinco marcos faciais previstos e os cinco marcos faciais do ground-truth associado ao i positivo. O erro da regressão densa L_{pixel} é definida pela Equação (2). Os parâmetros de balanceamento dos erros λ_1, λ_2 e λ_3 são definidos em 0,25, 0,1 e 0,01, o que significa que há um aumento na importância de melhores locais de caixa e ponto de referência a partir dos sinais de supervisão [32].

$$L_{pixel} = \frac{1}{W * H} \sum_{i}^{W} \sum_{j}^{H} \parallel \mathcal{R}(\mathcal{D}_{P_{ST}}, P_{cam}, P_{ill})_{i,j} - I_{i,j}^* \parallel_1, \tag{2}$$

após a convolução gráfica, vide [32], os autores computaram os parâmetros de forma e textura $P_{ST} \in R^{128}$ para projetar uma malha colorida $\mathcal{D}_{P_{ST}}$ em um plano da imagem 2D com parâmetros de câmera $P_{cam} = [x_c, y_c, z_c, x_c^{'}, y_c^{'}, z_c^{'}, f_c]$ (ou seja, localização da câmera, posição da câmera e distância focal) e parâmetros de iluminação $P_{ill} = [x_l, y_l, z_l, r_l, g_l, b_l, r_a, g_a, b_a]$ (ou seja, localização da fonte de luz pontual, valores de cores e cores da iluminação ambiente). Com a face 2D renderizada $\mathcal{R}(\mathcal{D}_{P_{ST}}, P_{cam}, P_{ill})$, foi comparado a diferença de pixel da face 2D renderizada e a face original 2D, como mostra a Equação 2, W e H são a largura e altura de i do corte da face $I_{i,j}^*$, respectivamente.

B. ResNet

A ResNet proposta por He et al. [30] revolucionou a corrida arquitetônica da CNN ao introduzir o conceito de aprendizado residual e o desenvolvendo de uma metodologia eficiente para o treinamento de redes profundas. Semelhante as Highway Networks [34] a ResNet também é colocada na categoria das CNNs com vários caminhos.

He et al. [30] adotaram na ResNet o aprendizado residual em todas as camadas empilhadas. A formulação matemática do bloco foi definida sendo:

$$y = \mathcal{F}(x, W_i) + x, \tag{3}$$

Tem-se x e y como os vetores de entrada e saída das camadas consideradas. A função $\mathcal{F}(x, W_i)$ representa o mapeamento residual a ser aprendido. Supondo um bloco de construção residual com duas camadas, portanto, se tem: $\mathcal{F} = W_2\sigma(W_1 x)$, na qual σ denota ReLU e os biases foram omitidos para simplificar as notações. A operação $\mathcal{F} + x$ é realizada por um salto de conexão e uma adição elemento a elemento, tendo assim então y. He et al. também consideraram uma segunda não linearidade após a adição, ou seja, $\sigma(y)$.

C. Neonatal Convolutional Neural Network

Devido aos números baixos de pesquisas relacionada à classificação da dor neonatal e principalmente pesquisas utilizando Aprendizagem Profunda, Zamzmi et al. [29] propuseram uma topologia de CNN, denominada Neonatal Convolutional Neural Network (N-CNN), para extração de característica e classificação da dor neonatal. Segundo Zamzmi et al. [29], tal topologia foi a primeira modelada utilizando Aprendizagem Profunda para classificar dor neonatal. Este artigo fez uso desse modelo conforme está descrito em [29], com uma vetorização entre a 9° camada e a 10° camada (tal vetorização não está descrita em [29]). Os parâmetros da N-CNN são sendo apresentados na Tabela I.

III. Materiais

Para a construção do arcabouço computacional para avaliar a dor neonatal nos modelos de Aprendizado Profundo, foi utilizado a linguagem de programação Python3 e o open source AI framework para aprendizado de máquina e computação numérica de alta performance da Google, denominado TensorFlow.

TABLE I
PARÂMETROS DA N-CNN [29].

Branch	Layer	Type	Input	Filters	Filter Size	Activation	Regularization
Left	Layer 1	Max Pool 1	120 × 120 × 3	-	10 × 10, st. 10, pd. 0	-	-
Central	Layer 2	Conv 1	120 × 120 × 3	64	5 × 5, st. 1, pd. 0	Leaky ReLU (0.01)	-
	Layer 3	Max Pool 2	Layer 2	-	3 × 3, st. 3, pd. 0	-	-
	Layer 4	Conv 2	Layer 3	64	2 × 2, st. 1, pd. 0	Leaky ReLU (0.01)	-
	Layer 5	Max Pool 3	Layer 4	-	3 × 3, st. 3, pd. 0	-	Dropout (0.1)
Right	Layer 6	Conv 3	120 × 120 × 3	64	5 × 5, st. 1, pd. 0	Leaky ReLU (0.01)	-
	Layer 7	Max Pool 4	Layer 6	-	10 × 10, st. 10, pd. 0	-	Dropout (0.1)
Merge Layer (Left —— Central —— Right)							
	Layer 8	Conv 4	Merge Layer	64	2 × 2, st. 1, pd. 0	ReLU	-
	Layer 9	Max Pool 5	Layer 8	-	2 × 2, st. 2, pd. 0	-	-
Vectorization(Layer 9)							
	Layer 10	FC1	Layer 9	-	-	ReLU	L2 Regularizer (0.01) Dropout (0.1)
	Layer 11	FC2	Layer 10	-	-	Sigmoid	-

Bancos de imagens para proporcionar estudos utilizando Visão Computacional, Aprendizado de Máquina e principalmente Aprendizado Profundo na análise e avaliação da dor neonatal ainda são poucos. Este artigo fez uso de dois bancos neonatais: COPE [35], um dos primeiros bancos de imagens na finalidade de analisar e avaliar a dor neonatal, e o banco de imagens neonatal da Unifesp [36].

A. COPE

Para o desenvolvimento do COPE, Brahnam et al. [35] aplicaram estímulos que provocam expressões faciais, com o objetivo de serem utilizados nas avaliações e análises da dor em neonatos. Os estímulos aplicados foram: punção do calcanhar (teste do pezinho), fricção na superfície lateral externa do calcanhar, transporte de um berço para outro e estímulo aéreo. O estímulo aéreo no nariz teve a intenção de provocar aperto nos olhos, simulando mudanças de iluminação [35]. O banco de imagens COPE contêm 288 imagens coloridas com 3008×2000 pixels de 26 recém-nascidos caucasianos, 13 meninos e 13 meninas.

B. Unifesp

Heiderich et al. [36] construíram um banco de imagens neonatais (banco de imagens Unifesp) para a elaboração de um software capaz de identificar automaticamente a expressão de dor do recém-nascido. O *back-end* do software utilizou a distância entre pontos específicos identificados no rosto do recém-nascido e a escala unidimensional NFCS [37], assim classificando a existência ou não da dor. O banco de imagens foi construído a partir de fotos capturadas antes, durante e depois de procedimentos dolorosos aplicados a essa população, como: punção venosa, capilar ou injeção intramuscular (procedimentos comuns e necessários). O banco de imagens da Unifesp contêm 360 imagens coloridas com 450×233 pixels de 30 recém-nascidos entre 34 e 41 semanas e idade gestacional e entre 24 e 168 horas de vida (prematuros tardios ou a termo).

IV. IMPLEMENTAÇÃO

Para detectar somente a face do neonato, utilizou-se do detector facial RetinaFace já treinado em cada imagem do banco. O detector retornou as coordenadas de cada face contida na imagem[1] e 5 pontos faciais (landmarks). O detector também enviou as coordenadas desses pontos e uma mensagem de falha para indicar se a detecção não ocorreu[2]. Para cada imagem foram utilizadas as coordenadas detectadas para registrar e cortar a região exata da face do neonato. Em seguida, cada imagem sofreu um específico redimensionamento conforme a dimensão de entrada das topologias. Para os experimentos com N-CNN cada imagem foi redimensionada para 120×120 e com ResNet50 o redimensionamento foi de 224 × 224. O método utilizado para redimensionar as imagens foi o bicubico.

Em todo o arcabouço foi utilizado o aumento dos dados (Data Augmentation). Portanto, este artigo fez o uso do mesmo aumento de dados utilizado no trabalho [29], no conjunto de treinamento e validação. Para a construção dos conjuntos de treinamento, validação e teste, após o conjunto original sofrer o corte da região exata da face do bebê, por meio do detector facial RetinaFace, foi dividido randomicamente em 50% gerando o conjunto I_{test} e I^*. No conjunto de dados I^* aplicou-se o aumento de dados, sendo:

- Cada imagem foi randomicamente rotacionada até 30° (1° a 30°), para gerar um total de 12 imagens para cada imagem
- Cada imagem rotacionada foi invertida horizontalmente e verticalmente, assim gerando um total de 24 imagens para cada imagem rotacionada.

Esse procedimento gerou um total de $36|I^*|$ e o novo número de amostra do conjunto ficou $|I^*| = 36|I^*| + |I^*|$. Os conjuntos de treinamento e validação foram obtidos a partir da divisão randomicamente do I^*, ficando 80% o conjunto de treinamento I_{train} e 20% o conjunto de validação I_{val}.

Os modelos avaliados por este artigo foram três: N-CNN [29], ResNet50 aplicando TL proposta por Zamzmi em [29] e ResNet50 aplicando TL proposta por este artigo. A ResNet50 porposta em [29], está removendo a última camada que define as classes do modelo e adicionando 1 único neurônio com

[1]As imagens no banco só contêm apenas uma face, sendo essa face de um bebê.

[2]Não houve falha nos bancos de imagens.

função sigmoid, em que 1 indica o estado dor e 0 o estado sem dor. O modelo adotado em [29] está sendo mostrada na Tabela II. Este artigo propôs também uma ResNet50, denominada ResNet50(ours), com a aplicação do TL, entretanto, ao invés de utilizar 1 único neurônio utilizou-se 2 neurônios com a função sotfmax, em que [0 1] define o estado dor e [1 0] o estado sem dor. A Tabela III mostra o modelo proposto.

TABLE II
ARQUITETURA RESNET50 PROPOSTA EM [29].

Global Average Pooling	Base model output
Dropout	0.5
Full 1	1, sigmoid
Total parameters	23.688.065

TABLE III
ARQUITETURA RESNET50(OURS) PROPOSTO POR ESTE ARTIGO.

Global Average Pooling	Base model output
Dropout	0.5
Full 1	2, softmax
Total parameters	23.788.418

Os modelos foram treinados usando o tamanho de lote (16) e taxa de aprendizado (0.0001) com o algoritmo de descida de gradiente RMSprop. [38].

V. EXPERIMENTOS E RESULTADOS

Os bancos de imagens que foram utilizados neste artigo passaram pela etapa de pré-processamento, mas antes de tal etapa, as imagens foram selecionadas para a construção dos conjuntos de dados contendo os estados "Dor" e "Sem Dor". O banco de imagens da Unifesp já contém os dois estados que este artigo fez uso, mas houve a necessidade de não utilizar 4 imagens, uma vez que estavam sem seus devidos rótulos. Portanto, utilizou-se 356 imagens do banco de imagens da Unifesp. Em relação ao segundo banco utilizado, banco de imagens COPE, visto que é um banco que além dos estados necessários para o experimento há mais estados (choro, estímulo de ar e fricção), contabilizando um total de 288 imagens. Sendo assim, foram selecionadas as imagens com os rótulos *rest* e *pain* sendo-os sem dor e dor, respectivamente. Após tal seleção, contabilizou-se 153 imagens do banco COPE. Após a seleção das imagens, os bancos passaram pela etapa de pré-processamento. Logo, os conjuntos de imagens ficaram:

- Banco de imagens Unifesp: $I_{train} = 5269$ imagens, $I_{val} = 1317$ imagens, $I_{test} = 178$ imagens
- Banco de imagens COPE: $I_{train} = 2279$ imagens, $I_{val} = 570$ imagens, $I_{test} = 76$ imagens

Nas Tabelas IV e V são mostradas as eficiências de cada modelo treinado, validado e testado com o banco da Unifesp e COPE, respectivamente, e também mostra uma segunda acurácia quando os modelos já treinados são testados com outro banco (os modelos que foram treinados com o banco de imagens da Unifesp após o treinamento foram submetidos ao banco de imagens da COPE, assim computando essa segunda acurácia, e mutuamente).

TABLE IV
RESULTADO DA AVALIAÇÃO DA DOR NEONATAL EM EXPRESSÃO FACIAL DO MODELO TREINADO, VALIDADO E TESTADO COM O BANCO DE IMAGENS UNIFESP (ACURÁCIA DE TESTE 2° COLUNA) E NA 3° COLUNA UMA SEGUNDA ACURÁCIA, NO QUAL O MODELO APÓS O TREINAMENTO FOI SUBMETIDO AO BANCO DE IMAGENS COPE.

Model (Unifesp)	**Accuracy (Unifesp)**	**Accuracy (COPE)**
N-CNN [29]	80.1%	70.3 %
ResNet50 [29]	78.6%	59.4%
ResNet50 (ours)	78.4%	57.8%

TABLE V
RESULTADO DA AVALIAÇÃO DA DOR NEONATAL EM EXPRESSÃO FACIAL DO MODELO TREINADO, VALIDADO E TESTADO COM O BANCO DE IMAGENS COPE (ACURÁCIA DE TESTE 2° COLUNA) E NA 3° COLUNA UMA SEGUNDA ACURÁCIA, NO QUAL O MODELO APÓS O TREINAMENTO FOI SUBMETIDO AO BANCO DE IMAGENS UNIFESP.

Model (COPE)	**Accuracy (COPE)**	**Accuracy (Unifesp)**
N-CNN [29]	81.2%	43.2%
ResNet50 [29]	71.9%	53.4%
ResNet50 (ours)	87.5%	53.4%

Os resultados trazidos das Tabelas IV e V demostram em uma primeira análise que o modelo ResNet50 proposto por este artigo é superior ao modelo ResNet50 proposto em [29], uma vez que o modelo treinado nos dois bancos de imagens obtive uma acurácia relativamente boa (acima de 70%). Tal afirmação pode ser validada analisando a acurácia no modelo junto com uma segunda acurácia, a qual foi obtida introduzindo outro banco de imagens. Sendo assim na Tabela IV a 3° e 4° linha indicam uma acurácia homogênea tanto do modelo (2° coluna) quanto da segunda acurácia (3° coluna), mas impedindo extrair uma conclusão de qual modelo é superior. Na Tabela V a 3° e 4° linha indicam que o modelo proposto em [29] foi inferior ao modelo proposto por este artigo, porém, quando os modelos foram testados com outro banco de imagens obtiveram uma mesma acurácia (3° coluna). Portanto, a alteração do modelo ResNet50 proposto por este artigo pode ser considerada melhor ao modelo ResNet50 proposto em [29], para o banco de imagens COPE, visto que obteve uma acurácia de 87.5% contra os 71.9%, mas ressaltando que tal afirmação está sendo realizada com ressalva, devido ao número de imagens dos dois bancos utilizados e principalmente às características dos bancos, uma vez que o banco da COPE consiste apenas de neonatos caucasianos e o banco da Unifesp consiste em um conjunto maior de raças, sendo um banco plurirracial.

O argumento em relação ao número de imagens presentes nos bancos e principalmente nas características dos bancos, torna-se mais visível quando analisa o modelo N-CNN treinado em ambos os bancos de imagens e testado de forma a verificar seu desempenho em outro banco. Os valores da 2° linha nas Tabelas IV e V, mostram que a N-CNN treinada com a COPE foi aproximadamente 1% melhor que a N-CNN treinada com o banco da Unifesp, entretanto, ao analisar a

N-CNN treinada com o banco da Unifesp mostrou-se mais flexível ao ser submetida a um teste com outro banco de imagens, obtendo uma acurácia de 70.3% e a N-CNN treinada com a COPE obteve 43.2% de acurácia. Logo, é visível que o número de imagens para treinamento é relevante no desempenho de um modelo de DL e mesmo impondo um aumento de dados não significa que o problema do número de imagens de treinamento foi resolvido, pois o aumento de dados não cria novos dados, apenas tenta resolver o problema de Posição, Orientação e Escala (POS). Além da questão do número de imagens, o que se torna mais relevante nesse artigo não é o fato que a N-CNN proposta em [29] foi melhor que a ResNet, um modelo que é considerado o estado-da-arte, uma vez que a N-CNN foi modelada para a tarefa de classificação da dor neonatal e a ResNet se mostrou com bom desempenho nas tarefas de reconhecimento e localização de imagens para muitas tarefas de reconhecimento visual. A grande relevância está na constatação de que, fora a importância do número de imagens no banco, a diversidade racial das imagens é imprescindível, uma vez que os modelos que foram treinados com o banco de imagens plurirracial (Banco da Unifesp) obtiveram uma flexibilidade melhor quando foram testados com outro banco de imagens, assim sendo superior aos modelos treinados com apenas neonatos caucasianos (Banco da COPE).

VI. Conclusão e Trabalhos Futuros

Este artigo avaliou a dor neonatal por meio da expressão facial explorando e propondo modelo de Aprendizado Profundo. Primeiro, explorou-se o modelo Neonatal Convolutional Neural Network (N-CNN), proposta por Zamzmi em 2019 [29] e afirmada pela mesma sendo a primeira topologia para realizar tal avaliação; segundo, o modelo ResNet50 modificada por Zamzmi em 2019 [29] através do conceito de Aprendizado por Transferência (TL). Modificações da ResNet50 desenvolvida por Zamzmi et al. [29] foram propostas, assim criando uma outra ResNet50 a partir da primeira.

Os experimentos com dois bancos de imagens distintos, Unifesp e COPE, demostraram que a alteração do modelo Resnet50 proposto por este artigo gerou resultados melhores de classificação, para o banco de imagens COPE. Verificou-se também que o número de imagens no banco influenciam na acurácia dos modelos, mesmo tentando contornar o problema com técnica de aumento de dados, uma vez que tal técnica não cria novos dados, mas, tenta resolver o problema de POS. Além da questão do número de imagens, os resultados mostraram que treinar modelos com bancos de imagens plurirraciais pode aumentar a robustez desses modelos, pois bancos plurirraciais possibilitam um espectro maior de semânticas do estado "Dor" e "Sem Dor".

Vislumbra-se, como trabalhos futuros, estender essas análises baseadas em modelos de Aprendizado Profundo utilizando validações cruzadas, matrizes de confusão e mapas de ativação dos grupos de padrões de interesse.

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001 e do Centro Universitário FEI.

Referências

[1] G. Zamzmi, "Automatic multimodal assessment of neonatal pain," Ph.D. dissertation, University of South Florida, 2018a.

[2] K. J. Anand and D. B. Carr, "The neuroanatomy, neurophysiology, and neurochemistry of pain, stress, and analgesia in newborns and children," *Pediatric Clinics of North America*, vol. 36, no. 4, pp. 795–822, 1989.

[3] B. Golianu, E. J. Krane, K. S. Galloway, and M. Yaster, "Pediatric acute pain management," *Pediatric Clinics of North America*, vol. 47, no. 3, pp. 559–587, 2000.

[4] O. F. COMMITTEE *et al.*, "Prevention and management of procedural pain in the neonate: An update." *Pediatrics*, vol. 137, no. 2, p. e20154271, 2016.

[5] J. Vinall, S. P. Miller, V. Chau, S. Brummelte, A. R. Synnes, and R. E. Grunau, "Neonatal pain in relation to postnatal growth in infants born very preterm," *Pain*, vol. 153, no. 7, pp. 1374–1381, 2012.

[6] M. DiLorenzo, R. Pillai Riddell, and L. Holsti, "Beyond acute pain: understanding chronic pain in infancy," *Children*, vol. 3, no. 4, p. 26, 2016.

[7] S. Brummelte, R. E. Grunau, V. Chau, K. J. Poskitt, R. Brant, J. Vinall, A. Gover, A. R. Synnes, and S. P. Miller, "Procedural pain and brain development in premature newborns," *Annals of neurology*, vol. 71, no. 3, pp. 385–396, 2012.

[8] R. E. Grunau, M. T. Tu, M. F. Whitfield, T. F. Oberlander, J. Weinberg, W. Yu, P. Thiessen, G. Gosse, and D. Scheifele, "Cortisol, behavior, and heart rate reactivity to immunization pain at 4 months corrected age in infants born very preterm," *The Clinical journal of pain*, vol. 26, no. 8, p. 698, 2010.

[9] R. E. Grunau, J. Weinberg, and M. F. Whitfield, "Neonatal procedural pain and preterm infant cortisol response to novelty at 8 months," *Pediatrics*, vol. 114, no. 1, pp. e77–e84, 2004.

[10] S. M. Walker, "Translational studies identify long-term impact of prior neonatal pain experience," *Pain*, vol. 158, pp. S29–S42, 2017.

[11] A. Marchant, "'neonates do not feel pain': a critical review of the evidence," *Bioscience Horizons: The International Journal of Student Research*, vol. 7, 2014.

[12] A. T. Bhutta and K. Anand, "Vulnerability of the developing brain: neuronal mechanisms," *Clinics in perinatology*, vol. 29, no. 3, pp. 357–372, 2002.

[13] K. Anand and F. M. Scalzo, "Can adverse neonatal experiences alter brain development and subsequent behavior?" *Neonatology*, vol. 77, no. 2, pp. 69–82, 2000.

[14] R. Grunau, "Self-regulation and behavior in preterm children: effects of early pain," *Progress in pain research and management*, vol. 26, pp. 23–56, 2003.

[15] B. Stevens, C. Johnston, P. Petryshen, and A. Taddio, "Premature infant pain profile: development and initial validation," *The Clinical journal of pain*, vol. 12, no. 1, pp. 13–22, 1996.

[16] K. Anand, "International evidence-based group for neonatal pain consensus statement for the prevention and management of pain in the newborn," *Arch Pediatr Adolesc Med*, vol. 155, no. 2, pp. 173–180, 2001.

[17] J. G. Zwicker, S. P. Miller, R. E. Grunau, V. Chau, R. Brant, C. Studholme, M. Liu, A. Synnes, K. J. Poskitt, M. L. Stiver *et al.*, "Smaller cerebellar growth and poorer neurodevelopmental outcomes in very preterm infants exposed to neonatal morphine," *The Journal of pediatrics*, vol. 172, pp. 81–87, 2016.

[18] R. Guinsburg, "Avaliação e tratamento da dor no recém-nascido," *J Pediatr (Rio J)*, vol. 75, no. 3, pp. 149–60, 1999.

[19] P. Hummel and M. van Dijk, "Pain assessment: current status and challenges," in *Seminars in Fetal and Neonatal medicine*, vol. 11, no. 4. Elsevier, 2006, pp. 237–245.

[20] S. H. Simons, M. van Dijk, K. S. Anand, D. Roofthooft, R. A. van Lingen, and D. Tibboel, "Do we still hurt newborn babies?: A prospective study of procedural pain and analgesia in neonates," *Archives of pediatrics & adolescent medicine*, vol. 157, no. 11, pp. 1058–1064, 2003.

[21] T. M. Heiderich, "Desenvolvimento de software para identificar a expressão facial de dor do recém-nascido." 2013.
[22] M. Schiavenato, J. F. Byers, P. Scovanner, J. M. McMahon, Y. Xia, N. Lu, and H. He, "Neonatal pain facial expression: Evaluating the primal face of pain," *Pain*, vol. 138, no. 2, pp. 460–471, 2008.
[23] M. Velana, S. Gruss, G. Layher, P. Thiam, Y. Zhang, D. Schork, V. Kessler, S. Meudt, H. Neumann, J. Kim *et al.*, "The senseemotion database: A multimodal database for the development and systematic validation of an automatic pain-and emotion-recognition system," in *IAPR Workshop on Multimodal Pattern Recognition of Social Signals in Human-Computer Interaction*. Springer, 2016, pp. 127–139.
[24] D. L. Martinez, O. Rudovic, D. Doughty, J. A. Subramony, and R. Picard, "Automatic detection of nociceptive stimuli and pain intensity from facial expressions," *The Journal of Pain*, vol. 18, no. 4, p. S59, 2017.
[25] P. Rodriguez, G. Cucurull, J. Gonzàlez, J. M. Gonfaus, K. Nasrollahi, T. B. Moeslund, and F. X. Roca, "Deep pain: Exploiting long short-term memory networks for facial expression classification," *IEEE transactions on cybernetics*, 2017.
[26] S. Walter, S. Gruss, H. Ehleiter, J. Tan, H. C. Traue, P. Werner, A. Al-Hamadi, S. Crawcour, A. O. Andrade, and G. M. da Silva, "The biovid heat pain database data for the advancement and systematic validation of an automated pain recognition system," in *2013 IEEE international conference on cybernetics (CYBCO)*. IEEE, 2013, pp. 128–131.
[27] P. Werner, A. Al-Hamadi, and R. Niese, "Comparative learning applied to intensity rating of facial expressions of pain," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 28, no. 05, p. 1451008, 2014.
[28] G. Zamzmi, D. Goldgof, R. Kasturi, and Y. Sun, "Neonatal pain expression recognition using transfer learning," *arXiv preprint arXiv:1807.01631*, 2018b.
[29] G. Zamzmi, R. Paul, D. Goldgof, R. Kasturi, and Y. Sun, "Pain assessment from facial expression: Neonatal convolutional neural network (n-cnn)," in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–7.
[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[31] F. Zhang, T. Zhang, Q. Mao, and C. Xu, "Joint pose and expression modeling for facial expression recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3359–3368.
[32] J. Deng, J. Guo, Y. Zhou, J. Yu, I. Kotsia, and S. Zafeiriou, "Retinaface: Single-stage dense face localisation in the wild," *arXiv preprint arXiv:1905.00641*, 2019.
[33] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.
[34] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," *arXiv preprint arXiv:1505.00387*, 2015.
[35] S. Brahnam, C.-F. Chuang, F. Y. Shih, and M. R. Slack, "Machine recognition and representation of neonatal facial displays of acute pain," *Artificial intelligence in medicine*, vol. 36, no. 3, pp. 211–222, 2006.
[36] T. M. Heiderich, A. T. F. S. Leslie, and R. Guinsburg, "Neonatal procedural pain can be assessed by computer software that has good sensitivity and specificity to detect facial movements," *Acta Paediatrica*, vol. 104, no. 2, pp. e63–e69, 2015.
[37] R. V. Grunau and K. D. Craig, "Pain expression in neonates: facial action and cry," *Pain*, vol. 28, no. 3, pp. 395–410, 1987.
[38] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

RUMICAM: A New Device for Cattle Rumination Analysis

1st Gilberto Luciano de Oliveira
Universidade Católica Dom Bosco
Campo Grande, Brazil
gilbertolucoli@hotmail.com

2nd Milena dos Santos Carmona
Universidade Católica Dom Bosco
Campo Grande, Brazil
milenascarmona@gmail.com

3rd Julia Gindri Bragato Pistori
Universidade Católica Dom Bosco
Campo Grande, Brazil
juliapistori1@gmail.com

4th Patricia Morais de Oliveira
Universidade Católica Dom Bosco
Campo Grande, Brazil
paatymorais@outlook.com

5th Rodrigo Gonçalves Mateus
Universidade Católica Dom Bosco
Campo Grande, Brazil
rf4789@ucdb.br

6th Geazy Vilharva Menezes
Universidade Federal de MS
Campo Grande, Brazil
geazyme01@gmail.com

7th Vanessa Ap. de Moraes Weber
Universidade Católica Dom Bosco
Universidade Estadual do MS
Campo Grande, Brazil
vamoraes@gmail.com

8th Cleonice Alexandre Le Bourlegat
Universidade Católica Dom Bosco
Campo Grande, Brazil
clebourlegat@ucdb.br

9th Hemerson Pistori
Universidade Católica Dom Bosco
Universidade Federal de MS
Campo Grande, Brazil
pistori@ucdb.br

***Abstract*—Rumination may reveal important behavioral aspects of livestock animals and has been increasingly studied using new sensors technologies. In this work a new device was developed to collect close-up videos from the animal mouth during the rumination period. Using shallow and deep machine learning techniques, a software that classifies the basic mouth movements from these images has also been developed. A baseline performance for this equipment has been established using the F-score metric. SVM achieved the highest F-score of 79.3% for the shallow learning approach. The best F-score using deep learning was 75% using VGG16.**

I. Introduction

More efficient production methods are constantly being sought to cope with the population growth and increasingly scarce areas for agriculture. Therefore, the precision livestock farming has been inserted into the daily life on-farm as a support tool for the cattle rancher, allowing the producer to identify needs and obtain individualized information of cattle. Precision livestock farming is also proving to be an important tool to mitigate the pressure over the environment as the need for food and other livestock derived products increase worldwide [1].

In addition, [2] points out that precision farming supported by information and communication is a practical approach to cattle management that enables the use of best practices and ensures high-quality meat. In today's world connected to the network, information is passed on a scale never seen in the history of humanity. In this perspective, the precision livestock industry gains new impulses and can favor technological innovation in a constant way. This managerial approach has been increasingly used in the field, in order to diagnose failures in the strategic planning of cattle ranch. Furthermore, computer management aims to maximize production, reduce productive inputs, aiming for differentiation in the market. In view of this, precision livestock farming is essential to obtain a competitive advantage, since markets are increasingly dynamic and globalized, which do not allow errors at the primary production points.

Several recent works report promising results regarding the automation of animal behaviour data gathering for precision livestock, using different kinds of sensors, like microphones, accelerometers, pressure sensors and video cameras [3], [4]. Among the many different behaviours of interest, those related to cattle rumination are one of the most important for nutrition and health analysis [5]. The process of feeding and rumination of the animals, as well as their implications on the vitality of the herd is observed over time by cattle ranchers. In agreement, [6] state that cattle producers can estimate the number of chewing done by animals during rumination. However, for a better efficacy in the productive processes it would be impracticable to observe bovine rumination, tacitly by cattle ranchers, given the time required for observation and inaccurate diagnosis.

Currently the pressure sensor is one of the most used methods for capturing animal rumination. [6] and [7] used this resource in their research on bovine rumination. In [6] a halter-mounted pressure sensor was used to capture chewing and rumination cycles in 300 healthy cows of three distinct lactating breeds for a period of 24 hours. The mapping of chewing and rumination cycles of these animals allowed to identify intervals that can be used as references for further studies and make it possible to point out sick animals from that

country. [7] conducted an experiment on 60 cows to verify the effectiveness of the Swiss-developed device, the RumiWatch, a sensor for capturing jaw movements.

In our work we propose a different approach based on a new low-cost camera device that is attached to the animal and can provide a very close look at the cow mouth. This device, named Rumicam, is an adaptation of an old common accessory used in many rural areas in Brazil, called *canga*. This accessory is placed over the cow's neck to avoid that it enter pastures outside the ones that are destined for grazing and so has the additional advantage of being easily mounted on the animal. Figure 1 shows a cow using the Rumicam device.

Fig. 1. Picture of a cow using the device called Rumicam

A dataset of images captured by the Rumicam was created and used to train and test several machine learning classifiers. In order to give a first baseline performance for this new equipment, we tackled the problem of detecting if the cow's mouth is opened or closed. Details about this new device, the dataset, the experiments, results and discussion are presented in the next sections.

II. Materials and Methods

The Rumicam is composed of a structural backbone that follows the same design as the traditional *canga* as seen in Figure 2 but carrying two portable cameras (Fig. 2a) positioned to capture frontal videos during grazing behavior and lateral videos for observing the passage of the food bolus through the esophagus. The size of the rods that carry the cameras can be adjusted through several sliding mechanisms (Fig. 2b) in order to be used by different size and breed animals. The upper parts of the rods are covered with leather (Fig. 2c) to turn the device more comfortable for the animal, as these are the parts that have the greatest contact with the animal body. A durable storage box (Fig. 2d) allows the inclusion of additional electronics, like programmable circuit boards and extra battery source and data storage. The two cameras are Spy-pens with

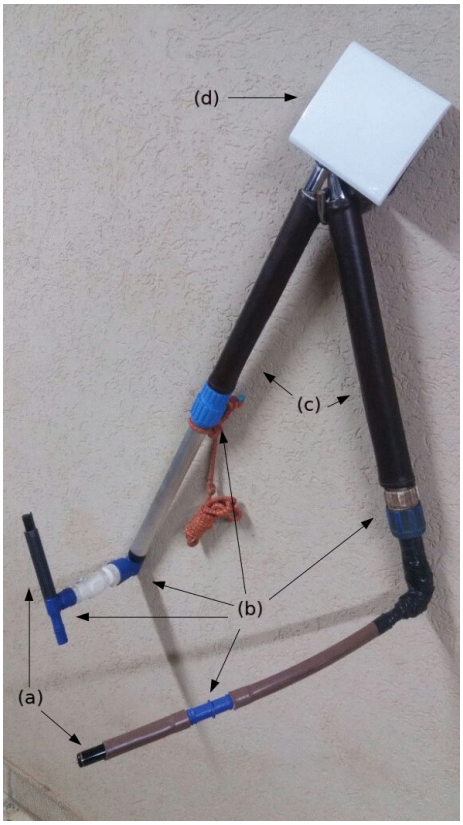


Fig. 2. Components of the Rumicam: (a) two cameras, (b) five handles for size and angles adjustments, (c) leather-coated aluminum rods and (d) durable electronics storage box

8 Gb of internal memory and record videos at 1280 X 960 pixels of spatial resolution at 30 frames per second.

For the experiments, three videos, around 30 minutes each, were recorded using the Rumicam, both in the same farm from the Brazilian city of Rio Verde de Mato Grosso (18º73'35"S, 55º12'77"W) using only the camera with the frontal view. The videos were recorded on 3 different days using 3 different cows: September 10, 2017 (late afternoon), November 5, 2017 (early morning) and January 20, 2018 (noon). Two of the cows are hybrids from Nelore (*Bos taurus indicus*) and Caracu (*Bos taurus taurus*) breeds. The third one, imaged on November, is a Nelore. Figure 3 shows one frame from each of the 3 videos and illustrates the background variability.

Two different experiments have been conducted to evaluate the performance of the equipment in the problem of detecting if the cow's mouth is opened or closed in each video frame. The first experiment used shallow learning techniques and frames extracted from the third video, capture in January 2018. The second experiment used deep learning techniques and frames extracted from the first and second videos captured on 2017. In the following, details for each experiment are presented.

(a) September 2017

(b) November 2017

(c) January 2018

Fig. 3. Pictures of the three cows captured with the Rumicam on (a) September 10, 2017 (a Nelore and Caracu Hybrid), November 5, 2017 (a Nelore) and January 20, 2018 (another Nelore and Caracu Hybrid).

(a) Opened

(b) Closed

(c) Intermediate

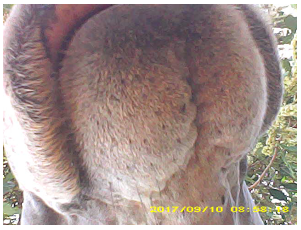
Fig. 4. One sample for each of the classes used in the first experiment: (a) cow with the mouth opened, (b) closed and (c) in a intermediate state

A. *Experiment I: Three Classes and Shallow Learning*

For the first experiment, 439 frames from the January's video have been extracted, one per second, and discarding some frames were it was not possible to see the mouth, due to the head position. The frames were divided into three classes (groups): 74 opened mouths (Fig. 4a), 170 closed mouths (Fig. 4b) and 195 images with the mouths in an intermediate position (Fig. 4c). This third class, called intermediate, represents the frames where it is not yet clear if the mouth was closed or opened.

(a) Opened Caracu Hybrid

(b) Opened Nelore



(c) Closed Caracu Hybrid

(d) Closed Nelore

Fig. 5. Two samples for each of the classes used in the second experiment, one for each different day of data collection (a) Caracu hybrid with the mouth opened, (b) Nelore with the mouth opened, (c) Caracu hybrid with the mouth closed, (d) Nelore with the mouth closed

Four supervised machine learning algorithms have been tested for the F-Score performance using a stratified 10-fold cross-validation as the sampling strategy: KNN [8], SVM [9], Adaboost [10] and Random Forest [10]. All algorithms have been configured using the default parameters values from Weka software version 3.9.1. The ANOVA hypothesis test has also been applied and the resulting p-value reported.

B. *Experiment II: Two Classes and Deep Learning*

The second experiment used the other two videos, captured in 2017. The dataset has 886 frames from these videos and is separated in only two classes: opened and (n=411) closed mouth (n=475). Figure 5 shows 4 sample frames from this dataset with two different cows with mouths opened (5a and 5b) and closed (5c and 5d).

Five deep learning architectures have been used in this second experiment: VGG16 [11], VGG19 [11], ResNet50 [12], InceptionV3 [13] and Xception [14]. All the five models were initialized with the Keras default hyper-parameters and pretrained (transfer learning) using the ImageNet weights and subsequently fine tuned. The dataset has been randomly divided to have 64% images for training, 16% for validation and 20% for testing. The following metrics have been used to measure the deep learning performance for each architecture and each class (opened and closed mouth): precision, recall and F-Score. The ANOVA hypothesis test has also been used in this experiment.

III. Results and Discussion

Regarding the first experiment, Table I shows the F-Scores for each class and classifier together with their weighted average. The SVM presented the highest mean F-Score of 79.3% and also the highest F-Score for the classes closed

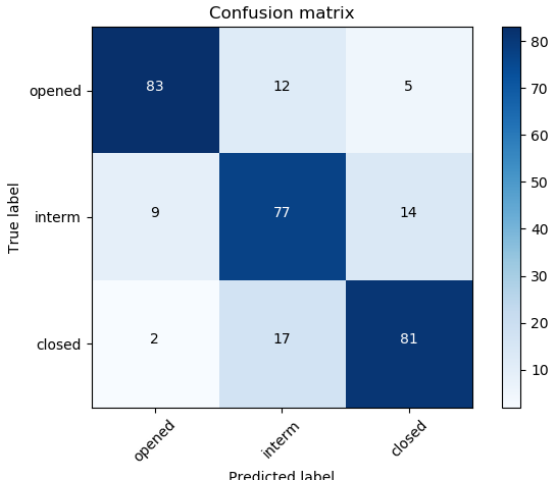


Fig. 6. Normalized confusion matrix for the SVM classifier (percentage values over the predicted values)

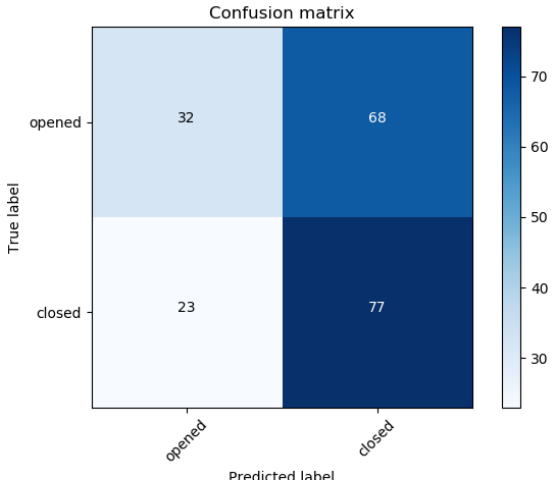


Fig. 7. Normalized confusion matrix for the VGG16 classifier (percentage values over the predicted values)

and intermediate, 81.3% and 77.9% respectively. Regarding the opened mouth class, the Random Forest algorithm stood out with a F-Score of 80.3%. The better results for SVM is consistent with [15] that used SVM to classify opened and closed mouths in humans and with [16] that used SVM to detected closed eyes in humans. We could not find any work directly related to the classification of opened and closed mouth in cattle, so this results can also serve as a baseline for future work.

TABLE I
F-SCORE FOR EACH CLASS AND CLASSIFIER TESTED - EXPERIMENT I (PERCENTAGE VALUES)

Class	SVM	KNN	Adaboost	Random Forests
Opened	78.2	76.6	58.3	**80.3**
Closed	**81.3**	72.0	24.7	78.8
Intermediate	**77.9**	69.6	55.1	76.8
Mean	**79.3**	71.7	43.5	78.2

The ANOVA test produced a p-value equal to 0.0163, indicating a statistically significant difference between the mean F-Score of the classifiers at a 5% significant level. The SVM has been chosen for a further analysis using the normalized confusion matrix shown in Figure 6. The matrix shows that just in 2% of the cases a closed mouth has been incorrectly classified as an opened mouth and in 5% the reverse happened. Most of the classification errors are related to the intermediate class. This may be linked to the difficult, even for humans, to correctly classify the mouth in this intermediate state and suggests that in the future we could rely on a different way to classify the mouth state, maybe concentrating on identifying only when the mouth is opened or closed.

Table II shows the overall results for the deep learning techniques related to the second experiment. VVG16 achieved the highest F-Score of 62.5%. Despite having a higher precision of 69%, ResNet50 presented a much lower recall rate, indicating that the model may be overfitting the training data. The ANOVA test, however, resulted on a p-value equal to 0.0512 which cannot be used to infer any statistically significant difference between the mean F-Score of the classifiers at a 5% significant level.

TABLE II
PERFORMANCE FOR EACH DEEP LEARNING ARCHITECTURE USING 4 DIFFERENT METRICS - EXPERIMENT II (PERCENTAGE VALUES)

Arquit.	Precision	Recall	F-Score
VGG19	57	52	54.4
VGG16	62	**63**	**62.5**
InceptionV3	59	51	54.7
Xception	60	43	50.1
ResNet50	**69**	32	43.7

The VVG16 has been chosen for a further analysis using the normalized confusion matrix shown in Figure 7. The matrix shows that most of the confusions are related to opened mouths being classified as closed (68%). Deep convolutional networks, like those used in this experiment, are known to not perform so well in small datasets [17] and this may be happened in this case. Further studies using data augmentation on the training set may be a future path for exploration.

Figure 8 shows examples of misclassified opened mouth. In the first example (Fig. 8a) we have a high contrast image due to the clear sky and the angle of the camera, turning the mouth very dark and hard to see. The second example (Fig. 8b) shows how close the camera can be when the mouth is opened and showing some feature from bovine papillae.

Figure 9 shows examples of misclassified closed mouth. High contrast and difficult angles are also a problem in these cases. These problems suggest that a more representative dataset should be provided in the future to better train the machine learning algorithms. Another types of cameras and angles should also be tried in future experiments. This baseline experiments used only one of the two cameras and it is

(a) Error Example 1

(b) Error Example 2

Fig. 8. Two opened mouths that have been misclassified as closed

(a) Error Example 3

(b) Error Example 4

Fig. 9. Two closed mouths that have been misclassified as opened

expected that the combination of images from different angles would further improve this first results.

The equipment production cost, considering only the materials used, like the rods, leather covers, connectors, pen-cameras, has been approximately $75.72 (American dollars converted from Brazilian currency on May 2020). The most expensive part being the two pen-cameras, $15.60 each, and the leather-covered rods, $35.96. The costs of the competing devices that use other kind of sensors are not reported in the papers reviewed.

IV. Conclusion

This paper presented a new device to collect videos of animals ruminating at an angle previously considered unprecedented, and that can contribute to identify hidden patterns in animal behavior. The experiments shows a baseline performance that can be improved in the future but already presented some initial results using machine learning techniques that are encouraging, although not optimum, with best F-Score of 79.3% achieved by SVM on a mouth state classification problem. In the future, this device and the information regarding the state of the mouth through time during a longer observation period could be used to estimate rumination parameters important to infer health conditions or to perform experiments with different feeding systems.

Acknowledgment

This work has received financial support from the Dom Bosco Catholic University and the Foundation for the Support and Development of Education, Science and Technology from the State of Mato Grosso do Sul, FUNDECT. Some of the authors have been awarded with Scholarships from the the Brazilian National Council of Technological and Scientific Development, CNPq and the Coordination for the Improvement of Higher Education Personnel, CAPES.

References

[1] E. Tullo, A. Finzi, and M. Guarino, "Review: Environmental impact of livestock farming and precision livestock farming as a mitigation strategy," *Science of The Total Environment*, vol. 650, pp. 2751 – 2760, 2019.

[2] E. Cáceres, H. Pistori, M. Turine, P. Pires, C. Soares, and C. Carromeu, "Computational livestock precision - position paper," in *Second Workshop of the Brazilian Institute for Web Science Research*, 2011.

[3] S. Neethirajan, S. K. Tuteja, S.-T. Huang, and D. Kelton, "Recent advancement in biosensors technology for animal and livestock health management," *Biosensors and Bioelectronics*, vol. 98, pp. 398 – 407, 2017.

[4] D. Lovarelli, J. Bacenetti, and M. Guarino, "A review on dairy cattle farming: Is precision livestock farming the compromise for an environmental, economic and social sustainable production?" *Journal of Cleaner Production*, vol. 262, p. 121409, 2020.

[5] G. Marchesini, D. Mottaran, B. Contiero, E. Schiavon, S. Segato, E. Garbin, S. Tenti, and I. Andrighetto, "Use of rumination and activity data as health status and performance indicators in beef cattle during the early fattening period," *The Veterinary Journal*, vol. 231, pp. 41 – 47, 2018.

[6] U. Braun, S. Zürcher, and M. Hässig, "Evaluation of eating and rumination behaviour in 300 cows of three different breeds using a noseband pressure sensor," *BMC veterinary research*, vol. 11, no. 09, p. 231, 2015.

[7] N. Zehner, C. Umstätter, J. J. Niederhauser, and M. Schick, "System specification and validation of a noseband pressure sensor for measurement of ruminating and eating behavior in stable-fed cows," *Computers and Electronics in Agriculture*, vol. 136, pp. 31 – 41, 2017.

[8] N. Ali, D. Neagu, and P. Trundle, "Evaluation of k-nearest neighbour classifier performance for heterogeneous data sets," *SN Applied Sciences*, vol. 1, no. 1559, 2019.

[9] J. Nalepa and M. Kawulok, "Selecting training sets for support vector machines: a review," *Artificial Intelligence Review*, vol. 52, pp. 857 – 900, 2019.

[10] J. A. Wyner, M. Olson, J. Bleich, and D. Mease, "Explaining the success of adaboost and random forests as interpolating classifiers," *Journal of Machine Learning Research*, vol. 18, 2017.

[11] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[13] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826.

[14] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1800–1807.

[15] C. Bouvier, A. Benoit, A. Caplier, and P.-Y. Coulon, "Open or closed mouth state detection: Static supervised classification based on log-polar signature," in *Proceedings of the 10th International Conference on Advanced Concepts for Intelligent Vision Systems*, vol. 5259, 10 2008.

[16] C. Souto Maior, M. Moura, J. Santana, L. Nascimento, J. Macedo, I. Lins, and E. Droguett, "Real-time svm classification for drowsiness detection using eye aspect ratio," in *Proceedings of Probabilistic Safety Assessment and Management PSAM*, 09 2018.

[17] J. G. A. Barbedo, "Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification," *Compute rs and Electronics in Agriculture*, vol. 153, pp. 46 – 53, 2018.

Classification of UAVs' distorted images using Convolutional Neural Networks

Leandro H. F. P. Silva*, Jocival D. D. Júnior*, Jean Fabrico Batista Santos*, João F. Mari†,
Mauricio C. Escarpinati*, André R. Backes*,
*School of Computer Science, Federal University of Uberlândia, Brazil
†Federal University of Viçosa, Brazil
arbackes@yahoo.com.br

***Abstract*—Currently, the use of unmanned aerial vehicles (UAVs) is becoming ever more common for acquiring images in precision agriculture, either to identify characteristics of interest or to estimate plantations. However, despite this growth, their processing usually requires specialized techniques and software. During flight, UAVs may undergo some variations, such as wind interference and small altitude variations, which directly influence the captured images. In order to address this problem, we proposed a Convolutional Neural Network (CNN) architecture for the classification of three linear distortions common in UAV flight: rotation, translation and perspective transformations. To train and test our CNN, we used two mosaics that were divided into smaller individual images and then artificially distorted. Results demonstrate the potential of CNNs for solving possible distortions caused in the images during UAV flight. Therefore this becomes a promising area of exploration.**

***Index Terms*—Convolutional Neural Networks, Precision Agriculture, Unmanned Aerial Vehicle, Linear Distortions, Image Processing**

I. Introduction

At the end of the 19th century, studies already indicated concern about the growth of the world population and the capacity of the planet to produce food to feed it [1]. At the time it was feared that the population would grow in geometric progression, while food production would grow in arithmetic progression. In the end, this would lead to a drastic food shortage and, as a consequence, hunger. Therefore, inevitably population growth should be controlled.

These predictions were not confirmed, largely due to the significant technological advances that occurred in the agricultural area between 1950 and the late 1960s [2], a set of research technology transfer initiatives that increased agricultural production worldwide, particularly in the developing world, beginning most markedly in the late 1960s. The initiatives resulted in the adoption of new technologies, including high-yielding varieties (HYVs) of cereals, especially dwarf wheats and rices, in association with chemical fertilizers and agro-chemicals, and with controlled water-supply (usually involving irrigation) and new methods of cultivation, including mechanization. All of these aspects mentioned were seen as a kind of "practice package" to replace "traditional" technology and thus being used as a whole [3].

Nowadays, we are seeing a new evolutionary phase in the field of agriculture researches. The main component of this phase is Precision Agriculture (PA), which is nothing more than a farming management concept based on observing, measuring and responding to inter and intra-field variability in crops. The goal of precision agriculture research is to define a decision support system (DSS) for whole-farm management with the goal of optimizing returns on inputs while preserving resources [4].

Dealing specifically with problems involving the PA area, it has shown itself to be heavily dependent on imaging and mapping technologies e.g. for estimating growth [5], or identifying other important agronomic characteristics [6] such as nitrogen stress. Advances in Unmanned Aerial Vehicles - UAV - technology led to its widespread popularization. With the corresponding drop in operational costs even smaller plantations are now able to afford the usage of imaging aided technologies. The latest economic report by the Association of Unmanned Aerial Vehicle International [7] points out the agricultural market is by far the largest segment for UAVs. In the United States alone is forecast to create thousands of new jobs and considerable revenue and taxes. With the growth of this market production costs are expected to drop. It, in turn, will allow smaller enterprises such as family and small agricultural cooperatives [8] to benefit from the diminished operational costs to also make use of precision agriculture aided by UAVs. Other countries like Japan are also making extensive use of UAVs in agriculture and in Brazil there is a growing number of startup companies producing and commercializing UAVs.

Different from all other aerial image acquisition devices, such as satellites and large aircraft, UAV's allow images to be captured at low and medium altitudes (50 to 400 m), providing a more detailed view of the region to be observed. Another important element for the effectiveness of the analysis performed with this equipment is the used sensors. There is a wide range of devices used in the process: RGB cameras; heat capture sensors, multi and hyperspectral cameras, among others. Each device, with its characteristics, produces information that leads to different types of analysis. However, the process of data acquisition, in general, is the same independent of the sensor used: the equipment is coupled to the aircraft and the images are sequentially captured during the flight. After finishing the process, with the aircraft already on the ground, these images are organized into a mosaic to represent the entire area.

As the images are taken during the flight, UAVs may

undergo some variations, such as wind interference and small altitude variations, which directly influence the captured images, causing a natural misalignment among the imagens that comprise the mosaic and, more often, among the different spectra witch form a specific frame. Usually, the distortion generated in this process are classified as linear distortion and can affect significantly the success of specific software used in agriculture images. Thus, in order to address this problem, the present work proposes a Convolutional Neural Network (CNN) trained for the classification of three linear distortions common in UAV flight: rotation, translation and perspective transformations.

The remainder of this paper is organized as follows. Section II shows some recent papers published in the area. In Section III we detail the problem and their implications. In Section IV, we present an overview of the CNN and how it was used to deal with our problem. Section V presents the image dataset used in the experiments. Sections VI and VII present the experiments and a discussion of the results. Section VIII presents the conclusions and future work.

II. RELATED WORK

In [9] the authors evaluated different techniques for obtaining control points in multispectral images of soy plantations obtained by UAVs. The authors also investigated whether the combination of characteristics derived from different techniques generates better results than when those techniques are used individually. The paper evaluated three detection algorithms with different characteristics (KAZE, MEF, and BRISK) and their combinations. Results show that KAZE techniques have the best results.

In [10] the authors presented a convolutional neural network to estimate homography from a pair of images. The network in question has 10 layers with feed-forward architecture and receives a pair of grayscale images. Subsequently, it produces a homography with 8 degrees of freedom, which can be used to map the pixels from the first to the second image.

The work in [11] introduces a hierarchical approach based on Siamese convolutional neural networks to estimate homography between two images. The networks are stacked sequentially to estimate of error limits. In each convolutional network module, the resources of each image are extracted independently, generating a shared set of kernels, which is known as the Siamese model. Subsequently, the image pairs are merged to estimate the homography. With this approach, the results show that through deep learning it is possible to estimate homography from an image pair.

III. PROBLEM DEFINITION

Due to the inherent aspects of UAV flight, image capture is subject to distortion that needs to be dealt with and corrected. These distortions may be linear or nonlinear. In this paper, we will consider only three linear distortions that may occur during flight: translation, rotation and perspective transformation.

In a translation operation all points are moved in a straight line in the same direction. In summary, a conversion operator will perform a geometric transformation that maps the position of each element of the image in an input image to a new position in the output image [12].

Rotation transformation is defined as a rotary movement on a fixed axis. According to Gonzalez (2002) [12], three transformations are needed to rotate a point relative to another arbitrary point in space: the first will translate the arbitrary point to the origin, the second will rotate, and finally the third will translate the point back to its original position.

A perspective transformation in general takes place with the conversion of the 3D world into a 2D image. This is the same principle that human vision works on and the same principle that the camera works on. In perspective projections, parallel lines converge (in 1, 2, or 3 axes) for a given point. This way, objects that are farther away are smaller than closer objects.

Perspective transformation will project three-dimensional points onto a plane. Such transformations play a fundamental role in image processing, as they offer a way of approximating the way in which the image is formed by looking at the three-dimensional world [12]. In general, these projective transformations allow us to capture natural motion dynamics through a mathematical mechanism. These transformations do not preserve size or angle but preserve incidence and cross-ratio.

IV. CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Networks (CNN) are a category of deep learning algorithms capable to mimic the human learning process. These networks are based on the concept of the receptive field from biological systems, which gives these networks the ability to learn different filters and characteristics from an image. This way, CNN can explore the spatial correlations among pixels in an image in order to extract image attributes that are relevant for different tasks, such as image classification and segmentation [13]–[15]. Most CNN models available in the literature are defined in terms of three types of layers, which are differently combined to improve image classification or segmentation: convolutional, pooling and fully connected layer. In the sequence, we present a brief description of each layer.

The convolutional layer is responsible for extracting meaningful attributes from an image. To accomplish that, it applies a series of convolution operations to the input data, which acts as receptive filters that highlight different attributes of a local region of the image. In general, theses filters are defined as kernels size 3×3 or 5×5. Additionally, the activation function ReLU (REctified Linear Unit) and a Batch Normalization operation are applied to the result of the convolutional layer. This helps to speed up the training of the network and to improve its results [16].

The convolutional layer is usually followed by a pooling layer. The main purpose of this layer is to reduce the feature maps computed by the previous layers, thus reducing the network sensitivity to distortions in the image and data shifting. In general, it is used a pooling mask of size 2×2, thus reducing

a 4 pixels region to a single value according to some criteria (e.g., maximum or the average pixel of the region) [17].

At the end of the CNN, we find the fully connected (or dense) layer. About 90% of the parameters of a CNN are found in these layers. This layer receives as input data the 2D features maps obtained from previous layers and its main goal is to learn a 1D feature vector capable to discriminate the input image. This feature vector is used as the input of a softmax classifier, which returns the most probable class for a given input image.

V. IMAGE DATASET

A. Selected Images

For our experiments, we considered two mosaics of images acquired using an unmanned aerial vehicle (UAVs) to create the datasets of images used in the experiments. These mosaics have 18543×2635 and 8449×11180 pixels size, respectively.

For each dataset, we selected grayscale patches of 150×150 pixels size. Subsequently, we discard patches that have little (or any) significant visual information. This was determined by the number of pixels (n) with a value of 0 in the patch. Thus, if $n < 20$, the patch is considered for the composition of the dataset; otherwise, the patch is discarded. Therefore, we built two datasets, which we will call DS1 and DS2 and which have, respectively, 3218 and 1586 images. Figure 1 illustrates two examples of images patches generated for each dataset.

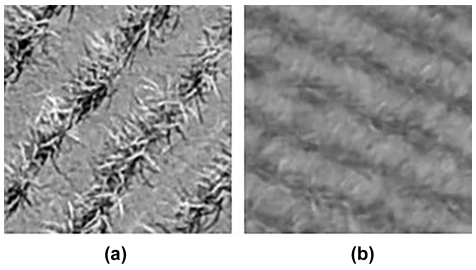


Fig. 1. Example of images that make up both datasets: (a) an image of DS1; (b) an image of DS2.

B. Dataset images distortions

For both datasets, DS1 and DS2, we artificially distorted the images using two affine (rotation and translation) and one projective (perspective) transformation. It is important to mention that, as a result of the transformation method, the transformed images have black areas, especially at the limits of the image area, which can directly influence the neural network training and testing (see in Figure 2). To avoid these black areas, we cropped a 64×64 pixels region aligned with the center of the image, thus removing any artifact added to the image by the selected transformation method.

In order to apply the transformation over the images, the following set of parameters were used:

- **Rotation:** we used $\theta = \{0°, 5°, 10°, 15°\}$, thus generating 4 classes of rotated patterns.

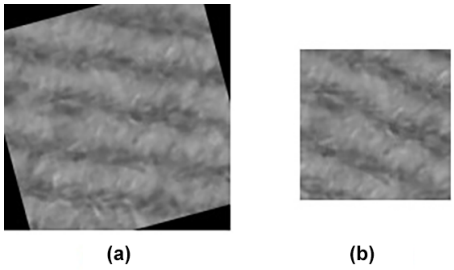


Fig. 2. (a) Image after a 15-degree rotation transformation. Notice that this image presents black areas which can directly influence the neural network training and testing; (b) Cropped region with 64×64 pixels size.

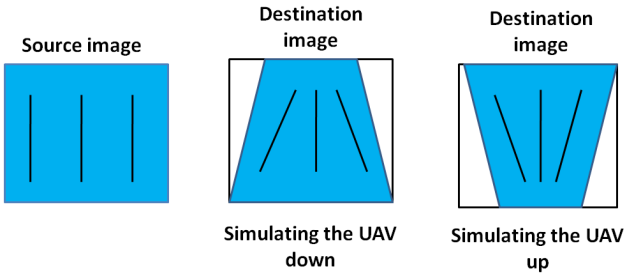


Fig. 3. Perspective transformation in UAV up and down simulations.

- **Translation:** images were translated by 25 pixels in 4 possible directions: right and top; right and down; left and top; and left and down, thus generating 5 equivalence classes (the original image is also included).
- **Perspective:** To simulate UAV up and down possibilities in moments of image capture, we also deal with perspective transformation. The Figure 3 illustrates the UAV up and down simulations and the respective distortions caused by pitch variations. For this transformation, we generated two variations for each of the two possibilities mentioned above, thus totaling 5 equivalence classes (the original image is also included). In this way, we choose four control points in a source image to map it to a destination image. Perspective transformation works with the row and column relationship. As we are only simulating the UAV up and down possibilities, we keep the proportion of lines identical to the original image. For the columns, the proportions in each of the distorted classes created were: (0.05, 0.66); (0.05, 0.77); (0.02, 0.66); (0.02, 0.77).

It is also necessary to define a mathematical operation that relates the distorted image to the base image, otherwise it is impossible to state that an image is distorted. Thus, all artificially distorted images underwent a subtraction operation from the original image. Let A be the distortion-free image and B the distorted image relative to A, we define X as the image resulting from the subtraction operation and to be processed by the CNN. The operation performed between A

and B is defined pixel by pixel. We must also consider that the subtraction operation may result in negative values and an image is expected to have only positive values. To avoid that, we normalized the computed x_{ij} values as follows:

$$x_{ij} = max(b_{ij} - a_{ij}, 0) \quad (1)$$

where $a_{ij} \in A$ represents a pixel of image A, $b_{ij} \in B$ represents a pixel of image B and $x_{ij} \in X$ represents a pixel of image X. Figure 4 illustrates a subtraction between an artificially distorted image (rotation) and a distortion-free image.

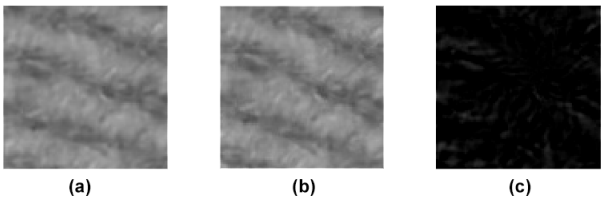


Fig. 4. Example of subtraction operation between two images: (a) Artificially distorted image (rotation); (b) Corresponding distortion-free image; (c) Result from the subtraction operation ($(c) = (a) - (b)$).

VI. EXPERIMENTS

We also carried out a data augmentation to reduce the possibility of overfitting in our experiments. In addition to the traditional CNNs, we proposed an alternative architecture that will be presented as follows. Our architecture is motivated by [18], [19], where simpler CNNs and sets of filters were used to solve less complex classification problems.

In order to address our image analysis problem, we proposed a network structure. Due to the reduced size of our samples (64×64 pixels size), our CNN presents fewer layers than conventional CNNs. To properly process our images we used a CNN with 5 convolutional layers. Each convolutional layer presents, respectively, 32, 64, 64, 128 and 256 filters. To improve the network performance and to speed up its training, we apply non-linearity ReLU activation function after each convolutional layer. We also apply a batch normalization after the ReLU filter, which is followed by a 2×2 max-pooling layer.

After the convolutional layers, we use the resulting volume ($2 \times 2 \times 256$ output shape and 1024 features) as input for the dense layers. The first and second dense layers have 128 neurons and the activation function ReLU. After each dense layer we applied dropout of 20%. Finally, the output layer has 4 or 5 neurons (4 for rotation; otherwise, 5 neurons) that determined the class, as we expound in subsection V-B.

To implement the convolutional neural networks used in this work we used the Python version of Tensorflow, an open-source library developed by Google [20] for efficient building, training and use of deep neural network models. TensorFlow is based on tensors and dataflow graphs. Tensors are numerical multidimensional arrays that represent the data. Dataflow graphs nodes represent operations while edges describe the flow of data throughout the processing steps. TensorFlow dataflow graphs are very modular and allow building complex models directly. These models can be trained and run in a myriad of environments taking advantage of the high parallelism of modern GPUs [21]–[23].

We evaluated our CNN model using both datasets, as defined in Section V. For each dataset we selected 75% of the samples to compose the training set, while the remaning images were used for validation. Motivated by work [24], we chose not to perform cross-validation for this purpose. The work [24] demonstrates that in problems in this context, the use of cross-validation does not generate much difference in the final results, except that it increases the computational cost considerably. Both datasets will be available for replication and other experiments as request.

Experiments were conducted on a Personal Computer with Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz, 32GB RAM, 64-bit Windows OS and GPU NVIDIA GeForce GTX 1050 Ti, 4GB GDDR5. We also used Python 3.6 and Keras 2.1.6-tf with TensorFlow 1.10.0 and CUDA Toolkit 9.0 to implement and test the experiments.

VII. RESULTS

First, for each dataset (DS1 and DS2) we generated a new dataset with one of the specified distortions. Then, this new dataset was split between training and validation samples and used to train our CNN model for 20 epochs. After this, we are able to analyze the accuracy of our model for detecting the distortions analyzed.

We notice that the best performance is obtained when dealing with the problem of image rotation, as illustrated in Figure 5(a). For the rotation problem, our CNN model is capable to classify the rotation distortion with 99.85% and 99.18% accuracy in the DS1 and DS2 datasets, respectively. Moreover, the CNN presents a good ability to generalize the features learned in the training set to the test set. This may be explained by the fact that the rotation operation results in a less distorted image in comparison to other image transformations, i.e., an easier classification problem.

Figure 5(c) shows the performance for perspective transformation. For this transformation, our CNN model achieves high accuracy, especially for DS1, which reached 95.50%. For the same problem, however, we obtained only 91.47% for DS2 accuracy. Notice that this accuracy is substantially lower when compared to the rotation experiment. One explanation for this behavior is that this kind of transformation affects differently the regions of the sample, while rotation affects all points of the sample equally. Moreover, although both datasets present a lower result, we notice that dataset DS1 presents a superior result when compared to dataset DS2. It may be the case that the number of samples in the training set, which is larger in DS1, contributes positively to learn this transformation.

For the translation transformation, we noticed a considerable drop in the accuracy of the network when evaluating the DS2 dataset (70.24%), as shown in Figure 5(b). Even though the dataset DS1 (Figure 5(b)) also present an inferior performance

when compared to the rotation transformation, its result is superior to the ones obtained for the perspective transformation. This result observed in the DS2 dataset is probably explained by the lack of details in their original images, as illustrated in Figure 1. Since crop lines and land regions present similar gray-level distributions, the result of the subtraction operation between the original and the translated image results in a mostly black image, i.e., an image without enough attributes for our CNN to learn.

In order to improve the evaluation of our CNN model we compared its results with the ones obtained by 4 traditional CNN models: InceptionV3 [25], ResNet [26], SqueezeNet [27] and VGG-16 [28]. For this comparison we used pre-trained networks on the 2012 ImageNet dataset and fine-tuned the whole CNN to our classification problem for 20 epochs. We must emphasize that these networks have a input size larger than the samples in our datasets so that all images have been scaled up to fit the input size of the respective network.

Table I summarizes the results of all CNN models. As we can see, our CNN surpasses the results of all compared ones, indicating that its architecture, although simpler than the compared ones (see Table II), is more effective to classify images obtained from the difference of intensities between two images and, therefore, presenting a small variation of gray levels.

TABLE I
ACCURACY (%) OBTAINED FOR OUR CNN AND THE COMPARED ONES.

	Translation		Rotation		Perspective	
CNN model	DS1	DS2	DS1	DS2	DS1	DS2
ResNet	91.83	48.13	95.00	96.84	59.55	62.63
InceptionV3	20.00	60.10	98.48	98.23	20.00	65.96
VGG-16	94.76	65.15	98.63	98.74	84.89	75.20
SqueezeNet	90.51	40.40	91.77	96.15	55.68	55.20
Proposed	96.92	70.24	99.85	99.18	95.50	91.47

TABLE II
NUMBER OF PARAMETERS OF EACH CNN MODEL.

CNN model	# of parameters
ResNet	23,595,908
InceptionV3	22,082,084
VGG-16	14,797,380
SqueezeNet	725,061
Proposed	477,573

VIII. CONCLUSION

In this paper, we addressed the problem of classifying different types of distortions in images acquired using unmanned aerial vehicles (UAVs). To accomplish that we proposed and trained a Convolutional Neural Network (CNN) model to learn the subtleties that distinguish each transformation studied: translation, rotation and perspective transformation.

Results showed that our CNN model is capable to correctly classify the different transformations, especially the rotation transformation. However, the performance of the CNN is dependent on the image resolution and gray-levels distributions present in the sample image evaluated so that datasets containing blurry images affects negatively the performance of our network. Also, our architecture, due to its low computational cost, can inspire embedded systems to UAVs in the context of precision agriculture, reducing financial costs inherent to the process. As future work, we intend to expand the dataset used in the experiments and to include images containing real distortions produced during a UAV flight and to explore other models of CNN.

ACKNOWLEDGMENT

André R. Backes gratefully acknowledges the financial support of CNPq (National Council for Scientific and Technological Development, Brazil) (Grant #301715/2018-1). This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) - Finance Code 001. The authors would like to thank the company Sensix Inovações em Drones Ltda (http://sensix.com.br) for providing the images used in the tests.

REFERENCES

[1] T. R. Malthus, *An Essay on the Principle of Population..*, 1872.
[2] P. B. Hazell, *The Asian green revolution*. Intl Food Policy Res Inst, 2009, vol. 911.
[3] B. Farmer, "Perspectives on the 'green revolution'in south asia," *Modern Asian Studies*, vol. 20, no. 1, pp. 175–199, 1986.
[4] A. Milella, G. Reina, and M. Nielsen, "A multi-sensor robotic platform for ground mapping and estimation beyond the visible spectrum," *Precision agriculture*, vol. 20, no. 2, pp. 423–444, 2019.
[5] T. Kataoka, T. Kaneko, H. Okamoto, and S. Hata, "Crop growth estimation system using machine vision," in *Proceedings 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, vol. 2. IEEE, 2003, pp. b1079–b1083.
[6] S. Sankaran, L. R. Khot, C. Z. Espinoza, S. Jarolmasjed, V. R. Sathuvalli, G. J. Vandemark, P. N. Miklas, A. H. Carter, M. O. Pumphrey, N. R. Knowles *et al.*, "Low-altitude, high-resolution aerial imaging systems for row and field crop phenotyping: A review," *European Journal of Agronomy*, vol. 70, pp. 112–123, 2015.
[7] D. Jenkins and B. Vasigh, *The economic impact of unmanned aircraft systems integration in the United States*. Association for Unmanned Vehicle Systems International (AUVSI), 2013.
[8] J. M. Turner, "Economic potential of unmanned aircraft in agricultural and rural electric cooperatives," Ph.D. dissertation, 2016.
[9] J. D. D. Junior, A. R. Backes, and M. C. Escarpinati, "Detection of control points for uav-multispectral sensed data registration through the combining of feature descriptors," 2019.
[10] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Deep image homography estimation," *arXiv preprint arXiv:1606.03798*, 2016.
[11] F. Erlik Nowruzi, R. Laganiere, and N. Japkowicz, "Homography estimation from image pairs with hierarchical convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 913–920.
[12] R. C. Gonzalez, R. E. Woods *et al.*, "Digital image processing," 2002.
[13] Y. L. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[14] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, "Deep learning for visual understanding: A review," *Neurocomputing*, vol. 187, pp. 27–48, 2016.
[15] M. A. Ponti, L. S. F. Ribeiro, T. S. Nazaré, T. Bui, and J. Collomosse, "Everything you wanted to know about deep learning for computer vision but were afraid to ask," in *SIBGRAPI Tutorials*. IEEE Computer Society, 2017, pp. 17–41.
[16] Y. LeCun, Y. Bengio, and G. E. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

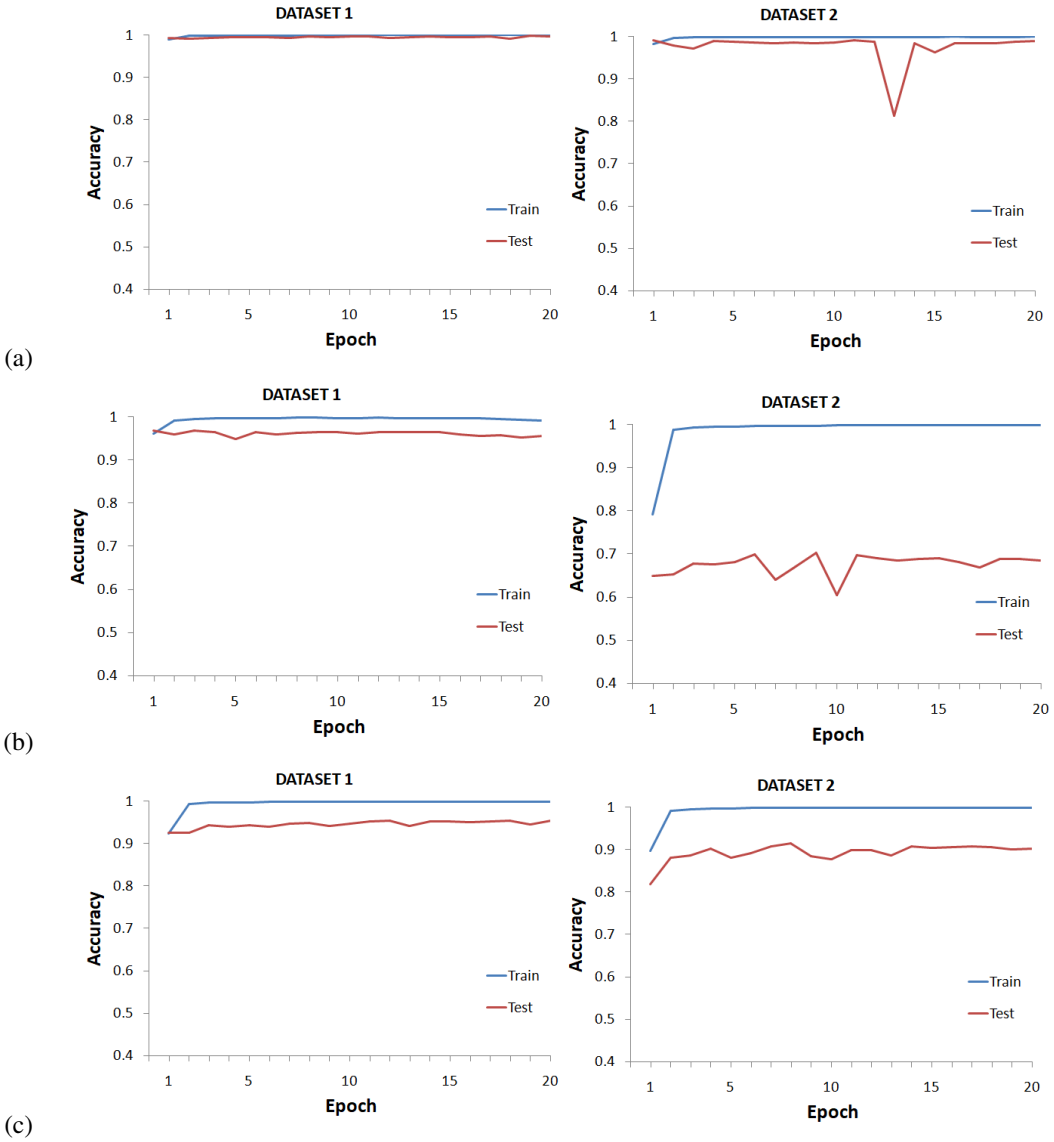


Fig. 5. Accuracy of our CNN to classify distortions in both datasets: (a) Rotation; (b) Translation; (c) Perspective.

[17] D. Scherer, A. C. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *Artificial Neural Networks - ICANN 2010 - 20th International Conference, Thessaloniki, Greece, September 15-18, 2010, Proceedings, Part III*, ser. Lecture Notes in Computer Science, vol. 6354. Springer, 2010, pp. 92–101.

[18] A. P. Marcos, N. L. S. Rodovalho, and A. R. Backes, "Coffee leaf rust detection using genetic algorithm," in *2019 XV Workshop de Visão Computacional (WVC)*. IEEE, 2019, pp. 16–20.

[19] ——, "Coffee leaf rust detection using convolutional neural network," in *2019 XV Workshop de Visão Computacional (WVC)*. IEEE, 2019, pp. 38–42.

[20] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," 2016.

[21] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2019.

[22] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[23] T. Hope, Y. S. Resheff, and I. Lieder, *Learning tensorflow: A guide to building deep learning systems*. " O'Reilly Media, Inc.", 2017.

[24] A. R. de Geus, A. R. Backes, and J. R. Souza, "Variability evaluation of cnns using cross-validation on viruses images." in *VISIGRAPP (4: VISAPP)*, 2020, pp. 626–632.

[25] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[27] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size," *arXiv:1602.07360*, 2016.

[28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

Maize leaf disease classification using convolutional neural networks and hyperparameter optimization

Erik Lucas da Rocha, Larissa Ferreira Rodrigues, João Fernando Mari
Instituto de Ciências Exatas e Tecnológicas
Universidade Federal de Viçosa - UFV
Caixa Postal 22 - 38.810-000 - Rio Paranaíba - MG - Brasil
Email: {erik.rocha, larissa.f.rodrigues, joaof.mari}@ufv.br

***Abstract*—Maize is an important food crop in the world, but several diseases affect the quality and quantity of agricultural production. Identifying these diseases is a very subjective and time-consuming task. The use of computer vision techniques allows automatizing this task and is essential in agricultural applications. In this study, we assess the performance of three state-of-the-art convolutional neural network architectures to classify maize leaf diseases. We apply enhancement methods such as Bayesian hyperparameter optimization, data augmentation, and fine-tuning strategies. We evaluate these CNNs on the maize leaf images from PlantVillage dataset, and all experiments were validated using a five-fold cross-validation procedure over the training and test sets. Our findings include the correlation between the maize leaf classes and the impact of data augmentation in pre-trained models. The results show that maize leaf disease classification reached 97% of accuracy for all CNNs models evaluated. Also, our approach provides new perspectives for the identification of leaf diseases based on computer vision strategies.**

***Keywords*—Convolutional neural networks; maize leaf; classification; data augmentation; hyperparameter; Bayesian optimization.**

I. Introduction

By 2050 the number of people worldwide is expected to be almost 10 billion, driving the farm and food system. However, agricultural production has several limitations related to the degradation of agricultural land, water resources, climate change, and food losses [1].

Maize, popularly known as "corn", is the most produced food crop in the world, exceeding wheat and rice production [2]. Also, maize is a primary food used in several industry sectors to produce food, beverage, and cattle feed. Recently, the number of maize diseases and the degree of harm they cause have increased, mainly due to the degradation of agricultural land and changes in cultivation systems. Among the various diseases that affect maize plantations, leaf disease is one of the most critical and causes scaling down the crop yield and food nutritional value [3].

Visual analysis of patterns in leaves is the procedure used to identify leaf diseases in maize crops, but this process is very subjective and time-consuming. Moreover, the inaccurate identification of maize leaf diseases may lead to the wrong usage of pesticides, which reduces the quality and quantity of maize production, as well as health problems in humans [4].

The most promising technique for overcoming these limitations is the development of automatic systems based on computer vision to reduce losses and increase productivity [5]. Also, these techniques are financially attractive, especially for farms in emerging countries.

With advances in computational resources, deep learning models significantly outperform approaches based on handcrafted features. In particular, Convolutional Neural Networks (CNN) provide automatic feature extraction from input images and demonstrate effective results in visual recognition tasks [6], [7]. Thus, CNNs can be used to identify maize crop diseases in the early stages, which can help improve the accuracy of plant protection and expand the use of technology in precision agriculture.

This paper identifies a suitable method based on CNNs for automatically classifying maize leaf diseases. Its main contributions are: (i) a comparison of the performance of three state-of-the-art CNN architectures in terms of accuracy, precision, recall, and F1-score; (ii) exploration of these CNNs with fine-tuning training; (iii) use of data augmentation strategies based on random rotations, vertical and horizontal flips, to overcome imbalance between the classes of the dataset.

The main novelty of this study is to find a suitable setup for hyperparameter optimization using Bayesian optimization, as finding the optimal hyperparameters to train CNN architectures is challenging due to the fact that there is no optimum method for the selection of hyperparameters. Also, the Bayesian optimization technique finds the best possible parameter setup faster than grid and random search.

To the best of our knowledge, no other study in the literature realizes such hyperparameter optimization considering different CNN architectures to classify maize leaf diseases. Our results suggest that hyperparameter optimization combined with fine-tuning training tends to be the best performing strategy to classify maize leaf diseases. In fact, our best result achieves an accuracy of 97%, which is very close to the highest accuracy score presented in the literature.

The remaining of this paper is organized as follows. Section II introduces the related work. Section III describes the material and methods. Section IV indicates and discusses the results. Section V presents conclusions and future work.

II. Related Work

Considerable efforts have been dedicated to the development of automatic systems based on computer vision for crop disease identification. Mohanty et al. [5] evaluated two CNN architectures to classify 14 crop species and 26 leaf diseases

from PlantVillage dataset, obtaining 99.35% of accuracy. The same approach was adopted by Sladojevic et al. [8] to identify plant disease from healthy leaves. However, they considered leaf image takes from several datasets and achieved an average of 96.3% accuracy on their experimental analysis. Too et al. [9] performed a comparative analysis to classify 38 categories of plant disease with different pre-trained CNN models and achieved 99.75% of accuracy.

When considering maize leaf disease classification, DeChant et al. [10] proposed an automatic identification of northern leaf blight of maize and achieved 96.7% of accuracy. Zhang et al. [11] improved two CNNs architectures and applied data augmentation technique to classify eight kinds of maize leaf diseases. Lin et al. [12] designed a multi-channel CNN to classify five types of maize diseases using images takes from Shandong Province farming area. Alehegn et al. [13] developed a technique based on color, texture, and morphological features to classify maize leaf diseases taken from Ethiopia farming areas.

Bhatt et al. [14] proposed an approach based on CNNs and adaptive boosting with decision tree-based classifier to classify corn leaf diseases. The model developed by [14] reached an accuracy of 90% using Inception-v2 with Random Forest and the accuracy was improved to 98% using AdaBoost.

Priyadharshini et al. [15], also considered the same dataset considered in this paper. They proposed a method for the maize leaf disease classification using a CNN that shares the basic architecture of LeNet-5, all images were preprocessed using PCA whitening, and achieved an accuracy of 97.89%. Sibiya & Sumbwanyambe [16] proposed a system based on CNN to classify maize leaf diseases and obtained 92.85% of accuracy. Hu et al. [17] tested a model based on pre-trained GoogLeNet to classify maize leaf disease and obtained 97.60% of accuracy.

Recently, Waheed et al. [18] presented an optimized DenseNet to classify corn leaf disease. The authors used grid search to find the optimal hyperparameter values and the model was trained using different sets of hyperparameters. However, the grid search algorithm may have problems such as the curse of dimensionality, and lack of resources to handle the time-consuming operations [19]. In this paper, we used Bayesian optimization, which allows obtaining better results in fewer evaluations compared to grid search [20] [21].

In contrast to all previous works, in this work we consider hyperparameter optimization, data augmentation and training based on fine-tuning.

III. Material and Methods

The main purpose of this work is to provide a method able to classify maize leaf disease images using CNNs improved by Bayesian hyperparameter optimization. Fig. 1 illustrates the steps of the proposed method. It is composed of three main stages: a) the dataset splitting in training, validation and testing sets; b) k-fold cross-validation training of the CNNs architectures using Bayesian optimization for hyperparameters selection; and c) decision-making of the models with the testing phase.

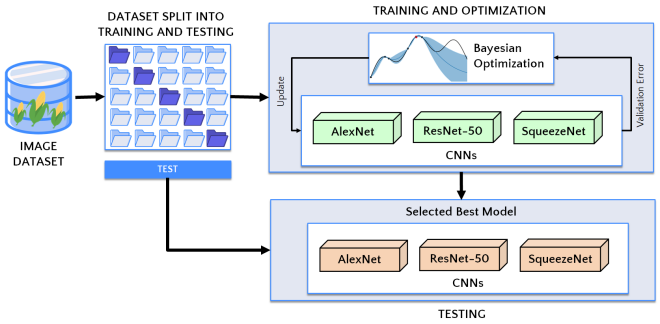


Fig. 1. Steps of proposed method.

A. Image dataset

The images used were taken from PlantVillage [1] dataset [5]. It contains 3852 images of maize leaf, each with a single leaf in evidence, categorized into one of four classes: gray leaf spot (513 images), common rust (1192 images), northern leaf blight (985 images), and healthy (1162 images). To illustrate the dataset, samples from each class are presented in Fig. 2.

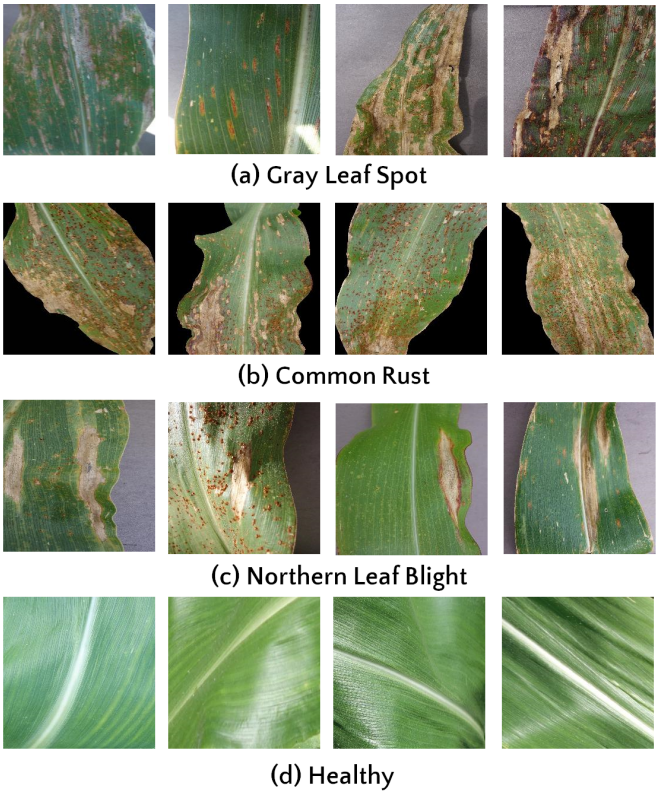


Fig. 2. Examples of images for each class.

B. Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are the state of the art in image classification tasks, were designed to extract visual patterns directly from input images generating feature maps to the next deep layer [6] [7]. In this paper, we tested

[1] Available in: https://github.com/spMohanty/PlantVillage-Dataset

three CNN architectures: AlexNet [22], ResNet-50 [23], and SqueezeNet [24].

AlexNet [22] won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012. It is composed of five convolutional layers, three max-pooling layers, and two fully connected layers. Also, this architecture applies dropout regularization to reduce overfitting and ReLU activation to accelerate the training.

ResNet was proposed by He et al. [23] and won the ILSVRC 2015. It is utilizes residual blocks to address the gradient degradation in the training step. The different versions of ResNet have 18,50, 101, 152, and 201 layers. In the present study, we use the ResNet with 50 layers: one convolutional layer of size 7×7, followed by several convolutional layers of size 3×3 and 1×1.

SqueezeNet [24] architecture requires fewer parameters and provides performance equivalent to AlexNet. SqueezeNet is composed mainly of fire blocks that are squeeze convolution layers of size 1×1, which goes into two expand layers, one with a filter size of 1×1 and the other has a size of 3×3. The feature maps obtained from the two expand layers feed into a concatenation layer, being input to the squeeze layer that outputs to the following fire module.

C. Hyperparameter Optimization

Hyperparameters are essential in deep learning algorithms since those parameters define the details of training and affect the performance of the models significantly [25]. The choice of values for the hyperparameters is a crucial task as there is no optimum method for the selection of hyperparameters. The choice of hyperparameters values is represented as an optimization problem, where the objective function is unknown (it is a black-box function) and the hyperparameters are defined as decision variables. The fine-tuned hyperparameters in this paper are as follows:

- Batch size: the batch size is the number of images that will be propagated through the CNN. A large batch size requires less RAM and GPU consumption but could result in a less accurate estimate of the gradient. On the other hand, a smaller batch size requires more RAM and GPU consumption, and fewer groups (batches) will propagate on CNN [26].
- Learning rate: the learning rate defines the level of adjustments of weight connections and network topology, applied at each training epoch, being the main parameter to tuning. This hyperparameter is optimized in order to improve the runtime when using Stochastic Gradient Descent (SGD) optimizer. A high learning rate may sacrifice the accuracy generating a lack of precision. On the other hand, a small learning rate requires more epochs of training to learn the difference between features [25].
- Momentum: the momentum coefficient is necessary to stop the oscillations in the regions of high curvature of the loss function generated by the SGD optimizer [7].

We employed the optimization in order to minimize the objective function, i.e., the cross-entropy function (loss function). The loss function was minimized by SGD optimizer with momentum.

D. Bayesian Optimization

Bayesian optimization [27] is an efficient algorithm composed by four parts: i) an objective function that defines what will be optimized; ii) the performance of the model that varies according to the hyperparameters setup; iii) the hyperparameter space search, which is a list of possible solutions; and iv) optimizer algorithm [20].

The objective function is unknown and is only defined after the setup definition, allocating, training, and testing the model. Therefore, the method considers the objective function as a random function. The Bayesian optimization is called Bayesian because the optimization strategy uses the Bayes' theorem. In this context, given the evidence provided by data D, the posterior probability $P(m|D)$ where m is a model proportional to the probability $P(D|m)$ of overserving D given model m multiplied by the prior probability of $P(m)$, defined in Equation 1.

$$P(m|D) = \frac{P(D|m)P(m)}{P(D)} \tag{1}$$

In this study, we used Gaussian Process (GP) as the probabilistic model [28]. GP is a model that generates data located throughout some domain (here, the hyperparameters). This method performs a hypothesis about unknown parameters, based on parameters already found. The choices of the Bayesian approach increasing the time for selecting hyperparameters but reduces the time need in the evaluation of the objective function, requiring a less computational cost. Therefore, Bayesian Optimization is high efficiency compared to grid and random searches [20] [21].

E. Models Training

The classification validation was performed by stratified k-fold cross-validation [29]. The dataset was randomly partitioned into six stratified sub-datasets. Of the six sub-datasets, five sub-datasets were used for cross-validation, and a single sub-dataset is retained as the test data for testing the model. The cross-validation process is then repeated five times, with each of the k sub-datasets used exactly once as the validation data. The five validation results were averaged to identify the best model. After identifying the best model, we tested this model using the single sub-dataset retained as the test data.

Fig. 3 shows the overall process of selecting optimal hyperparameters. First, we separate the maize leaf disease dataset into learning and test data. After, the learning data is separated into training and validation data, and k-fold cross-validation based on a Bayesian optimization is carried. It performs the verification of the model on the basis of the preset hyperparameter and the k-fold value. Finally, the model is trained by using the optimal hyperparameters found by Bayesian optimization. Also, in order to analyze the general classification performance, we have chosen the macro-avg evaluation, which makes an averaging calculation by class

and measured using the test data and considering accuracy, precision, recall, and F1-score indices, computed from the confusion matrix [30].

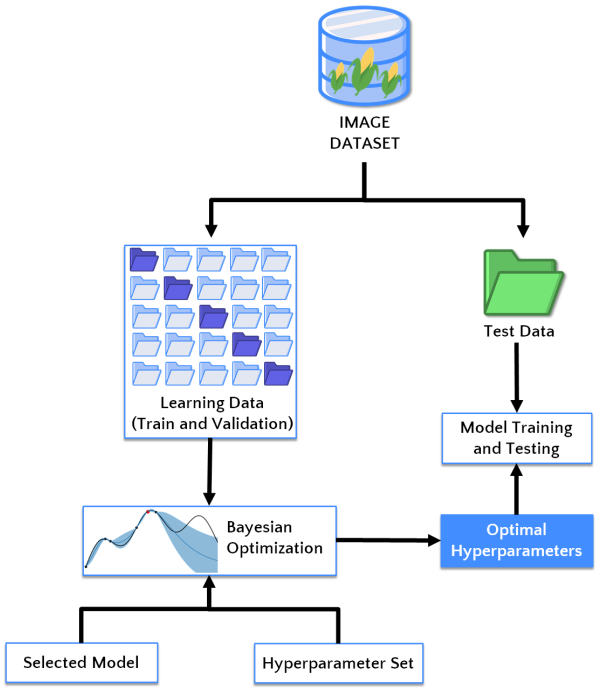


Fig. 3. Process of hyperparameter selection and model evaluation with the k-fold cross-validation.

For the hyperparameters of each CNN model, we consider: (i) batch size; (ii) learning rate; and (iii) momentum (as described in Section III-C). The values used to perform the Bayesian optimization for the hyperparameters are listed in Table I and all hyperparameters are searched considering a uniform distribution.

TABLE I. HYPERPARAMETER SEARCH SPACE USED FOR BAYESIAN OPTIMIZATION.

Hyperparameter	Value
Batch Size	[16, 32]
Learning Rate	[0.001, 0.01]
Momentum	[0, 1]

The best values found through the cross-validated Bayesian optimization process are depicted in Table II.

IV. RESULTS AND DISCUSSION

In this section, we compare the model prediction performance across each CNN evaluated.

A. Experimental Environment

All experiments were executed on a machine with an Intel i5 3.00 GHz processor, 16 GB RAM, a GPU NVIDIA GeForce GTX Titan Xp with 12 GB memory, and operating system Ubuntu 16.04.2 LTS. The models were programmed using Python (version 3.6) and PyTorch (version 1.4) deep learning framework [31] with CUDA version 8.0 and cuDNN 6.0. The hyperparameter optimization algorithm Bayesian was drawn from the *bayesian-optimization*[2] library, version 1.2.0.

According to the process shown in Fig. 3, the five-fold-cross-validation Bayesian optimization was performed and the optimal hyperparameters for each CNN are found through each model tuning. The training was performed using the optimal hyperparameters (see Table II) defined for each CNN and the learning epochs were set to 30. All CNNs evaluated were previously trained using ImageNet dataset [32], adjusting the parameters in the deepest layers. Also, we resized all images to 224 $\times$ 224 pixels to adapt for the input of each CNN architecture evaluated, and we applied data augmentation strategies [22] in the training images by random rotation (considering angles between 0° and 360°), vertical and horizontal flips.

B. Experimental Results

The experiment is performed to compare the performance of each CNN to classify the maize leaf disease dataset. Tables III, IV, and V presents the accuracy, precision, recall, and F1-score obtained when testing each CNN with optimal hyperparameters defined by Bayesian Optimization. Interestingly, the three CNNs achieved 97% of accuracy and this indicates that optimization allowed a better generalization in all models. Also, our best result of 97% accuracy was better or very close to the accuracy scores presented in the literature (92.85% in Sibiya & Sumbwanyambe [16], 97.60% in Hu et al. [17], and 97.89% in Priyadharshini et al. [15], 98% in Bhatt et al. [14]).

We observed that when comparing the overall performance of all four classes, the class gray leaf spot showed slightly lower performance indices. This is due to the imbalance between the classes: the gray leaf spot is the smallest class, about 43% of the size of the largest class (common rust). To overcome the imbalance, we applied data augmentation strategies, which not allowing a significant decreasing in the classification performance.

In addition to classification evaluation, we also analyzed the correlation between each of the four maize leaf classes registered from experimental results considering the testing set, as shown in Fig. 4. In this representation, we observed that there are a few correlations, which are categorized into weak (≤ 0.39) and moderate (≥ 0.40). We will focus our analysis on the moderate correlation, which is of most interest in our study. As can be seen in the Fig. 4, the correlations between the gray leaf spot, common rust, and healthy classes have a moderate intensity of the correlation for ResNet-50 and SqueezeNet.

The correlation between the common rust and gray leaf spot classes indicates that when a leaf is common rust, there is a possibility of classifying it as a gray leaf spot. And, the correlation between the gray leaf spot and healthy leaf indicates that there are situations where gray leaf spot is classified as healthy. This result suggests that there are some cases in which the patterns between the leaves are similar, although the leaves are of different classes.

Although the studies proposed by [14], [15], [16], and [17] use the same dataset considered in our study, they considered a hold-out validation technique, which generates biased sets

[2]https://pypi.org/project/bayesian-optimization/

TABLE II. HYPERPARAMETERS OPTIMIZED FOR EACH CNN.

Hyperparameter	AlexNet	ResNet-50	SqueezeNet
Batch size	32	32	18
Learning rate	0.0036931912206946992	0.0042498507802230421	0.0026763419023990384
Momentum	0.13865257470287484	0.4755170748794698	0.34556072704304774

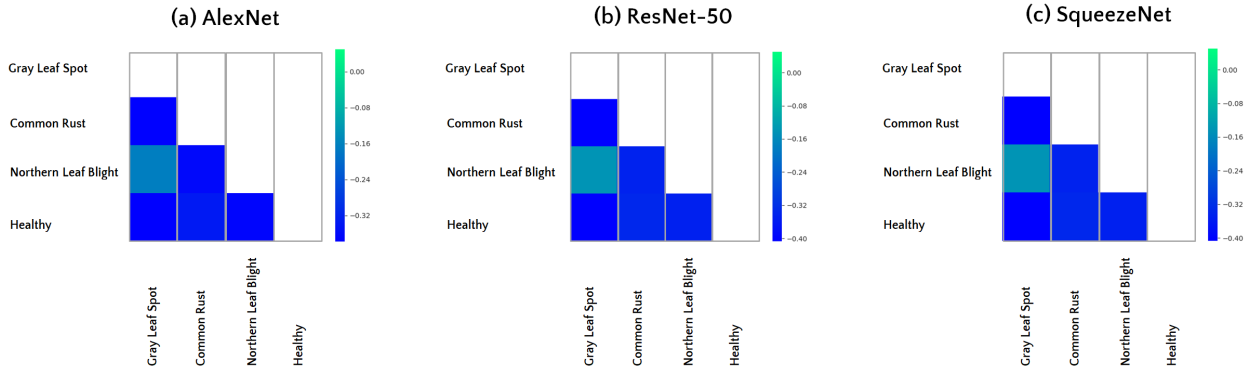


Fig. 4. The correlation between four maize leaf classes considering each CNN evaluated.

TABLE III. TESTING PERFORMANCE FOR ALEXNET ARCHITECTURE.

	Precision	Recall	F1-Score
Gray Leaf Spot	91%	85%	88%
Common Rust	100%	99%	100%
Northern Leaf Blight	92%	95%	93%
Healthy	99%	100%	100%
Average	96%	95%	95%
Accuracy		97%	

TABLE IV. TESTING PERFORMANCE FOR RESNET-50 ARCHITECTURE.

	Precision	Recall	F1-Score
Gray Leaf Spot	86%	93%	89%
Common Rust	100%	99%	100%
Northern leaf blight	96%	91%	93%
Healthy	99%	100%	100%
Average	95%	96%	96%
Accuracy		97%	

TABLE V. TESTING PERFORMANCE FOR SQUEEZENET ARCHITECTURE.

	Precision	Recall	F1-Score
Gray Leaf Spot	86%	93%	89%
Common Rust	100%	99%	100%
Northern leaf blight	96%	91%	93%
Healthy	99%	100%	100%
Average	95%	96%	96%
Accuracy		97%	

and unexpected values of accuracy. In contrast, we adopted k-fold cross-validation technique to better estimate the accuracy of the studied CNN architectures, which is more robust to outliers and eventual overfitting.

The general quality of our optimized models was estimated using the F1-Score. This metric is an excellent alternative to deal with the imbalance between the classes because it is the harmonic average between precision and recall calculations. Therefore, in terms of F1-Score, the best result was obtained by ResNet-50 and SqueezeNet models (96%).

V. CONCLUSION

The results presented in this study allow us to conclude that for maize leaf disease classification, the use of CNNs improved through Bayesian hyperparameter optimization is a promising alternative. Based on the comparative analysis, we could conclude that our best result of 97% was better or very close to the accuracy scores presented in the literature. Moreover, it is important to stress that our method is validated using a cross-validation strategy, which is more robust to over-fitting and generates results more reliable. Our results suggest that hyperparameter optimization improved the performance of all CNNs evaluated. The models generated have been able to extract important features about visual patterns of maize leaf. Although the main focus of this study is to classify maize leaf diseases, a classifier system that identifies with high performance a healthy leaf is attractive for farmers to manage the need resources on the crop.

We believe that our approach requires less time investment in a real-world context because the maize leaves can be acquired without the need to place them on a homogeneous background. Thus, this study is suitable for farmers looking for early detection or breeders evaluating the incubation period for a given disease.

As future work, we hope to apply our approach to classify more types of maize leaf and other types of leaf diseases, evaluate further optimization algorithms, and exploit more data augmentation strategies.

ACKNOWLEDGMENTS

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the TITAN Xp GPU used for this research. We would like to thanks CAPES and FAPEMIG for the financial support. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

REFERENCES

[1] Nikos Alexandratos and Jelle Bruinsma. World agriculture towards 2030/2050: the 2012 revision. Technical report, Jun 2012. FAO Document Repository: http://www.fao.org/3/a-ap106e.pdf.

[2] World Health Organization et al. *The state of food security and nutrition in the world 2018: building climate resilience for food security and nutrition*. Food & Agriculture Org., 2018.

[3] Bekele Shiferaw, Boddupalli M. Prasanna, Jonathan Hellin, and Marianne Bänziger. Crops that feed the world 6. past successes and future challenges to the role played by maize in global food security. *Food Security*, 3(3):307, Aug 2011.

[4] Jayme Garcia Arnal Barbedo. A review on the main challenges in automatic plant disease identification based on visible range images. *Biosystems Engineering*, 144:52 – 60, 2016.

[5] Sharada P. Mohanty, David P. Hughes, and Marcel Salathé. Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, 7(September):1–10, 2016.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[7] M. A. Ponti, L. S. F. Ribeiro, T. S. Nazare, T. Bui, and J. Collomosse. Everything you wanted to know about deep learning for computer vision but were afraid to ask. In *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, pages 17–41, Oct 2017.

[8] Srdjan Sladojevic, Marko Arsenovic, Andras Anderla, Dubravko Culibrk, and Darko Stefanovic. Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification. *Computational Intelligence and Neuroscience*, 2016, 2016.

[9] Edna Chebet Too, Li Yujian, Sam Njuki, and Liu Yingchun. A comparative study of fine-tuning deep learning models for plant disease identification. *Computers and Electronics in Agriculture*, 161:272 – 279, 2019. BigData and DSS in Agriculture.

[10] Chad DeChant, Tyr Wiesner-Hanks, Siyuan Chen, Ethan L. Stewart, Jason Yosinski, Michael A. Gore, Rebecca J. Nelson, and Hod Lipson. Automated identification of northern leaf blight-infected maize plants from field imagery using deep learning. *Phytopathology®*, 107(11):1426–1432, 2017. PMID: 28653579.

[11] X. Zhang, Y. Qiao, F. Meng, C. Fan, and M. Zhang. Identification of maize leaf diseases using improved deep convolutional neural networks. *IEEE Access*, 6:30370–30377, 2018.

[12] Zhongqi Lin, Shaomin Mu, Aiju Shi, Chao Pang, Xiaoxiao Sun, et al. A novel method of maize leaf disease image identification based on a multichannel convolutional neural network. *Transactions of the ASABE*, 61(5):1461–1474, 2018.

[13] Enquhone Alehegn. Ethiopian maize diseases recognition and classification using support vector machine. *International Journal of Computational Vision and Robotics*, 9(1):90–109, 2019.

[14] Prakruti Bhatt, Sanat Sarangi, Anshul Shivhare, Dineshkumar Singh, and Srinivasu Pappula. Identification of diseases in corn leaves using convolutional neural networks and boosting. In *Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods - Volume 1: ICPRAM,*, pages 894–899. INSTICC, SciTePress, 2019.

[15] Ramar Ahila Priyadharshini, Selvaraj Arivazhagan, Madakannu Arun, and Annamalai Mirnalini. Maize leaf disease classification using deep convolutional neural networks. *Neural Computing and Applications*, 31(12):8887–8895, Dec 2019.

[16] Malusi Sibiya and Mbuyu Sumbwanyambe. A computational procedure for the recognition and classification of maize leaf diseases out of healthy leaves using convolutional neural networks. *AgriEngineering*, 1(1):119–131, 2019.

[17] Rongjie Hu, Shan Zhang, Peng Wang, Guoming Xu, Daoyong Wang, and Yuqi Qian. The identification of corn leaf diseases based on transfer learning and data augmentation. In *Proceedings of the 2020 3rd International Conference on Computer Science and Software Engineering*, CSSE 2020, page 58–65, New York, NY, USA, 2020. Association for Computing Machinery.

[18] Abdul Waheed, Muskan Goyal, Deepak Gupta, Ashish Khanna, Aboul Ella Hassanien, and Hari Mohan Pandey. An optimized dense convolutional neural network model for disease recognition and classification in corn leaf. *Computers and Electronics in Agriculture*, 175:105456, 2020.

[19] Ian Dewancker, Michael McCourt, Scott Clark, Patrick Hayes, Alexandra Johnson, and George Ke. A stratified analysis of bayesian optimization methods, 2016.

[20] James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2546–2554. Curran Associates, Inc., 2011.

[21] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2951–2959. Curran Associates, Inc., 2012.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[24] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size, 2016.

[25] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade: Second Edition*, pages 437–478, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[26] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *CoRR*, abs/1404.5997, 2014.

[27] Martin Pelikan, David E. Goldberg, and Erick Cantú-Paz. Boa: The bayesian optimization algorithm. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation - Volume 1*, GECCO'99, page 525–532, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[28] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[29] Pierre A. Devijver and Josef Kittler. *Pattern Recognition: A Statistical Approach*. Prentice-Hall, 1982.

[30] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2Nd Edition)*. Wiley-Interscience, New York, NY, USA, 2000.

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc., 2019.

[32] J. Deng, W. Dong, R. Socher, L. J. Li, Kai Li, and Li Fei-Fei. Imagenet:

A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.

MS-DIAL: Multi-Source Domain Alignment Layers for Unsupervised Domain Adaptation

Lucas Fernando Alvarenga e Silva

Instituto de Ciência e Tecnologia
Universidade Federal de São Paulo – UNIFESP
12247-014, São José dos Campos, SP – Brazil
Email: e.lucas@unifesp.br

Jurandy Almeida

Instituto de Ciência e Tecnologia
Universidade Federal de São Paulo – UNIFESP
12247-014, São José dos Campos, SP – Brazil
Email: jurandy.almeida@unifesp.br

***Abstract*—In general, deep neural networks trained on a given labeled dataset are expected to produce equivalent results when tested on a new unlabeled dataset. However, data are generally collected by different devices or under varying conditions and thus they often are not part of a same domain, yielding poor results. This is due to the domain shift between data distributions and has been the goal of a research area known as unsupervised domain adaptation. Many prior works have been designed to transfer knowledge between two domains: one source to one target. Since data may be taken from different sources and with different distributions, multi-source domain adaptation has received increasing attention. This paper presents the Multi-Source DomaIn Alignment Layers (MS-DIAL), which reduce the domain shift between multiple sources and a given target by embedding domain alignment layers in any given network. Except for the embedded layers, all the other network parameters are shared among all domains, saving processing time and memory usage. Experiments were performed on digit and object recognition tasks with five public datasets widely used to evaluate domain adaptation methods. Results show that the proposed method is promising and outperforms state-of-the-art approaches.**

I. Introdução

Nos últimos anos, o campo da visão computacional tem alcançado resultados surpreendentes em uma variedade de problemas desafiadores, em especial, na classificação de imagens em grandes bases de dados propostas para tarefas amplamente consideradas como difíceis, como a ImageNet [1]. Esses resultados têm sido obtidos a partir da utilização de métodos de aprendizado de maquina, em especial, graças aos avanços significativos introduzidos pela aprendizagem profunda (do inglês, *deep learning*) com as redes neurais convolucionais (do inglês, *convolutional neural networks* – CNNs).

Em geral, é esperado que modelos treinados em bases de dados anotadas (*i.e.*, conjunto de treinamento) produzam resultados equivalentes quando aplicados à novos dados não-anotados (*i.e.*, conjunto de teste). Tal premissa parte do pressuposto de que os dados anotados usados no treinamento e os dados não-anotados usados no teste pertencem ao mesmo domínio, isto é, apresentem uma mesma distribuição de probabilidade. Porém, na prática, os dados são geralmente coletados por dispositivos diferentes ou sob condições variadas e, portanto, não necessariamente fazem parte de um mesmo domínio, o que pode produzir resultados insatisfatórios. Isso acontece

Figura 1. Imagens do conjunto de dados Office-Home, em que as linhas representam, respectivamente, os domínios: arte, clipart, produto e mundo real; e nas colunas estão representadas algumas categorias do conjunto de dados, como: colher, pia, xícara, caneta e faca. Adaptado de Venkateswara *et al.* [2].

devido a mudança de domínio (do inglês, *domain shift*) que há entre as distribuições de dados e é objeto de pesquisa do campo denominado adaptação de domínio não-supervisionada (do inglês, *unsupervised domain adaptation* – UDA).

A grosso modo, as soluções existentes em UDA se enquadram em duas vertentes bem definidas: (*i*) as que exploram características invariantes entre domínios (do inglês, *domain invariant features*) ou (*ii*) as que reduzem a discrepância entre as distribuições de dados [3]. A maioria dos trabalhos anteriores considera a transferência de conhecimento entre dois domínios: um fonte (do inglês, *source*) e um alvo (do inglês, *target*). Todavia, na prática, os dados são normalmente provenientes de várias fontes e com distribuições distintas, como ilustrado na Figura 1, na qual imagens de uma mesma classe são coletadas de diversos sites da Internet e adquiridas sob condições distintas [4]. Nesse cenário, é comum agrupar os dados de vários domínios-fonte em um único conjunto e, em seguida, aplicar métodos projetados para lidar com a adaptação de um único domínio-fonte para um único domínio-alvo [5]. Entretanto, essa abordagem geralmente não produz resultados satisfatórios, uma vez que os domínios-fonte não necessariamente contribuem da mesma maneira para o processo de transferência de conhecimento para o domínio-alvo [6].

O problema de adaptação de domínio de várias fontes (do inglês, *multi-source domain adaptation* – MSDA) é mais

complexo e desafiador, já que pode haver um deslocamento entre as distribuições dos domínios-fonte, bem como eles podem fornecer informações complementares para o processo de transferência de conhecimento para o domínio-alvo. Além desses fatores, também podem ser encontradas classes diferentes entre os domínios-fonte (do inglês, *category shift*) [4].

Este trabalho contribui com uma nova proposta para o problema de MSDA, denominada MS-DIAL (do inglês, *multi-source domain alignment layers*), a qual reduz a discrepância entre as distribuições dos domínios-fonte e do domínio-alvo a partir da inserção de camadas de alinhamento de domínio em diversos níveis da rede. Nessa abordagem, o nível de alinhamento das distribuições é ajustado de forma automática pela rede por meio do uso de parâmetros aprendíveis em tempo de treinamento. Para uma melhor separação entre categorias do domínio-alvo, a entropia das predições obtidas para amostras do lote do domínio-alvo é usada como medida de erro para o otimizador, buscando assim ajustar os parâmetros da rede às características extraídas dos dados do domínio-alvo.

Experimentos foram realizados em cinco conjuntos de dados públicos usados para avaliar métodos de UDA: MNIST [7], MNIST-M [8], SVHN [9] e *Synthetic Digits* [10], que foram propostos para tarefas de reconhecimento de dígitos; e Office-Home [2], que aborda a tarefa de reconhecimento de objetos. Os resultados obtidos demonstram que o método proposto é eficaz, superando abordagens do estado da arte.

O restante deste trabalho está organizado da seguinte maneira. A Seção II discute trabalhos relacionados. A Seção III introduz o MS-DIAL e mostra como ele pode ser usado para lidar com tarefas de MSDA. A Seção IV apresenta o protocolo experimental e a comparação dos resultados do MS-DIAL com outros métodos. Por fim, conclusões e direções para trabalhos futuros são oferecidos na Seção V.

II. Trabalhos Relacionados

A adaptação de domínio não-supervisionada de única fonte para único alvo (do inglês, *single-source to single-target domain adaptation*) conta com um domínio-fonte anotado e um domínio-alvo não-anotado, e tem por objetivo adaptar um modelo treinado em dados anotados do domínio-fonte para reconhecer instâncias provenientes de dados não-anotados do domínio-alvo. É um problema desafiador e com diversas propostas de solução, algumas com modelos rasos e atualmente tem se voltando ao uso de redes neurais profundas. Os métodos rasos se baseiam na redução da discrepância entre domínios e buscam obter características invariantes, como a análise de componentes de transferência (do inglês, *transfer component analysis* – TCA) [11] e a incorporação de correspondência de distribuição (do inglês, *distribution-matching embedding* – DME) [12]. Em trabalhos recentes, redes neurais profundas, normalmente submetidas a um treinamento adversário, têm sido usadas em duas vertentes: (*i*) para mapear dados de ambos os domínios em uma distribuição comum ou (*ii*) para distinguir amostras provenientes de domínios fonte e alvo. Alguns exemplos são as redes neurais de domínio adversário (do inglês, *domain-adversarial neural networks* – DANN) [10] e a discrepância média máxima ponderada (do inglês, *weighted maximum mean discrepancy* – WMMD) [13]. Outras vertentes, como a inserção de camadas de alinhamento de domínio [3], [14], estão intimamente relacionadas a este trabalho.

Já a adaptação de domínio não-supervisionada de várias fontes para único alvo (do inglês, *multi-source to single-target domain adaptation*) é ainda mais desafiadora. É um problema emergente nos últimos anos e atualmente existem algumas propostas de solução, como a correspondência de momento para adaptação de domínio de várias fontes (do inglês, *moment matching for multi-source domain adaptation* – M3SDA) [15], a rede de cauda profunda (do inglês, *deep cocktail network* – DCTN) [4], a rede de agregação de domínio (do inglês, *domain aggregation network* – DARN) [6], a rede de correspondência de vários domínios (do inglês, *multiple domain matching network* – MDMN) [16], as redes adversárias de domínio de várias fontes (do inglês, *multisource domain adversarial networks* – MDAN) [17] e a adaptação de domínio de destilação de múltiplas fontes (do inglês, *multi-source distilling domain adaptation* – MDDA) [18]. Em geral, as abordagens existentes baseiam-se em redes neurais de múltiplos fluxos na qual a quantidade de classificadores e/ou extratores de características é ajustada proporcionalmente à quantidade de domínios. M3SDA [15] é uma rede que contém um único extrator de características comum a todos os domínios, porém, com um classificador para cada domínio-fonte, cujas saídas são agrupadas por média ponderada. De maneira similar, DCTN [4] e DARN [6] usam um único conjunto de pesos que é compartilhado pelos extratores de características de todos os domínios. Contudo, para realizar a adaptação de domínios, DCTN adota medidas de perplexidade e discriminadores de domínio, enquanto DARN emprega módulos de discrepância. MDDA [18] é uma rede adversária composta por um extrator de características e um classificador para cada domínio-fonte, cuja predição final é dada pela média ponderada das predições de todos os domínios, na qual os pesos são obtidos a partir de métricas de discriminação de domínio.

III. MS-DIAL: Multi-Source DomaIn Alignment Layers

Sejam os conjuntos de dados anotados $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M$ referentes a M domínios-fonte que compartilham o mesmo conjunto de rótulos $\mathcal{Y}$ com um conjunto de dados não-anotados $\mathcal{T}$ referente ao único domínio-alvo. Suponha que cada domínio-fonte $\mathcal{S}_i = \{(\mathbf{x}_i^j, \mathbf{y}_i^j)\}_{j=1}^{N_i}$ corresponde a um conjunto de tuplas que associa dados observados $X_i = \{\mathbf{x}_i^j\}_{j=1}^{N_i}$ a seus respectivos rótulos $Y_i = \{\mathbf{y}_i^j\}_{j=1}^{N_i}$, os quais foram extraídos da distribuição-fonte $p_i(\mathbf{x}, \mathbf{y})$, em que N_i é o número de amostras em $\mathcal{S}_i$. Como os rótulos do domínio-alvo $\mathcal{T}$ não são conhecidos, assuma que ele seja formado por dados $X_T = \{\mathbf{x}_T^j\}_{j=1}^{N_T}$ extraídos da distribuição-alvo $p_T(\mathbf{x}, \mathbf{y}), \mathbf{y} \in \mathcal{Y}$, em que N_T é o número de amostras em $\mathcal{T}$. Assim, o problema de MSDA consiste em encontrar um conjunto de parâmetros $\theta \in \Theta$ para uma rede neural de forma que suas predições para o conjunto de rótulos $Y_T = \{\mathbf{y}_T^j\}_{j=1}^{N_T}$ ainda não conhecidos do domínio-alvo $\mathcal{T}$ sejam as melhores possíveis.

Em geral, trabalhos anteriores de MSDA usam topologias de rede com múltiplos fluxos, normalmente com um fluxo independente para cada domínio, algumas com um conjunto de parâmetros θ diferente para cada fluxo e, assim, cada domínio tem o seu próprio extrator de características e classificador; e outras com um conjunto de parâmetros θ compartilhado entre os fluxos, geralmente, pelos extratores de características de todos os domínios, mas cada um tendo o seu próprio classificador. Diferente dessas abordagens, na topologia de rede desenvolvida neste trabalho, o conjunto de parâmetros θ é compartilhado entre os fluxos de todos os domínios, tanto pelos extratores de características quanto pelos classificadores, exceto nas camadas de alinhamento de domínio com várias fontes (do inglês, *multi-source domain alignment layers* – MS-DIAL), como ilustrado na Figura 2. Durante a fase de treinamento, as amostras contidas nos mini-lotes são agrupadas de acordo com o domínio a qual pertencem e cada grupo de amostras segue um caminho diferente, sendo encaminhado a uma camada de normalização de lote associada ao seu respectivo domínio. Dessa forma, é possível utilizar uma única instância da topologia de rede e, portanto, um mesmo conjunto de parâmetros θ para todos os domínios, reduzindo assim o custo computacional e a utilização de memória.

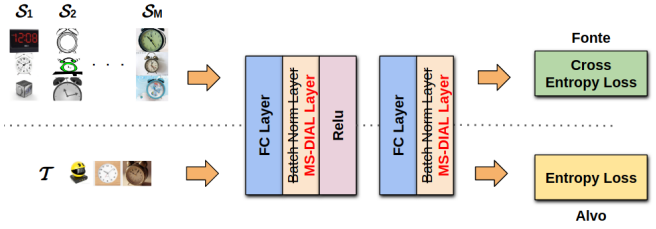


Figura 2. Exemplo de uma topologia de rede adaptada com camadas MS-DIAL para realizar a MSDA dos domínios-fonte $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M$ para o domínio-alvo $\mathcal{T}$. Para isso, as camadas de normalização de lote foram substituídas por camadas MS-DIAL, mantendo, assim, todas as demais camadas compartilhadas entre todos os domínios $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M, \mathcal{T}$.

A. *Preditores de Fonte e Alvo*

O ponto de partida para a abordagem proposta neste trabalho são as camadas de alinhamento de domínio (do inglês, *domain alignment layers* – DIAL) [3], que reduzem a discrepância entre as distribuições dos domínios fonte e alvo ao longo do fluxo de dados na topologia de rede, levando as diferentes distribuições a uma mesma distribuição de referência. Tais camadas foram projetadas para adaptação de domínio de única fonte para único alvo. Inicialmente, as amostras $x \subseteq \{X_S \cup X_T\}$ dos mini-lotes de entrada são divididas em dois grupos: (*i*) amostras do domínio-fonte $x_S \subseteq X_S$ e (*ii*) amostras do domínio-alvo $x_T \subseteq X_T$. Em seguida, cada grupo de amostras é encaminhado para uma camada de normalização de lote [19] associada ao seu respectivo domínio, as quais ajustam cada uma das distribuições dos domínios a uma distribuição de referência, porém, sem realizar transformações afins, como mostrado na Equação 1. Por fim, para controlar a sobreposição de todos os domínios, foram inseridos dois parâmetros aprendíveis pela rede em tempo de treinamento, denominados α e β, os quais tem por objetivo transformar linearmente as distribuições sobrepostas na distribuição referência, ou seja, deslocar e/ou escalar as distribuições de modo que maximize o acerto nas predições das amostras do domínio-alvo, como apresentado na Equação 2, em que $\oplus$ denota a operação de concatenação das saídas de camadas distintas.

$$BN(x) = \frac{x - E[x]}{\sqrt{Var[x] + \epsilon}} \tag{1}$$

$$DIAL(x) = \{BN_S(x_S) \oplus BN_T(x_T)\} \cdot \alpha + \beta \tag{2}$$

A grosso modo, este trabalho estende as camadas DIAL para realizar a adaptação de domínio de várias fontes para único alvo. Formalmente, as camadas MS-DIAL generalizam as transformações realizadas na Equação 2, aplicando-as a todos os domínios-fonte, como mostrado na Equação 3. Similar do DIAL, as amostras $x \subseteq \{X_1 \cup X_2 \cup \cdots \cup X_M \cup X_T\}$ dos mini-lotes de entrada são inicialmente agrupadas em $M + 1$ sub-lotes $x_1 \subseteq X_1, x_2 \subseteq X_2, \ldots, x_M \subseteq X_M, x_T \subseteq X_T$ de acordo com o domínio a qual pertencem e, em seguida, encaminhadas às camadas de normalização de lote $BN_1, BN_2, \ldots, BN_M, BN_T$, respectivamente, ajustando a distribuição de todos os domínios para uma mesma distribuição de referência e, por fim, a sobreposição das distribuições fonte e alvo é controlada por parâmetros α e β aprendíveis em tempo de treinamento.

A Figura 3 ilustra o fluxo de dados inerente às camadas MS-DIAL, desde o recebimento do mini-lote, a sua separação em vários domínios, a transformação para a distribuição referência e, por fim, a transformação afim para controle do alinhamento dos domínios. O caminho indicado em vermelho representa o fluxo de dados durante a fase de teste.

$$MS\text{-}DIAL(x) = \left\{ \left\{ \bigoplus_{i=1}^{M} BN_i(x_i) \right\} \oplus BN_T(x_T) \right\} \cdot \alpha + \beta \tag{3}$$

B. *Treinamento e Inferência*

Durante a fase de treinamento, os mini-lotes devem conter amostras $x = x_1 \oplus x_2 \oplus \cdots \oplus x_M \oplus x_T$ provenientes de todos os domínios-fonte, ou seja, $x_1 \subseteq \mathcal{S}_1, x_2 \subseteq \mathcal{S}_2, \ldots, x_M \subseteq \mathcal{S}_M$, as quais são acompanhadas de seus respectivos rótulos; e também do domínio-alvo, isto é, $x_T \subseteq \mathcal{T}$, para as quais os rótulos não são conhecidos. Ao final da passagem de um mini-lote pela rede, são obtidas predições $\{f_i^\theta(\mathbf{y}_i^k; \mathbf{x}_i^k)\}_{k=1}^{|x_i|}$ para amostras $x_1, x_2, \ldots, x_M$ dos domínios-fonte e também predições $\{f_T^\theta(\mathbf{y}; \mathbf{x}_T^k)\}_{k=1}^{|x_T|}$ para amostras x_T do domínio-alvo.

Similar ao DIAL, o valor do erro entregue ao otimizador é obtido a partir de uma função de perda $\mathcal{L}(\theta)$ composta por duas componentes, uma supervisionada $\mathcal{L}_\mathcal{S}(\theta)$ calculada a partir das predições obtidas para amostras dos domínios-fonte $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M$ e outra não-supervisionada $\mathcal{L}_\mathcal{T}(\theta)$ calculada a partir das predições das amostras do domínio-alvo $\mathcal{T}$.

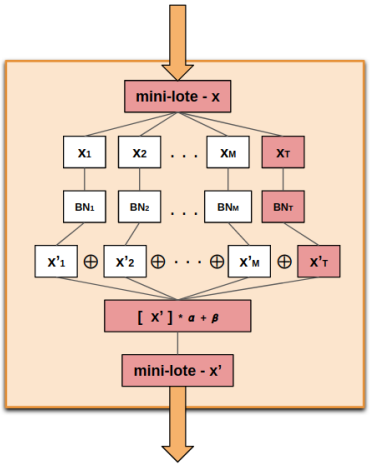


Figura 3. Fluxo de dados dos mini-lotes x nas camadas MS-DIAL durante a fase de treinamento, em que $BN_1, BN_2, \ldots, BN_M$ são as camadas de normalização de lote aplicadas às amostras $x_1, x_2, \ldots, x_M$ dos domínios-fonte $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M$, respectivamente; BN_T é a camada de normalização de lote aplicada às amostras x_T do domínio-alvo $\mathcal{T}$; e α e β são parâmetros aprendíveis pela rede. O caminho destacado em vermelho refere-se ao fluxo de dados durante a fase de inferência.

A componente supervisionada $\mathcal{L}_\mathcal{S}$ é a entropia cruzada das amostras dos domínios-fonte, que é calculada pela Equação 4.

$$\mathcal{L}_\mathcal{S}(\theta) = -\sum_{i=1}^{M} \frac{1}{|x_i|} \sum_{k=1}^{|x_i|} \log f_i^\theta(\mathbf{y}_i^k; \mathbf{x}_i^k) \tag{4}$$

Já a componente não-supervisionada $\mathcal{L}_\mathcal{T}$ refere-se a entropia das amostras do domínio-alvo e é usada para forçar o modelo a decidir com mais confiança, sendo dada pela Equação 5.

$$\mathcal{L}_\mathcal{T}(\theta) = -\frac{1}{|x_T|} \sum_{k=1}^{|x_T|} \sum_{\mathbf{y}\in\mathcal{Y}} f_T^\theta(\mathbf{y}; \mathbf{x}_T^k) \log f_T^\theta(\mathbf{y}; \mathbf{x}_T^k) \tag{5}$$

A função de perda $\mathcal{L}(\theta)$ é a soma ponderada de $\mathcal{L}_\mathcal{S}(\theta)$ e $\mathcal{L}_\mathcal{T}(\theta)$, ou seja, $\mathcal{L}(\theta) = \mathcal{L}_\mathcal{S}(\theta) + \lambda\mathcal{L}_\mathcal{T}(\theta)$, em que λ é um hiperparâmetro associado ao peso da contribuição de $\mathcal{L}_\mathcal{T}(\theta)$. Nos experimentos, o hiperparâmetro λ foi fixado em 0,1.

Uma vez ajustados os parâmetros θ, os fluxos de dados associados aos domínios-fonte $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M$ não são mais necessários. Dessa forma, as camadas MS-DIAL passam a operar como camadas padrões de normalização de lote quando usadas para realizar inferências, encaminhando os mini-lotes através dos caminhos associados somente ao domínio-alvo $\mathcal{T}$.

IV. Experimentos

Esta seção apresenta detalhes sobre o protocolo experimental adotado para avaliar o método proposto e também relata os resultados obtidos. A avaliação experimental foi conduzida em conjuntos de dados de pequeno e grande porte e o método proposto foi comparado com abordagens do estado da arte.

A. Conjuntos de Dados

O método proposto foi avaliado em duas tarefas distintas que envolvem cinco conjuntos de dados públicos amplamente usados para avaliar métodos de UDA: (*i*) no reconhecimento de dígitos dos conjuntos de dados MNIST [7], MNIST-M [8], SVHN [9] e *Synthetic Digits* [10]; e (*ii*) no reconhecimento de objetos usando o conjunto de dados Office-Home [2]. A seguir, são fornecidos detalhes de cada um desses conjuntos.

O conjunto de dados MNIST [7] é composto por imagens monocromáticas com resolução de 28x28 *pixels*, sendo 60000 imagens para treinamento e 10000 para teste. Essas imagens referem-se a dígitos manuscritos dos algarismos de 0 a 9, cada um correspondente a uma classe distinta.

O conjunto de dados MNIST-M [8] é composto por imagens coloridas de tamanho 32x32 *pixels*, sendo 59001 imagens para treinamento e 9001 para teste. Elas resultam da combinação das imagens do MNIST com padrões aleatórios extraídos de fotos coloridas do conjunto de dados BSDS500 [20], na qual os *pixels* que compõem os dígitos tem suas cores invertidas. MNIST-M, assim como MNIST, possui 10 classes que correspondem aos algarismos de 0 a 9. Embora para humanos a tarefa se torne um pouco mais difícil, a inserção de padrões aleatórios ao fundo e a cor não uniforme dos dígitos categorizam uma grande mudança de domínio.

O conjunto de dados SVHN (do inglês, *Street View House Number*) [9] é composto por imagens coloridas com resolução de 32x32 *pixels*, sendo 73257 imagens para treinamento e 26032 para teste. Tais imagens contém fotos de dígitos tiradas da numeração de casas e foram agrupadas em 10 classes correspondentes a dígitos no intervalo de 0 a 9. Apesar das semelhanças com MNIST e MNIST-M, o desbalanceamento no número de imagens por classe, as alterações severas de iluminação e a descentralização dos dígitos nas imagens representam mudanças significativas de domínio.

O conjunto de dados *Synthetic Digits* (Synth) [10] é composto por imagens coloridas com resolução de 32x32 *pixels*, sendo 479400 imagens para treinamento e 9553 para teste. Ele é composto por imagens sintéticas obtidas a partir de transformações de posição, orientação, borramento e coloração de dígitos de fontes do Windows™, cujos parâmetros foram manualmente ajustados para mimetizar amostras do conjunto SVHN. Tais imagens, assim como SVHN, possuem 10 classes que correspondem a dígitos no intervalo de 0 a 9 e, apesar de imitar o SVHN, seu desbalanceamento é muito maior, o que dificulta a transferência de conhecimento.

Office-Home [2] é um conjunto de dados de grande porte usado como referência para avaliar métodos de UDA. Ele é composto por 15500 imagens coletadas de vários sites e repositórios de imagens da Internet, apresentando resoluções que variam de 18x18 até 6500x4900 *pixels*. Essas imagens estão distribuídas em 65 classes de objetos e divididas em 4 domínios distintos: arte (2427), clipart (4365), produto (4439) e mundo real (4357). Os números entre parênteses indicam a quantidade de imagens em cada domínio.

As tarefas de reconhecimento de dígitos e de objetos nesses conjuntos de dados são bastante desafiadoras devido à grande

Tabela I
ACURÁCIA DE CLASSIFICAÇÃO (%) NOS CONJUNTOS DE DADOS DE DÍGITOS.

Métodos	Domínios				Média
	MNIST	MNIST-M	SVHN	Synth	
SRC	96,78 ± 0,08	60,80 ± 0,21	68,99 ± 0,69	84,09 ± 0,27	77,66 ± 0,14
DANN	96,41 ± 0,13	60,10 ± 0,27	70,19 ± 1,30	83,83 ± 0,25	77,63 ± 0,35
M3SDA	96,95 ± 0,06	65,03 ± 0,80	71,66 ± 1,16	80,12 ± 0,56	78,44 ± 0,36
MDAN	97,10 ± 0,10	64,09 ± 0,31	77,72 ± 0,60	85,52 ± 0,19	81,11 ± 0,21
MDMN	97,15 ± 0,09	64,34 ± 0,27	76,43 ± 0,48	85,80 ± 0,21	80,93 ± 0,16
DARN	**98,09 ± 0,03**	**67,06 ± 0,14**	81,58 ± 0,14	86,79 ± 0,09	83,38 ± 0,06
MS-DIAL	94.33 ± 0,06	61.24 ± 1,27	**85.61 ± 0,47**	**92.86 ± 0,20**	**83.51 ± 1,53**
TAR	99,02 ± 0,02	94,66 ± 0,10	87,40 ± 0,17	96,90 ± 0,09	94,49 ± 0,07

diferença entre os domínios, como ilustrado na Figura 4.

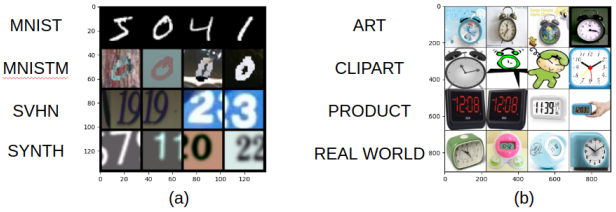


Figura 4. Exemplos de imagens do MNIST, MNIST-M, SVHN e Synth em (a) e dos domínios arte, clipart, produto e mundo real do Office-Home (b).

B. Protocolo Experimental

Os resultados do método proposto foram comparados com os relatados recentemente por Wen *et al.* [6] para cinco abordagens de referência: **DANN** [10], **M3SDA** [15], **MDAN** [17], **MDMN** [16], **DARN** [6]. Além disso, foram também considerados os resultados relatados por Wen *et al.* [6] para duas linhas de base: (*i*) **SRC**, que refere-se ao treinamento do modelo em um único conjunto composto por dados rotulados de todos domínios-fonte e, portanto, sem adaptação de domínio; e (*ii*) **TAR**, que refere-se ao treinamento do modelo em dados do domínio-alvo, porém, valendo-se do conhecimento prévio de seus rótulos verdadeiros, constituindo assim um limite superior para o desempenho de qualquer método de UDA.

Para se ter uma comparação justa, foi adotado o mesmo protocolo experimental usado por Wen *et al.* [6]. Em cada experimento, um domínio foi tomado como alvo e os demais foram usados como fonte. Esse processo foi repetido várias vezes, cada vez tomando um domínio diferente como alvo. Foi adotado a acurácia como métrica de classificação, descrita na Equação 6, que é calculada através da razão da quantidade de predições corretas, $VP+VN$, sendo VP e VN referentes a verdadeiros positivos e verdadeiros negativos, pela quantidade total de predições, $VP+VN+FP+FN$, onde FP e FN referem-se a falso positivo e falso negativo.

$$Acc = \frac{VP+VN}{VP+VN+FP+FN} \tag{6}$$

Para o reconhecimento de dígitos, os conjuntos de dados MNIST, MNIST-M, SVHN e Synth foram tratados como 4 domínios distintos. Para cada domínio, foram sub-amostrados aleatoriamente 20000 imagens para treinamento e 9000 para teste. Foram realizadas 20 repetições de cada experimento, sendo reportados a média e o erro padrão da acurácia do melhor modelo obtido em cada rodada. A topologia de rede adotada nesses experimentos foi a mesma usada por Peng *et al.* [15], substituindo-se camadas de normalização de lote padrão por camadas MS-DIAL. A rede foi treinada do zero por 120 épocas usando o algoritmo de otimização Adam [21] com tamanho de mini-lote de 64 (*i.e.*, 16 por domínio), decaimento de peso de 0,0005 e taxa de aprendizado inicial de 0,001 com decaimento programado por um fator de 10 nas épocas 50 e 90. Esse mesmo conjunto de parâmetros foi usado no trabalho de Roy *et al.* [22].

Para o reconhecimento de objetos, foram sub-amostrados aleatoriamente 2000 imagens de cada domínio para treinamento e as imagens restantes foram usadas para teste. Os resultados referem-se a média e o erro padrão das acurácias obtidas ao final de cada uma das 20 repetições que foram realizadas de cada experimento. Seguindo o trabalho de Roy *et al.* [22], foi adotada a rede ResNet-50 [23], substituindo-se camadas de normalização de lote padrão por camadas MS-DIAL. Primeiro, a rede foi inicializada com pesos pré-treinados na ImageNet [1] e a camada de saída foi substituída por uma camada totalmente conectada com 65 neurônios de saída e pesos inicializados aleatoriamente. Em seguida, a rede foi treinada por 60 épocas usando o algoritmo de otimização SGD (do inglês, *Stochastic Gradient Descent*) com tamanho de mini-lote de 80 (*i.e.*, 20 por domínio), fator de momentum de 0,9, decaimento de peso de 0,0005, taxa de aprendizado inicial de 0,01 para os parâmetros da camada de saída e de 0,001 para os demais parâmetros da rede. Decaimento programado foi usado para reduzir as taxas de aprendizado iniciais por um fator de 10 na época 54.

Os experimentos foram realizados em um servidor equipado com dois processadores Intel Xeon E5-2683v4 (16 núcleos de 2,1 GHz), 128 GBytes de memória DDR4 e 2 GPUs NVIDIA Tesla K80. O servidor executa o sistema operacional Linux CentOS 7.4 (*kernel* 3.10.0) e o sistema de arquivos ext4. Todos os códigos-fonte foram implementados em Python (versão 3.6.7) usando a biblioteca PyTorch (versão 1.2.0).

C. Resultados

Nas Tabelas I e II, são comparados os resultados obtidos pelo MS-DIAL e os relatados por Wen *et al.* [6] para tarefas de

Tabela II
ACURÁCIA DE CLASSIFICAÇÃO (%) NO CONJUNTO DE DADOS OFFICE-HOME.

Métodos	Domínios				Média
	Arte	Clipart	Produto	Mundo Real	
SRC	58,02 ± 0,47	57,29 ± 0,30	74,26 ± 0,22	77,98 ± 0,25	66,89 ± 0,16
DANN	57,39 ± 0,69	57,35 ± 0,35	73,78 ± 0,27	78,12 ± 0,21	66,66 ± 0,19
M3SDA	64,05 ± 0,61	62,79 ± 0,37	76,21 ± 0,30	78,63 ± 0,22	70,42 ± 0,18
MDAN	68,14 ± 0,58	67,04 ± 0,21	81,03 ± 0,22	82,79 ± 0,15	74,75 ± 0,18
MDMN	68,67 ± 0,55	67,75 ± 0,20	81,37 ± 0,18	83,32 ± 0,14	75,28 ± 0,15
DARN	70,00 ± 0,38	68,42 ± 0,14	**82,75 ± 0,21**	**83,88 ± 0,16**	76,26 ± 0,13
MS-DIAL	**82.85 ± 0,10**	**76.71 ± 0,10**	80.74 ± 0,09	82.70 ± 0,09	**80.75 ± 0,28**
TAR	71,19 ± 0,38	79,16 ± 0,16	90,66 ± 0,15	85,60 ± 0,14	81,65 ± 0,12

MSDA com os conjuntos de dados de dígitos e Office-Home, respectivamente. O melhor resultado obtido para as amostras de teste de cada domínio-alvo está destacado em negrito. Exceto pelo DARN, o MS-DIAL supera todas as demais abordagens comparadas em todas as tarefas de MSDA. Apesar do DARN alcançar uma acurácia média de classificação ligeiramente superior a do MS-DIAL para alguns domínios-alvos, o MS-DIAL tem um desempenho médio melhor que o DARN tanto para o reconhecimento de dígitos quanto de objetos.

V. CONCLUSÃO

Neste trabalho, foi apresentado o MS-DIAL, uma nova abordagem para o problema de MSDA. Nesse método, camadas de alinhamento de domínio são inseridas em diversos níveis da rede e, assim, o alinhamento entre as distribuições dos domínios-fonte e do domínio-alvo é realizada de forma automática a partir de parâmetros aprendíveis em tempo de treinamento. Dessa forma, os parâmetros das demais camadas da rede podem ser compartilhados entre todos os domínios, otimizando o tempo de processamento e uso de memória.

O MS-DIAL foi avaliado em tarefas de reconhecimento de dígitos e de objetos com cinco conjuntos de dados públicos amplamente usados para avaliar métodos de UDA. Os resultados obtidos com uso das camadas MS-DIAL foram promissores e superaram, em média, abordagens do estado da arte.

Em trabalhos futuros, pretende-se avaliar o MS-DIAL em outros conjuntos de dados. Além disso, pretende-se também investigar o uso do MS-DIAL em outras tarefas desafiadoras, como adaptação de domínio de conjunto aberto (do inglês, *open set domain adaptation* – OSDA) e generalização de domínio (do inglês, *domain generalization* – DG).

AGRADECIMENTOS

Este trabalho foi apoiado pela Fundação de Amparo à Pesquisa do Estado de São Paulo - FAPESP (processos 2017/25908-6 e 2019/10998-5) e pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (processos 423228/2016-1, 313122/2017-2 e 167857/2019-3).

REFERÊNCIAS

[1] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F.-F. Li, "Imagenet large scale visual recognition challenge," *IJCV*, vol. 115, no. 3, pp. 211–252, 2015.

[2] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *CVPR*, 2017, pp. 5385–5394.

[3] F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulò, "Just DIAL: domain alignment layers for unsupervised domain adaptation," in *Int. Conf. Image Analysis and Processing*, 2017, pp. 357–369.

[4] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin, "Deep cocktail network: Multi-source unsupervised domain adaptation with category shift," in *CVPR*, 2018, pp. 3964–3973.

[5] S. Zhao, B. Li, X. Yue, P. Xu, and K. Keutzer, "MADAN: multi-source adversarial domain aggregation network for domain adaptation," *CoRR*, vol. abs/2003.00820, 2020.

[6] J. Wen, R. Greiner, and D. Schuurmans, "Domain aggregation networks for multi-source domain adaptation," in *ICML*, 2020, pp. 10 927–10 937.

[7] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[8] Y. Ganin and V. S. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *ICML*, 2015, pp. 1180–1189.

[9] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS Work. Deep Learning and Unsupervised Feature Learning*, 2011.

[10] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. S. Lempitsky, "Domain-adversarial training of neural networks," *JMLR*, vol. 17, pp. 59:1–59:35, 2016.

[11] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Networks*, vol. 22, no. 2, pp. 199–210, 2011.

[12] M. Baktashmotlagh, M. T. Harandi, and M. Salzmann, "Distribution-matching embedding for visual domain adaptation," *JMLR*, vol. 17, pp. 108:1–108:30, 2016.

[13] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo, "Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation," in *CVPR*, 2017, pp. 945–954.

[14] F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulò, "Autodial: Automatic domain alignment layers," in *ICCV*, 2017, pp. 5077–5085.

[15] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *ICCV*, 2019, pp. 1406–1415.

[16] Y. Li, M. Murias, G. Dawson, and D. E. Carlson, "Extracting relationships by multi-domain matching," in *NeurIPS*, 2018, pp. 6799–6810.

[17] H. Zhao, S. Zhang, G. Wu, J. M. F. Moura, J. P. Costeira, and G. J. Gordon, "Adversarial multiple source domain adaptation," in *NeurIPS*, 2018, pp. 8568–8579.

[18] S. Zhao, G. Wang, S. Zhang, Y. Gu, Y. Li, Z. Song, P. Xu, R. Hu, H. Chai, and K. Keutzer, "Multi-source distilling domain adaptation," in *AAAI Conf. Artificial Intelligence*, 2020, pp. 12 975–12 983.

[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, F. R. Bach and D. M. Blei, Eds., 2015, pp. 448–456.

[20] P. Arbelaez, M. Maire, C. C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *T-PAMI*, vol. 33, no. 5, pp. 898–916, 2011.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[22] S. Roy, A. Siarohin, E. Sangineto, S. R. Bulò, N. Sebe, and E. Ricci, "Unsupervised domain adaptation using feature-whitening and consensus loss," in *CVPR*, 2019, pp. 9471–9480.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

Water Tanks and Swimming Pools Detection in Satellite Images: Exploiting Shallow and Deep-Based Strategies

Eduardo A. M. Fernandes, Pedro F. Wildemberg, Jefersson A. dos Santos
Department of Computer Science
Universidade Federal de Minas Gerais
Belo Horizonte, Brazil
Email: {eduardofernandes,pedrowildemberg,jefersson}@dcc.ufmg.br

***Abstract*—This paper aims to study and to evaluate two distinct approaches for detecting water tanks and swimming pools in satellite images, which can be useful to monitor water-related diseases. The first approach, shallow, consists of using a Support Vector Machine in order to classify into *positive* and *negative* a discretized color histogram of a given segment of the original image. The second method employs the Faster R-CNN framework for detecting those objects. We built up swimming pools and water tanks datasets over the city of Belo Horizonte to support our experimental analysis. Our results show that the deep learning method greatly outperforms the shallow strategy, achieving an average precision at 0.5 IoU of over 93% on the swimming pool detection task, and over 73% on the water tank one. All the code and datasets are publicly available.**

***Index Terms*—Remote Sensing, Swimming Pool, Water Tank, Detection, Deep-Learning, SVM**

I. Introduction

Remote sensing is of great importance for vector-borne disease control, since, through unmanned aerial vehicles (UAV) and satellite imagery, it can provide fast, precise, and large-scale surveillance of areas infected by the vector. This information can, then, be used by health officers to locate the agent's breeding sites and determine where it should act to best combat the infection [1].

When it comes to mosquitoes, responsible for transmitting a variety of deadly and expanding diseases across the world with millions of cases registered every year, the main breeding spots are containers and debris holding still water pools, such as tires, plant vases, bottles, water tanks and poorly maintained swimming pools [2] [3]. More specifically, the last two are usually large enough to be spotted in satellite images, making them possible objects of interest when applying remote sensing techniques to detect the breeding sites of such species. On this subject, with the advent of Deep Learning (DL) techniques for image segmentation and object detection, heavily dependent on GPUs usage, Shallow Learning (SL) methods, mostly CPU demanding, are becoming more obsolete every day for said problems. A big advantage of the former method is the automation of the feature extraction step when attempting to classify an image or detect an entity in it, whereas, in the latter, it is the data scientist responsibility to identify and design visual feature extraction strategies to better represent the image for the task [4]. Moreover, in recent years, DL methods have been outperforming SL ones in several computer vision problems in terms of accuracy, leading to a major shift in approaches used for solving such tasks.

With water tanks and swimming pools as targets, this paper aims to compare the performance of SL and DL approaches for object detection. Such analysis is of great importance in this scenario, since such mosquito-borne diseases are a life threatening problem in least developed and developing countries, where the choice for a computational application can be heavily influenced by the available budget for hardware expenses. Accordingly, two new datasets were assembled, containing thousands of annotated swimming pools and water tanks in high resolution satellite images. Our datasets and analysis of two well-established approaches aim to serve as a new starting point for works related to the detection of mosquitoes breeding spots [2], [3], focusing on the trade-off between computational complexity and performance, and all the code has been made publicly available [1].

II. Related Work

Literature contains a variety of papers related to both the topic of detecting swimming pools in satellite images and comparing SL and DL methods in several different tasks. Specifically in the former problem, some papers stand out for the use of shallow methods to try and solve the task and the motivations that led them to do so.

Tien, Rudra & Hope employed a support vector machine (SVM) [5] to detect swimming pools in satellite imagery, by calculating the difference between the blue-red and blue-green combinations of all pixels and feeding this information to the machine. The goal of this work was to locate water bodies in Australia and help in bushfire fights in the country [6].

In [7], McFeeters proposes the Normalized Difference Water Index (NDWI) [7] to delineate open water features in aerial imagery, by making use of the near-infrared and green bands of the given image, and in [8], Kim, Holt, Eisen, Padgett, Reisen,

[1] https://github.com/EduardoFernandes1410/PATREO-Dengue

& Crof integrate the index with the rectangular fit space metric [9], proposing to delineate water features in aerial images in order to identify pools and aiming to assist in the control of the Culex mosquito population, vector of the West Nile Virus, in the United States of America. The method achieved a user's accuracy of 92.8% in pool negative samples and 80.1% in pool positive ones. Moreover, Alonso and Rodríguez-Cuenca described and used the Normalized Difference Swimming Pools Index (NDSPI) to semi-autonomously detect swimming pools, alongside with region adjacency graph (RAG) and principal component analysis (PCA) [10], scoring an overall accuracy of 99.86% according to the authors [11].

When it comes to comparing shallow and deep learning methods in tasks from different areas of study, Pasupa & Sunhem tested the performance of a convolutional neural network (CNN) [4] and a SVM in a classification problem with a small dataset, and found that, without applying data-augmentation techniques, the latter performed better then the former. However, when making use of such techniques to improve the dataset, the CNN method presented results that were competitive to the shallow method [12].

In [13], Koutsoukas, Monaghan, Li & Huan compare the performance of a deep neural network (DNN) [4], Naïve Bayes [14], K-nearest neighbours [15], random forest [16] and SVM methods for the problem of modeling bio-activity data, and conclude that DNNs, with optimal hyper-parameters and low noise levels, outperforms every other method applied.

Finally, Liu, Abd-Elrahman, Morton, & Wilhelm compared CNNs, random forest and SVM in a remote sensing task to map wetlands from a object-based level [17]. In agreement to [12], the authors found that the shallow learning methods performed better than the deep learning ones when the training dataset was small, but once more training samples were used, the latter obtained the superior results.

III. Assembly of the Dataset

Two separate datasets were assembled in this work: BH-Pools and BH-WaterTanks, with annotated swimming pools and water tanks respectively, and can be freely downloaded in this link [2]. Both datasets consist of imagery from several neighbourhoods in the city of Belo Horizonte, Minas Gerais, Brazil. The data was acquired through the Google Earth Pro tool. The images were exported from an eye altitude of 330 meters with a resolution of 3840x2160 (4K), and the image bands are the three visible ones: red, green and blue. For each occurrence of the target objects found on the images, a polygon was drawn in order to generate the segmentation masks of the instance. For the detection problem, a bounding box for each of the annotated objects was calculated through said masks. Fig. 1 and Fig. 2 show a few examples of the images and ground-truths in BH-Pools and BH-WaterTanks, respectively, and Table I summarizes the datasets specifications.

[2]http://www.patreo.dcc.ufmg.br/bh-pools-watertanks-datasets/

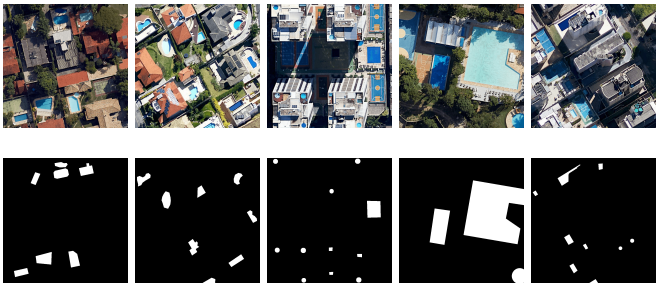


Fig. 1: Example crops from BH-Pools

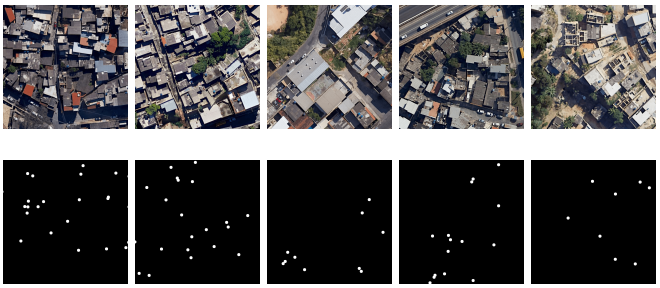


Fig. 2: Example crops from BH-WaterTanks

A. Data Preparation

Each 4K image was cropped into 6 smaller ones of size 1280x1080, without overlap, and then the ones without annotations were removed. Afterwards, 80% of the images from each neighbourhood were used as a training dataset, as the other 20% were used as a test dataset. The same preparation was made for the images of BH-Pools and BH-WaterTanks. These prepared datasets were used for both the SL and DL methods.

B. BH-Pools

The BH-Pools dataset consists of 200 4K images of 8 different neighbourhoods (25 images for each one) and contains 3980 annotated pools. The data preparation step resulted in 655 images designated for training and 160 for testing.

C. BH-WaterTanks

The BH-WaterTanks dataset is made up of 150 4K images of 6 neighbourhoods (25 images for each one) and contains 16216 annotated water tanks. The data preparation step resulted in 608 cropped images designated for training and 148 for testing.

IV. Shallow-Based Approach

A. Methodology

The shallow-learning-based method consists of several different steps, ranging from feature extraction processes to learning and prediction ones, as illustrated in Fig. 3. In contrast to the deep-learning method, the feature extraction strategies had to be selected and designed manually for this approached,

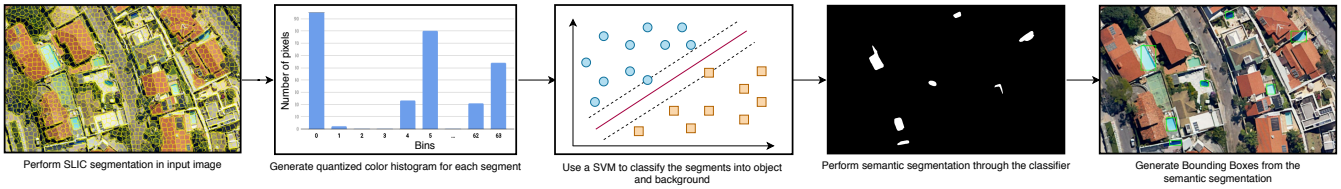


Fig. 3: Diagram illustrating the shallow learning method

TABLE I: BH-Pools and BH-WaterTanks specifications

	BH-Pools	**BH-WaterTanks**
Source	Google Earth Pro	Google Earth Pro
Image resolution	3840x2160	3840x2160
Eye altitude (meters)	330	330
Image bands	RGB	RGB
Number of images	200	150
Number of annotated objects	3980	16216

focusing on what visual features of any given image would better represent the target objects, whereas, in the other, this process is completely automated.

1) SLIC: The first step of the object detection process is to divide the original image into *superpixels* through the SLIC algorithm [18]. Due to the resolution of the images in the two datasets and the average size of the target objects in each of them, the parameters chosen were: 2,000 desirable labels for BH-Pools images and 10,000 ones for BH-WaterTanks. For both datasets, the sigma value was set to 5. The implementation used was the one available in the *scikit-image* toolkit [19].

2) Color Histogram: Moving forward, to each pixel of the image it was attributed an integer number between 0 and 63, accordingly to (1), so that pixels with similar RGB values are allocated in the same group. This way, the number of values describing a given pixel is reduced from three to one, simplifying the classification process down the pipeline.

$$\nu = \frac{\rho}{64} + 4 \times \frac{\gamma}{64} + 16 \times \frac{\beta}{64} \quad (1)$$

where ν = number attributed to pixel
ρ = value of the red channel
γ = value of the green channel
β = value of the blue channel

Afterwards, the histogram of each segment generated by the SLIC algorithm was obtained, based on the new number calculated for every individual pixel, containing 64 bins representative of the color distribution of the given superpixel. This information is stored in a matrix, where the rows represent an individual segment and the columns 0 to 63 contain the number of pixels with that value on the segment.

3) Support-vector Classifier: In the next step, a Linear Support-vector Classifier (SVC) was used to classify the segments of the input image into *positive* or *negative*, using the color histogram obtained in the previous step. The implementation of the SVC used was the one available in the *scikit-learn* [20] toolkit.

Therefore, the batch of images designated for training in each dataset was used for fitting the classifier. It was decided that a given segment is representative of a target object if 50% or more of it is annotated as positive. Moreover, a standardization of the data was performed, in order to center the features around 0 and make them have unit-variance.

Lastly, the trained classifier was applied on the testing batch, in order to perform the semantic segmentation of the target objects in the images. The SVC outputs a confidence score for each given segment, which represents the signed distance of that sample to the hyperplane that separates the two classes. In this case, if said distance is greater than zero, than the segment belongs to the *positive* class.

4) Bounding Boxes: Once the semantic segmentation is completed, another operation is performed in order to obtain the bounding boxes of the detected objects, alongside with their confidence scores. For that, the Multi-dimensional Image Processing packet from the SciPy ecosystem was used to aggregate individual positive segments which are next to each other and that, combined, represent a full object. The bounding box coordinates would then be calculated over such objects. Finally, the confidence score of each bounding box was defined as the average of the confidence scores of the segments that compose it.

B. Hardware and Software Setup

This method was implemented using the Python (version 3.6) programming language. The experiment was performed on a Intel i7-5930X machine with 3.50GHz of clock and 64GB of RAM.

V. Deep-based Approach

A. Framework

The deep learning method for water tanks and swimming pools detection is based on the Faster R-CNN framework [21]. For that, it was used the implementation from the TorchVision package.

Faster R-CNN unifies in one architecture the module consisting of a Region Proposal Network (RPN) and the module responsible for object detection (Fast-RCNN [22]), as illustrated in Fig. 4. In simple terms, a CNN receives the input image and then provides a feature map, which is used by the RPN to indicate to the Fast-RCNN where to look for objects. From this information, a series of Fully Connected layers make

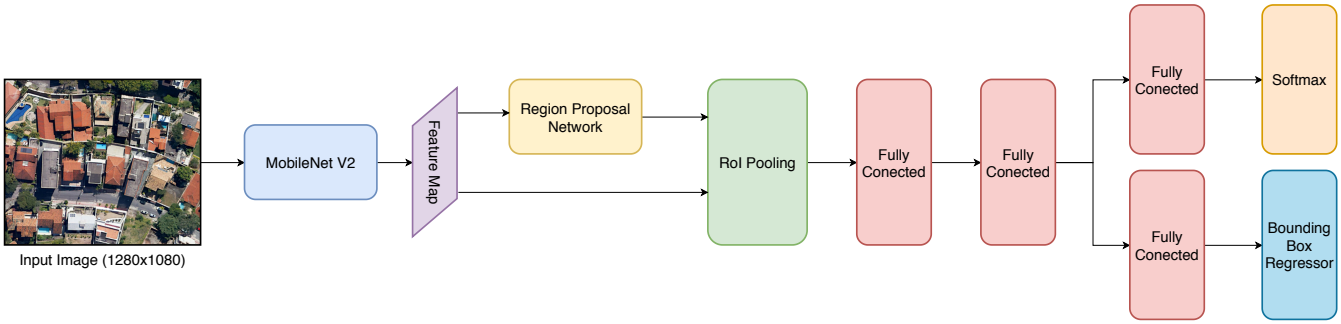


Fig. 4: Simplified diagram illustrating the Faster R-CNN architecture

predictions of the location of bounding boxes in the image and their respective labels.

B. Methodology

The feature extraction network utilized as backbone was the MobileNetV2 [23], pre-trained on the COCO dataset [24]. By using pre-trained weights of a CNN, this DL method performs a fine tuning to improve the training process speed. The MobileNetV2 is a light weight model with improvements suitable for the use of on-device computer vision deep applications, like in mobile devices or any device with low computational power. This choice of backbone helps possible future applications with this deep learning method to not be compromised by a low-budget hardware.

For the training step, it was used the Adam optimizer and a learning rate of 0.0001, chosen empirically. The model was trained for 50 epochs with a batch size of 4 and a random horizontal flip (0.5) transform in the data loader.

Lastly, the testing step was performed on both a GPU-available environment and a CPU-only one. This way, it would be possible to evaluate the possibility of training the network on a high-end device to reduce its training time, but deploying it on inexpensive machines to reduce its cost in regions with budget limitations.

C. Hardware and Software Setup

This method was implemented using Python (version 3.6) with the Pytorch library. The experiment was performed on the Google Colaboratory (Colab) platform, with a configuration composed of an Intel Xeon processor (not specified) with 2.3 GHz of clock, 13 GB RAM and a Tesla T4 GPU.

VI. Results and Discussion

In order to best evaluate the performance of both methods on the proposed datasets, two standard metrics were chosen to be calculated: Average Precision (AP) at an IoU of 0.5, used in the PASCAL VOC challenge, and AP averaged over 10 IoU values, ranging from 0.5 to 0.95 with a step of 0.05, adopted by the COCO challenge [24].

AP takes into account the true predicted positives / total predicted positives (precision) and true predicted positives / total real positives (recall) ratios. An intersection over union (IoU) threshold is defined to determine if a prediction is a true positive or a false positive one. The IoU measures how much the predicted bounding box overlaps with the ground truth bounding box annotation.

The AP with IoU of 0.5 would indicate if the method applied is simply able to correctly detect the target objects in an image, whereas the AP with IoU of 0.5:0.05:95 (the average AP for IoU threshold from 0.5 to 0.95 with a step size of 0.05) would measure how precisely the technique is able to draw the bounding boxes around such objects. Alongside with these metrics, the training and testing times were also measured for each approach, so that it would be possible to compare their usability in real-case scenarios.

All the obtained results of the evaluated methods are presented in Table II. Moreover, Fig. 5 and Fig. 6 show qualitative results for each technique, illustrating the differences between them.

The deep learning method performed significantly better in both datasets compared to the shallow learning one according to the metrics used. On the BH-Pools dataset, the latter was able to detect many of the swimming pools present, but usually only their brightest parts, leading to low IoU values, and also made a lot of False Positive predictions. Conversely, the deep method achieved very high results, scoring an AP at IoU=0.50 equal to 2.27x the one achieved by the other, and an AP at IoU=0.50:0.05:0.95 3.06x the one scored by its opponent, indicating that it not only detected more swimming pools, but also drew more precise bounding boxes around them. Meanwhile, the shallow method obtained extremely poor results on BH-WaterTanks, failing to detect almost every single water tank in it, in contrast to the deep method, which presented satisfactory results. However, the AP at IoU=0.50:0.05:0.95 scored by the deep-learning approach was still lower than half of its score with an IoU=0.50, indicating a greater difficulty in precisely delineating those objects.

Finally, the DL techniques were able to perform the training and testing steps many times faster than the SL ones, including completing the test phase in a matter of minutes, even on CPU-only machines, as opposed to the several hours needed for the others.

VII. Conclusion

In this work, we evaluated and compared two approaches for swimming pool and water tanks detection. The shallow method

TABLE II: Results obtained on the proposed datasets using different object-detection approaches

(a) BH-Pools dataset

Method	AP at IoU=0.50 (%)	AP at IoU=0.50:0.05:0.95 (%)	Training Time (Hours)	Testing Time (Hours) (CPU)	Testing Time (Hours) (GPU)
Shallow learning	40.97	15.96	8.02	1.10	-
Deep learning	**93.13**	**64.79**	**2.70**	**0.08**	**0.01**

(b) BH-WaterTanks dataset

Method	AP at IoU=0.50 (%)	AP at IoU=0.50:0.05:0.95 (%)	Training Time (Hours)	Testing Time (Hours) (CPU)	Testing Time (Hours) (GPU)
Shallow learning	0.13	0.03	19.00	4.20	-
Deep learning	**73.43**	**32.99**	**4.63**	**0.09**	**0.02**

(a) Ground Truth (b) Shallow Method (c) Deep Method

Fig. 5: Example comparing swimming pools detection by the shallow and deep learning methods

(a) Ground Truth (b) Shallow Method (c) Deep Method

Fig. 6: Example comparing water tanks detection by the shallow and deep learning methods

consists of a segmentation using SLIC [18] followed by a classification with SVM [5]. The deep-based approach consists of a Faster-RCNN framework [21] with MobileNetV2 backbone [23]. The methods were trained and then evaluated with our two proposed datasets: BH-Pools and BH-WaterTanks. Both metrics and visual results were compared for a final analysis.

It was clear that the shallow method did not work well for the water tanks detection. Residential water tanks are relatively small objects in satellite images and they could not get a precise segmentation with SLIC, compromising the rest of this approach. On the other hand, the shallow-based approach performed well with the swimming pools detection. Swimming pools are considerably larger and could get a more precise segmentation. However, unusual pools formats, shady areas and blue geometric terrain (e.g. sports courts) were easily misclassified.

The deep learning method worked really well with the water tanks detection, despite their small size. Moreover, it increased the precision of the swimming pool detection. For this reason, we can infer that the deep method used the spatial context more wisely. This method performed better, faster and has been shown of great potential for the task of water tanks and swimming pools detection in high resolution satellite images in practical applications, being a great option even on environments where a GPU is not available to perform the prediction step, given that the network has been trained previously.

Finally, when it comes to the detection of swimming pools, the shallow-learning method can still be a reasonable option if a powerful enough GPU is not available to train the deep-learning network. The method is able to detect many of the swimming pools presented to it in a fraction of the time it would take for a human operator to do so, and does not require a highly computationally complex machine, demonstrating its usefulness in this remote sensing task. Unfortunately, the same cannot be said about residential water tanks detection, due to their smaller size in satellite images.

REFERENCES

[1] S. Kalluri, P. Gilruth, D. Rogers, and M. Szczur, "Surveillance of arthropod vector-borne infectious diseases using remote sensing techniques: a review," *PLOS Pathogens*, 2007.

[2] M. A.Tolle, "Mosquito-borne diseases," *Current Problems in Pediatric and Adolescent Health Care*, vol. 39, pp. 97–140, 2009.

[3] F. Chiaravalloti Neto and H. A. d. S. L. Pereira, "Aedes aegypti na região de são josé do rio preto, estado de são paulo," Master's thesis, Universidade de São Paulo, 1993.

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[5] C. Cortes and V. Vapnik, "Support-vector networks," *V. Machine Learning*, pp. 273–297, 1995.

[6] D. Tien, T. Rudra, and A. Hope, "Swimming pool identification from digital sensor imagery using svm," 01 2008, pp. 523–527.
[7] S. K. McFEETERS, "The use of the normalized difference water index (ndwi) in the delineation of open water features," *International Journal of Remote Sensing*, vol. 17, no. 7, pp. 1425–1432, 1996.
[8] M. Kim, J. Holt, R. Eisen, K. Padgett, W. Reisen, and J. Croft, "Detection of swimming pools by geographic object-based image analysis to support west nile virus control efforts," *Photogrammetric Engineering and Remote Sensing*, vol. 77, pp. 1169–1179, 11 2011.
[9] J. A. Saghri and D. A. Cary, "A rectangular-fit classifier for synthetic aperture radar automatic target recognition," in *Applications of Digital Image Processing XXX*, A. G. Tescher, Ed., vol. 6696, International Society for Optics and Photonics. SPIE, 2007, pp. 511 – 521.
[10] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.
[11] M. Alonso and B. Rodríguez-Cuenca, "Semi-automatic detection of swimming pools from aerial high-resolution images and lidar data," *Remote Sensing*, vol. 6, pp. 2628–2646, 04 2014.
[12] K. Pasupa and W. Sunhem, "A comparison between shallow and deep architecture classifiers on small dataset," *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2016.
[13] A. Koutsoukas, K. J. Monaghan, X. Li, and J. Huan, "Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data," *Journal of Cheminformatics*, vol. 9, no. 1, 2017.
[14] A. McCallum, K. Nigam *et al.*, "A comparison of event models for naive bayes text classification," in *AAAI-98 workshop on learning for text categorization*, vol. 752, no. 1. Citeseer, 1998, pp. 41–48.
[15] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.
[16] V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston, "Random forest: a classification and regression tool for compound classification and qsar modeling," *Journal of chemical information and computer sciences*, vol. 43, no. 6, pp. 1947–1958, 2003.
[17] T. Liu, A. Abd-Elrahman, J. Morton, and V. L. Wilhelm, "Comparing fully convolutional networks, random forest, support vector machine, and patch-based deep convolutional neural networks for object-based wetland mapping using images from small unmanned aircraft system," *GIScience & Remote Sensing*, vol. 55, no. 2, pp. 243–264, 2018.
[18] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels," *Technical report, EPFL*, 06 2010.
[19] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors, "scikit-image: image processing in Python," *PeerJ*, vol. 2, p. e453, 6 2014.
[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2015.
[22] R. Girshick, "Fast r-cnn," 2015.
[23] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2018.
[24] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

Braille character detection using deep neural networks for an educational robot for visually impaired people

Diego Gonçalves*, Gabriel G. Santos*, Márcia B. Campos†, Alexandre M. Amory*, Isabel H. Manssour*
* School of Technology - PUCRS, Porto Alegre, Brazil
Email: dlngoncalves@gmail.com, gabriel.giordani@acad.pucrs.br, amamory@gmail.com, isabel.manssour@pucrs.br
†CESUCA University Center, Cachoeirinha, Brazil
Email: marcia.campos@cesuca.edu.br

***Abstract*—Teaching computer programming to the visually impaired is a difficult task that has sparked a great deal of interest, in part due to its specific demands. Robotics has been one of the strategies adopted to help in this task. One system that uses robotics to teach programming for the visually impaired, called Donnie, has as its key part the need to detect Braille characters in a scaled-down environment. In this paper, we investigate the current state-of-the-art in Braille letter detection based on deep neural networks. For such, we provide a novel public dataset with 2818 labeled images of Braille characters, classified in the letters of the alphabet, and we present a comparison among some recent detection methods. As a result, the proposed Braille letters detection method could be used to assist in teaching programming for blind students using a scaled-down physical environment. The proposal of EVA (Ethylene Vinyl Acetate) pieces with pins to represent Braille letters in this environment is also a contribution.**

***Index Terms*—Educational Robotics, Assistive Robotics,Deep Neural Networks, Braille.**

I. Introduction

There is a great concern of the community in developing new ways to teach programming to the visually impaired learners [1]–[3]. Several strategies have been proposed in the last years to make the programming learning process more accessible for these learners. Among several strategies that have been adopted, as the use of physical artifacts to provide tactile information [3], robotics has been used as a tool to assist in this task [4]–[7].

In previous work [8], we presented a new educational/assistive robot called Donnie and its programming environment, which allows both the practice of computational thinking and the training of orientation and mobility skills that is inclusive for visually impaired people[1].

We gratefully acknowledge the support of NVIDIA Corporation with the donation of Titan Xp GPU used for this research. The result of this work was achieved in cooperation with HP Brasil Indústria e Comércio de Equipamentos Eletrônicos LTDA. using incentives of Brazilian Informatics Law (Law nº 8.2.48 of 1991). This work was supported by the CNPq/MCTIC/SECIS Nº 20/2016, National Council for Scientific and Technological Development – CNPq. It was also partially supported by PUCRS (Edital 01/2018 - Chamada Geral).

[1]Extensive documentation and manuals about Donnie are available at https://github.com/lsa-pucrs/donnie-assistive-robot-sw

It's well known that visually impaired people have orientation and mobility difficulties when facing a new environment. Let us assume a scenario where a blind student moves to a new school. One way this project can help the student is by modeling a scaled-down version of the school's floorplan. Then the student can practice some programming skills to perform mobility challenges in this scenario. For instance, the student can practice going from his classroom to the bathroom. The shorter distance he can move the robot, the more points are earned. In parallel, the student is also learning how to move independently in this new environment. However, there are some challenges to reach this goal. How can the blind student know he is driving the robot to the correct direction? How can the student know that he is getting closer or reached the goal? The approach chosen for this project is to place markers in the environment that will give hints to the students of the robot's location.

When executing the virtual scenario by software simulation, this task of object or place identification can be easily accomplished by using simulated fiducial markers or tags spread in the virtual environment. However, when using the real robot, the same task is much more complex. It is necessary to set up an environment with objects that must be recognized by the robot. For this reason, Donnie has a camera on its head with horizontal movement, to search for objects.

Using normal markers such as QR Code would be easy for the robot perception, but it is not useful for blind users that need to assemble and identify the objects in a dynamic physical environment. One of the goals of the project was accessibility for both users of the simulator and of the actual robot. Thus, the alternative is to build markers using the Braille alphabet, which can be both felt by the users and detected by the robot camera, as exemplified in Figure 1.

In this paper, we present a new approach for the detection of markers in the Braille alphabet using Convolutional Neural Networks to help in the assistive physical environment of the Donnie robot. Our main contributions include a novel public dataset [2] with 2818 labeled images of Braille characters, classi-

[2]https://www.kaggle.com/dlngoncalves/donnie-braille

fied in the letters of the alphabet; a proposal of EVA (Ethylene Vinyl Acetate) pieces with pins to represent Braille characters in the physical environment that need to be recognized; and a comparison among the use of recent methods for detection of Braille alphabet.

The remainder of this paper is organized as follows. We briefly describe some related work in Section II. Section III describes our novel dataset. The detection methods we employ are presented in Section IV. Our results and main findings regarding the suitability of employing state-of-the-art deep neural approaches for Braille character recognition are described in Section V. Finally, we end this paper with our conclusions and future work directions in Section VI.

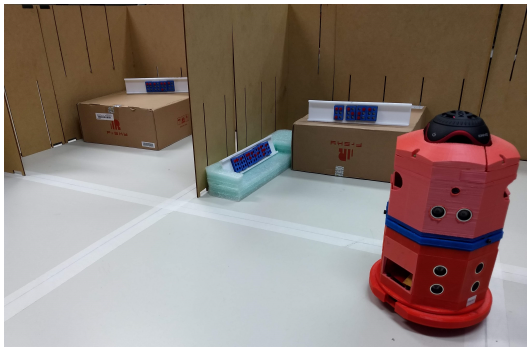


Fig. 1. Example of the robot in a real environment. The Braille markers are in blue with dots in red.

II. Background

Optical Braille Recognition (OBR) is the sequence of steps involved in converting the contents of images of Braille text documents into natural language. A description of the Braille script and a general methodology for OBR is presented by Isayed and Tahbou [9]. Their work compares different OBR techniques based on criteria such as image acquisition and Braille dots detection techniques. The authors notice that most of the techniques found in literature use scanner image acquisition, with few using mobile or standard cameras. They also observed a lack of artificial intelligence approaches as neural networks. Later, a review of some OBR methods that use a camera for image acquisition was presented by Nugroho et al. [10].

One early approach that used a multi-layer perceptron neural network for Braille character recognition was presented by Morgavi and Morando [11]. Another early work [12] used image processing techniques and probabilistic neural networks to recognize and transcribe Braille documents. Zhang and Yoshino [13] use images acquired by a mobile phone with image processing techniques for the recognition of Braille used in Japan. Li and Yan [14] used Support-Vector Machine to recognize Braille characters from an image. Artificial neural networks were also recently used by Waleed [15] to identify numbers in Braille from images.

Recently, deep learning methods have been used for OBR, such as the stacked denoising autoencoder proposed by Li et al. [16], which achieved good results compared to traditional methods. Shimomura et al. [17] also used deep learning techniques to convert Braille books into machine-readable electronic data.

Despite the different detection techniques used, almost all the works listed above use images acquired by traditional flat-bed scanners. Moreover, few of them use recent deep learning techniques. In this work, we use images acquired by a common camera (thus the Braille characters are viewed from different angles) and deep neural networks, which have shown promising results for OBR.

III. A Novel Braille Dataset

To the best of our knowledge, there is no public dataset of images of Braille characters, at least not one large enough to be used with machine learning techniques. In the review of OBR techniques presented by Isayed and Tahboub [9], the authors mention that there is no benchmark to test the algorithms and researchers use their database and Braille images to measure performance.

Thus, for this work, a dataset of images of Braille characters was produced, with the goal of using it to train a convolutional neural network for the detection and recognition of such characters. The images produced were of cropped individual letters of the Portuguese Braille alphabet. Some of such images used in this training are shown in Figure 2.

(a) (b) (c)

Fig. 2. A selection of Braille character images. (a) Braille character *a*. (b) Braille character *f*. (c) Braille character *r*.

For the Donnie project, it was necessary to be able to visually detect and classify Braille characters in a scaled-down real-world environment, with the robot's camera being used for image acquisition, as exemplified in Figure 1. The robot should be able to detect characters in distances up to a meter. Additionally, scenes needed to be dynamic and easily modifiable by users without additional difficulties for the visually impaired. However, this detection of Braille characters in a real-world environment proved to be challenging, because they are small, usually showing very little visual contrast with their backgrounds, making them hard to detect. Some characters are also very similar, a characteristic that makes them hard to classify from a distance when relying only on visual information.

Considering these problems and the needs of the Donnie project, we designed and produced small acrylic pieces that can be used to represent Braille characters, as shown in Figures 3(a) and 3(b). The pieces are larger and have significantly more contrast with the environment than traditional Braille

characters printed on paper. Moreover, they facilitate the task of identifying objects by blind users.

(a) (b) (c)

Fig. 3. Types of acrylic Braille pieces. (a) Pins representing characters. (b) Magnets representing characters. (c) EVA Braille pieces with acrylic pins.

We used smartphones and the robot camera to collect images to compose the dataset. The images contained the pieces in different backgrounds and positions. Thus, we were able to capture several images of the Braille character from different angles and with lighting variations more quickly. Since the camera of the Donnie robot uses a resolution of 640×480, the images acquired with smartphones were resized to this size by a Python script before annotation. To annotate the images, we used the software labelImg[3]. In the end, we had a total of 26 classes, one for each letter.

Initially, the dataset was composed of images of two types of Braille pieces, both containing a magnetic layer overlaid with acrylic and with six holes. The first type showed in Figure 3(a), used acrylic colored pins attached to magnets to fit into the holes and form the characters. The second one showed in Figure 3(b), used only magnets to fit into the holes. This dataset had 3527 images but was incomplete since 6 letters did not have any instances. The testing phase showed that the neural network couldn't identify any instance of some letters.

Since the results weren't satisfactory, we decided to compose a dataset where all the pieces had similar pins. However, after some visually impaired people had manipulated the Braille pieces, they suffered changes in its design due to the difficulty encountered to use the magnets to compose the characters. The new pieces, shown in Figure 3(c), have the background part made of EVA, with acrylic pieces of a different color for the pins used to fit into the holes and represent the characters. This configuration was preferred by visually impaired users with regards to usability and accessibility after they performed some writing and reading activities with the pieces. The colors were chosen based on preliminary detection results.

To compose the new dataset, we removed all images that had pieces that only used magnets to represent characters. After the removal, the dataset remained with 991 images of pieces that used colored pins attached to magnets. Then, we added images of the new EVA pieces so that every letter had at least 100 different images. At the end of the process, the data set was with 2818 images separated in the 26 letters of the alphabet.

[3]https://github.com/tzutalin/labelImg

After several tests, we noticed that videos containing words composed by the Braille pieces presented worse results than the videos with individual Braille pieces. Thus, we decided to add annotated images of sequences of characters to the dataset. These images presented the Braille pieces side by side in a white acrylic base, as shown in Figure 4. A total of 217 images were annotated, with sequence length varying between 4 and 10 characters.

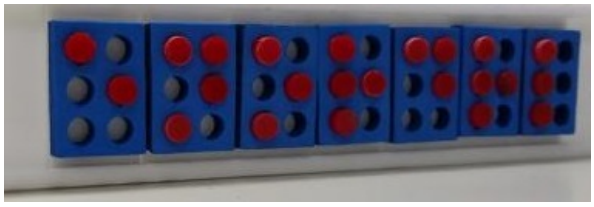


Fig. 4. Example of a sequence of pieces.

For the sake of testing false positives an additional 32 images were captured with the following scenarios:

- Pieces with configurations that do not represent a valid character;
- Two pieces forming one character using one column of each piece;
- Pieces with no configuration;
- Acrylic pins forming a character out of the base.

A sample of those scenarios is shown in Figure 5.

(a) (b) (c)

Fig. 5. Invalid configurations. (a) One character in two pieces. (b) Empty background. (c) Pins without a background.

IV. Detection Methods

In recent years Convolutional Neural Networks (CNN) have become very successful in image classification and object detection tasks, as shown in one pioneer work [18] that won the 2012 ImageNet image classification competition. They are one of the deep learning methods, which has many convolutional layers. In this work, we used state-of-the-art deep learning techniques for object detection and classification on our custom Braille dataset, and we evaluated their performances. The used techniques were Mask R-CNN and YOLOv2. In the next sections, we briefly describe and compare these techniques.

A. *Mask R-CNN*

Mask R-CNN [19] is a framework for object instance segmentation that extends the Faster R-CNN [20]. Both are based on the Region-based Convolutional Neural Network (R-CNN) [21] approach to bounding-box object detection.

R-CNN first generates regions of interest for potential bounding boxes and then runs a classifier on those regions. Faster R-CNN consists of two stages: in the first one occurs the proposition of candidate bounding boxes; the second stage use region of interest pool to extract features and perform classification and bounding box regression.

Mask R-CNN extends Faster R-CNN by adding the calculation of a binary mask to each region of interest. The calculation of the binary mask occurs in parallel with the second stage of Faster R-CNN, and the first stage continues the same.

B. YOLO - You Only Look Once

YOLO [22] presents a new approach to object detection. It treats object detection as a regression problem, using a single convolutional network to simultaneously predict bounding boxes and generate class probabilities for those boxes. This is done by analyzing the entire image during training, dividing it into a grid, and for each grid cell predicting a number of bounding boxes, the confidence score of those boxes, and a set of class probabilities.

Using this approach YOLO can achieve good results in terms of precision metrics, being the fastest object detector. One downside of the YOLO approach is that it has lower precision than other methods. Also, the spatial constraints imposed by the small number of bounding boxes predicted by each grid cell limit the number of nearby objects YOLO can predict. It also struggles with small objects [22].

V. Experimental Analysis

To validate the use of our dataset in the task of training models for detecting Braille characters and use on the Donnie project, we performed experiments with both the YOLO and Mask R-CNN approaches on it. The training was carried out on a dual Intel Xeon E5-2620 system, with 48GB of RAM, and an NVIDIA Titan Xp GPU with 12GB of VRAM.

For better initialization, we used pre-trained weights from ImageNet in the networks' backends. We also used data augmentation techniques to enlarge our dataset, such as blurring, sharpening, adding noise, and rotating the images. However, due to the similarities between characters, some techniques, as flipping the images, were avoided. To train the models, 80% of the dataset was used for training and the rest of the images were used for validation.

A. Ablation Experiments

We performed an ablation experiment on the networks by analyzing the use of different backbone architectures. Backbone architectures are used for feature extraction over an image before classification. We trained the YOLO network using the Full YOLO, Tiny YOLO, Inception v3 [23], SqueezeNet [24] and MobileNet [25] backbones. Mask R-CNN was trained using both the ResNet101 and the ResNet50 backbones. The backbone architectures differ mainly in relation to the number and organization of their convolutional layers. All models were trained for 200 epochs.

Another experiment was done to understand what features of the characters were learned by the networks. The obtained results are presented in the next sections.

B. Comparative Performance

The evaluation metric used to measure the accuracy of the trained models was mean Average Precision (mAP) as defined by the COCO dataset [26]. In this case, predictions are considered correct when their intersection over union (IOU) values regarding the ground truth bounding box are over 0.5. Scores for the models are shown in Table I.

TABLE I
MAP Scores for the different models trained.

Model	mAP Score
YOLO Full	0.9419
YOLO Tiny	0.9167
YOLO Inception V3	0.8872
YOLO SqueezeNet	0.8229
YOLO MobileNet	0.7905
Mask R-CNN (Resnet50)	0.9345
Mask R-CNN (Resnet101)	0.9248

Table II shows the average time in seconds that each of the models took to analyze a still image or a single frame of video, and generate predictions. Detection tests were run on the same platform where training was performed, using an NVIDIA Titan Xp GPU. Images used are always in the same resolution as those of the camera of the Donnie robot, i.e., 640x480 pixels.

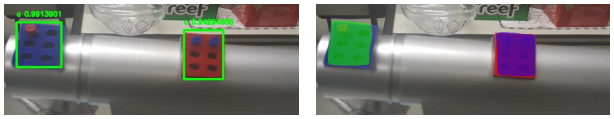
TABLE II
Detection times for the different models trained.

Model	Detection Speed (seconds)
YOLO Full	1.61
YOLO Tiny	1.21
YOLO Inception V3	4.15
YOLO SqueezeNet	0.95
YOLO MobileNet	1.89
Mask R-CNN (Resnet50)	3.76
Mask R-CNN (Resnet101)	4.40

C. Performance Analysis

What we can gather from the mAP scores is that all the trained models achieved good accuracy results. Of note, we see that the bigger number of layers in the resnet101 network actually decreases accuracy in the Mask R-CNN model, when compared to the smaller resnet50 network. The YOLO network benefits from the deeper architectures such as Full YOLO.

The greatest performance difference between the models is not their accuracy, but the detection time, as shown in Table II, with the YOLO models generally performing over three times faster than both Mask R-CNN configurations on the same hardware. This is expected, as one of the goals of the YOLO model is detection speed, with some implementations running detections at real-time speeds. Only the YOLO model using the Inception V3 backbone performed at similar speeds



(a) (b)

Fig. 6. Comparison between YOLO and Mask R-CNN prediction results. (a) YOLO bounding boxes. (b) Mask R-CNN detection masks.

of Mask R-CNN. There is not a clear correlation between detection speeds and mAP scores.

D. Qualitative Analysis

A benefit of the Mask R-CNN approach lies in the pixel mask it generates for predicted results. This mask is useful for increased precision in the detection of specific shapes of objects. A comparison of the bounding boxes predicted by the YOLO model and the masks of Mask R-CNN is shown in Figure 6. In our case, the shape of all objects of interest is the same, so this benefit is not particularly relevant. As such, given the good accuracy obtained and fast detection speeds, we decided to use the Full YOLO model for integration with the Donnie robot.

E. Feature Learning

As described in Section III, the characters in our dataset are formed from the same base components, a background piece, and a number of pins. This makes the classes to be detected and classified by the models very similar, with few distinctive visual characteristics.

It was of our interest to discover if those characteristics were enough to differentiate the characters, as the trained models were not originally proposed for this specific task. We decided to investigate which parts of the characters were more relevant to the learning. We did this by testing the trained models on images such as the ones shown in Figure 5.

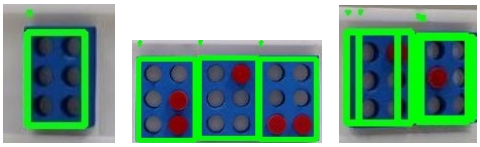
Figure 5(a) with one character in two background pieces was chosen to simulate a situation where two pieces are placed next to each other and a potentially valid character is accidentally formed between them. This test was intended to discover if the shape of the individual pieces was relevant compared to the contrast of the pins against the background color.

The empty background piece (Figure 5(b)) and the loose pins (Figure 5(c)) images were used to test the recognition of the models on the specific parts of the characters.

Another test was performed with images such as Figure 7(b), displaying invalid character configurations. This was to allow us to determine how characteristics such as color density and distribution, and the orientation of the characters were being used in the classification process.

None of the trained models detected the loose pins as a valid character. By contrast, some of the models detected the empty background piece as one or more characters, as shown in Figure 7(a). This suggests that the background pieces have a stronger influence on the detection than the pins.

Some models also detected false positives in images with invalid characters, like in Figure 7(b). However, there was no



(a) (b) (c)

Fig. 7. Detections performed on a number of test configurations. (a) Detection in empty background piece. (b) Detections in invalid configurations. (c) Detections in two pieces with one combined character.

correlation between the characters the models reported finding and the configurations in the images. The reported characters did not have a similar amount of pins, or pin positions that were reverse of the ones in the images. In addition, the confidence of those detections was lower than the one of the valid characters.

False positives also appeared in detections performed in the situation exemplified in Figure 5(a). This is shown in Figure 7(c). But the false positives were of wrong characters in the individual background pieces, not of the character between them.

The false positives on empty pieces and invalid characters suggest that the color of the pins and their positions don't provide enough differentiation for the classes.

F. Word Detection

Some of the trained models have problems in accurately detecting and classifying images with a large number of characters. This is more frequent in the models trained with smaller backbone architectures such as Tiny YOLO. An example is shown in Figure 8(a).

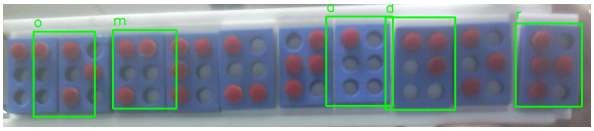
Another problem arises with the detection of more than one character in a single piece. An example is presented in Figure 8(b). However, we solve this problem by choosing the detected character with the highest confidence value.

As our goal was to detect complete words in the images, characters detected are sorted from left to right and grouped by how close their position is. A single wrong character prediction can cause an entire word to be wrong, despite the confidence of the detection of the other characters. To fix this problem, the words are passed through the spellchecking script pyspellchecker [4]. This also helps users who might have made a mistake placing pins.

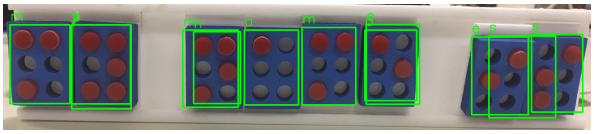
VI. Conclusions and Future Work

In this paper, we presented a system for detecting Braille-like characters in scaled-down real-world environments. Our contributions are mainly regarding the novel public dataset of labeled images of those Braille characters classified in the letters of the alphabet, and the controlled comparison among the deep learning state-of-the-art methods for detecting and classifying the characters. Another novelty is the proposal of the EVA pieces to represent Braille characters to allow the generation of the dataset and to allow the visually impaired

[4]https://pypi.org/project/pyspellchecker/



(a)



(b)

Fig. 8. Errors in word detection. (a) Partial Detection of Words. (b) Detection of overlapping characters in the same pieces.

to set up the real environment that must be recognized by the Donnie robot.

We are now working to embed the models in the Donnie robot, and further developing them with the aim of helping the visually impaired. Thus, for future work, we intend to perform tests with users that present visual impairment. We also intend to further explore possibilities of detecting regular embossed Braille characters in real-world locations, not only in the context of the Donnie system but possibly as a general tool to assist Braille literacy.

REFERENCES

[1] L. Luque, L. O. Brandão, E. Kira, and A. A. F. Brandão, "On the inclusion of learners with visual impairment in computing education programs in brazil: practices of educators and perceptions of visually impaired learners," *Journal of the Brazilian Computer Society*, vol. 24, no. 1, p. 4, Feb 2018. [Online]. Available: https://doi.org/10.1186/s13173-018-0068-0

[2] M. Konecki, N. Ivković, and M. Kaniški, "Making programming education more accessible for visually impaired," in *2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, May 2016, pp. 887–890.

[3] A. Hadwen-Bennett, S. Sentance, and C. Morrison, "Making programming accessible to learners with visual impairments: A literature review," *International Journal of Computer Science Education in Schools*, vol. 2, no. 2, 2018.

[4] S. Ludi and T. Reichlmayr, "The use of robotics to promote computing to pre-college students with visual impairments," *Trans. Comput. Educ.*, vol. 11, no. 3, pp. 20:1–20:20, Oct. 2011. [Online]. Available: http://doi.acm.org/10.1145/2037276.2037284

[5] R. Dorsey, C. H. Park, and A. Howard, "Developing the capabilities of blind and visually impaired youth to build and program robots," 2014.

[6] S. L. Ludi, L. Ellis, and S. Jordan, "An accessible robotics programming environment for visually impaired users," in *Proceedings of the 16th international ACM SIGACCESS conference on Computers & accessibility*. ACM, 2014, pp. 237–238.

[7] R. P. Barros, A. M. F. Burlamaqui, S. O. de Azevedo, S. T. de Lima Sa, L. M. G. Goncalves, and A. A. R. S. da Silva Burlamaqui, "Cardbot - assistive technology for visually impaired in educational robotics: Experiments and results," *IEEE Latin America Transactions*, vol. 15, no. 3, pp. 517–527, March 2017.

[8] G. H. M. Marques, D. C. Einloft, A. C. P. Bergamin, J. A. Marek, R. G. Maidana, M. B. Campos, I. H. Manssour, and A. M. Amory, "Donnie robot: Towards an accessible and educational robot for visually impaired people," in *2017 Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR)*, Nov 2017, pp. 1–6.

[9] S. Isayed and R. Tahboub, "A review of optical Braille recognition," *2015 2nd World Symposium on Web Applications and Networking, WSWAN 2015*, pp. 1–6, 2015.

[10] B. Nugroho, I. Ardiyanto, and H. A. Nugroho, "Review of optical braille recognition using camera for image acquisition," in *2018 2nd International Conference on Biomedical Engineering (IBIOMED)*. IEEE, 2018, pp. 106–110.

[11] G. Morgavi and M. Morando, "A neural network hybrid model for an optical braille recognitor," in *International Conference on Signal, Speech and Image Processing*, vol. 2002, 2002.

[12] L. Wong, W. Abdulla, and S. Hussmann, "A software algorithm prototype for optical recognition of embossed braille," in *Proceedings - International Conference on Pattern Recognition*, vol. 2, 2004, pp. 586–589.

[13] S. Zhang and K. Yoshino, "A braille recognition system by the mobile phone with embedded camera," in *Second International Conference on Innovative Computing, Informatio and Control (ICICIC 2007)*, Sep. 2007, pp. 223–223.

[14] J. Li and X. Yan, "Optical braille character recognition with support-vector machine classifier," in *Computer Application and System Modeling (ICCASM), 2010 International Conference on*, vol. 12. IEEE, 2010, pp. V12–219.

[15] M. Waleed, "Braille identification system using artificial neural networks," *Tikrit Journal of Pure Science*, vol. 22, no. 2, 2017.

[16] T. Li, X. Zeng, and S. Xu, "A deep learning method for Braille recognition," *Proceedings - 2014 6th International Conference on Computational Intelligence and Communication Networks, CICN 2014*, pp. 1092–1095, 2014.

[17] Y. Shimomura, H. Kawabe, H. Nambo, and S. Seto, "Construction of restoration system for old books written in braille," in *International conference on management science and engineering management*. Springer, 2017, pp. 469–477.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[19] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2017, pp. 2980–2988.

[20] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Jun 2014. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2014.81

[22] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[23] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[24] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[25] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[26] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

Viable Yeast Identification using Bag of Visual Words in Colored images

1st Junior Silva Souza
Instituto Federal de Mato Grosso do Sul (IFMS)
Campo Grande, Brazil
junior.souza@ifms.edu.br

2nd Vanessa Ap. de Moraes Weber
Universidade Catlica Dom Bosco (UCBD)
Universidade Estadual do MS (UEMS)
Campo Grande, Brazil
vamoraes@gmail.com

3rd Ariadne Barbosa Gonçalves
Universidade Federal de MS (UFMS)
Campo Grande, Brazil
ariadne.gon@gmail.com

4th Marco Alvarez
University of Rhode Island
Rhode Island, United States
malvarez@cs.uri.edu

5th Marney Pascoli Cereda
Agro: Laboratories, of Research Processes and Products
Campo Grande, Brazil
mpcereda@gmail.com

6th Wesley Nunes Gonçalves
Universidade Federal de MS (UFMS)
Campo Grande, Brazil
wnunesgoncalves@gmail.com

7th Valguima V. V. Aguiar Odakura
Universidade Federal da Grande Dourados (UFGD)
Dourados, Brazil
valguimaodakura@ufgd.edu.br

8th Hemerson Pistori
Universidade Catlica Dom Bosco (UCBD)
Universidade Federal de MS (UFMS)
Campo Grande, Brazil
pistori@ucdb.br

***Abstract*—In this research it is reported a system to automate the process of identification of viable yeasts whose population control is a crucial task in the ethanol production process. The identification and counting of yeasts made by human vision under a light microscope, is repetitive and susceptible to errors. We used computer vision techniques such as BoVW, Color Coherence Vectors (CCV), Color Moments (CM), Bag-of-Color (BoC) and Opponent Color (OpC) were applied for extracting characteristics that were classified by the Naive Bayes, KNN, SVM and J48 algorithms in 2614 images of yeasts separated into three classes: viable, non-viable and background. The results were analyzed using software R, which in the ANOVA test resulted in a p value equal to $2e^{-16}$ indicating a significant difference between the techniques. The OPC with SVM classifier showed the highest performance using the PCC Percent Correct Classification metric, about 95% compared to other techniques.**

Index Terms*—bag of visual words, color, supervised learning, *saccharomyces cerevisiae

I. Introduction

In the oil crisis in 1973, Brazil has adopted a new source of fuel, the ethanol. Fiscal incentives and funding have been proposed in order to increase the planting of sugar cane and installation of industries for processing. Sugar cane culture was inserted in Mato Grosso do Sul in 1980, after the adoption of the National Alcohol Program (ProAlcool). Due to increase of industries and sugar cane planted area, there is a search for new technologies to improve productivity and quality of ethanol productions [1].

Ethanol production is characterized by the fermentation of the wort (resulting from the dilution of sugarcane juice with water). In this process, the *Saccharomyces cerevisiae* yeasts are added to the wort for ethanol production by fermentation. This process occurs with the sugar consumption from the wort by yeasts, this process enable the ethanol and carbon gas production. To reduce costs in this production process, fermentation "MelleBoinot" is adopted due to its recycling of yeast feature. In this recycling, the yeasts are reused in successive fermentations. The recycling or reuse of yeasts reduce costs in the production process [2].

The quality of ethanol production is related to the yeast viability. Therefore, to ensure the production, it is necessary to check the microbiological control in laboratory, whereas viable yeasts are responsible for fermentation and non-viable yeasts have not action in the fermentation as they should [3].

In activities related to the microbiological control, wort samples are taken from the fermentation tanks and examined in the laboratory by a technical manager. This activity is visually, because requires to identify and counting yeast with the support of a light microscope (LM). Being a repetitive and visual activity, this task is susceptible to human errors, because its process can be tiring and subjective. To facilitate the yeasts identification, samples are mixed in water and methylene blue. The non-viable yeast turns colored in blue [4]. Two types of yeast: viable yeasts are marked by red squares while non-viable yeasts are marked by blue squares shown in Figure 1.

Activities such as the identification and counting of yeasts are repetitive tasks that can be performed automatically by computers programs. The yeast analyzes are done in microscopy images, that is way the proposal of this research is automate this process through computer vision and supervised

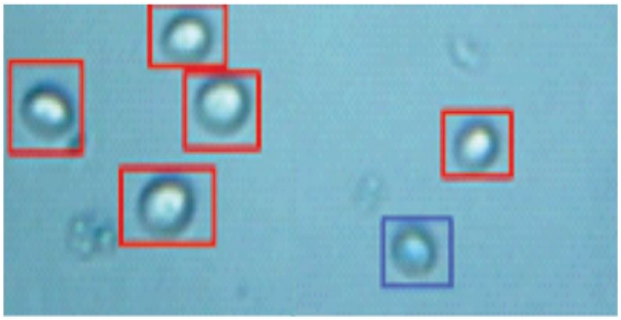


Fig. 1. Example of two types of yeasts: viable (red squares) and non-viable (blue squares). The non-viable yeasts are colored with blue when in contact with methilene blue corant.

learning.

In our research we use the BoVW for features extracting, because it is a widespread technique in many papers that used image classification. However, the color is a very important feature in many images, including yeast dye images [5].

This research aim is to evaluate the performance achieved by BoVW and its color variations on the yeast recognition. In this research, the following color variations were evaluated: color coherence vectors (CCV) [6], and color moments (CM) [7], bag of color (BoC) [8] and opponent color (OpC) [9]. CCV extracts color information by regions or clusters of a single color. The CM extracts color information from the average and variance applied to each image. The BoC is a color histogram, whose goal is to extract the frequency of certain colors. OpC is a variant applying BoVW in each color channel.

In order to evaluate the BoVW variants and classifiers (J48, NB, SVM and KNN), we performed ANOVA hypothesis test. The p-value indicated that the variants differ from each other. According to the experimental results, the OpC with SVM classifier achieved the highest performance. A second experiment was conducted with OpC and SVM, and the result showed that the dictionary with 256 visual words had the best result.

In section II is presented some related works and in section III is explained the materials and methods. The results are reported in section IV, followed by the discussion in section V, conclusion section and future works in the end of this paper.

II. Related Works

The viability of yeasts is commonly used to determine the efficiency of the production process. Usually, the physiological and metabolic changes of yeasts is observed using fluorescence microscopy or flow cytometry [10] [11]. However, these methods are time-consuming and prone to human-error, since they are not automatic or do not quantitatively analyze a large number of cells.

Recently, image-based methods and systems have been proposed to overcome these issues. Including [12] [13] who demonstrated the use of a fluorescence-based image cytometry system Cellometer Vision [10] for the analysis of vitality of *S. cerevisiae.* In order to observe the behavior of *S. cerevisiae*, [14] presented the CellStar, a tool for tracking yeast cells in long-term experiments. They compared CellStar with six other tools and demonstrated its high performance and accuracy. In addition, [15] [16] presented a platform for measuring viability of yeast cells by capturing an in-line hologram of the sample. This hologram sample is classified as live or dead by a Support Vector Machine for measuring viability as well as concentration. Furthermore [17] developed an automated cell counting for estimating the total number and the viability of yeast cells. To avoid reagent-based methods, [18] proposed a novel system to classify yeast viability based on wavelet features, feature selection and Support Vector Machine classifier.

Image-based methods are also proposed to classify yeast cells based on morphological characteristics. In this sense [19] proposing an image processing method designed to classify microscopic images of yeast cells in no budding, small bud, and large bud cells, which has been improved and included in a device [20]. The method is composed of four parts: image preprocessing to remove background noises, segmentation to separate yeast cells from background, extraction of morphological features (compactness, axis ratio and bud size) from each cell and classification using k-nearest neighbors. Texture features has also been used in the analysis of yeast cells. Since [21] proposed an image-based method for determining yeast floc dimension using co-occurrence matrices. They show that the energy of these matrices can be correlated with the mean particle diameter and therefore can be used to quantify changes in yeast floc size during fermentations.

III. Materials and Methods

In the classification stage, we use the following supervised learning techniques: decision tree (J48), naives bayes (NB), support vector machine (SVM) and k-nearest neighbor (KNN). These supervised learning techniques were used with different BoVW variants. Thus, we use computer vision to extract features and supervised learning algorithms to generate classifiers and make the yeast identification. In the feature extraction, we used an image database with 2614 yeast images. The images were manually segmented and defined in three classes: background without yeast, non-viable and viable yeast.

A. Yeast Database

Yeast samples were obtained from the fermentation process the which *Saccharomyces cerevisiae* yeast were added to the wort (water with sugarcane juice) at a concentration of 1% (w / v). The wort was set to 12 Brix and also used the samples when the value of Brix was at 6 and 3.

The yeast images used in this study have been built by INOVISO group (Development and Innovation in Computer Vision). The yeast images from Brix 03 were taken by LM at a 100x magnification. It was obtained 30 yeast images, they were manually segmented and separated in three classes: non-viable with 727 images, 292 images viable and background with 1595 images, totaling 2614 images for the yeast database.

IV. Results

The main hypothesis testing used in this study were the ANOVA and Friedman test with Tukey and Wilcoxon post-tests. The metric used to compare the techniques was the percentage of correct classification (PCC). It is the number of correctly identified images of all class, divided by the total number of images.

The techniques performance were analyzed using ANOVA and Friedman hypothesis test to compare the performance of techniques regarding the percentage of correct classification metric. The dictionary size used by BoVW and variants using the color information has been set to the value 512, because showed better results in relation to values comprehended 64 and 1024.

Combinations between features extractors and classifiers found at Weka software were carried out. The chosen classifiers were: KNN, J48, NB and SVM. The used features extractors were: BoC, BoW, CCV, CM and OpC. Thus, for example, the abbreviation KNNBoC indicates the combination between KNN classifier with BoC features extractor, giving origin to the KNNBoC technique. Similarly, the following abbreviations were defined: KNNBoC, KNNBoW, KNNCCV, KNNCM, KNNOpC, J48BoC, J48BoW, J48CCV, J48CM, J48OpC, NBBoC, NBBoW, NBCCV, NBCM, NBOpC, SVMBoC, SVMBoW, SVMCCV, SVMCM and SVMOpC.

A box-plot diagram obtained through R software shown in Figure 2. In this diagram is shown the performance of each technique. It was done by comparison of the medians, that are the darker strip located on each box. In the diagram we can see that the SVMOpC technique had the highest median performance. This technique results from the combinations between SVM classifier with the Opponent Color feature extractor. We can observe some unusual behavior, such as the combinations of the BoC feature extractor with the classifiers, almost have the best performances, except with the SVM classifier. The combination between KNN classifier with any features extractors (BoW, BoC, CCV and CM), presented almost all the worst results.

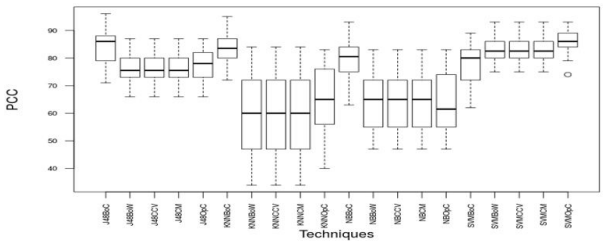


Fig. 2. Results of experiments. The performance is described in y-axis and all techniques are viewed in x-axis.

The variance analysis it was found a p-value $<2e^{-16}$. This result indicates that the null hypothesis can be ruled out, the medians indicate that there is a statistical difference between the techniques. The results of techniques with Turkey post-test showed with SVMOpc is similar with the follows techniques: KNNBoC, J48BoC, SVMBoW, SVMCCV and SVMCM. It happened because these techniques had a good performance in some sets of images than others, however SVMOpC technique maintained a higher. SVMOpC technique had the best performance. The SVMOpC is a technical variation of the BoW algorithm, allows the change of the dictionary size. The dictionary size Opponent Color was adjusted to: 128, 256, 512, 1024 and 2048 dictionary values.

The best result with SVMOpC technique, using the dictionary 256 shown in Figure 3. The p-value obtained by ANOVA was $1.58e^{-8}$, which indicates that the null hypothesis can be discarded, so these variations were very different from each other.

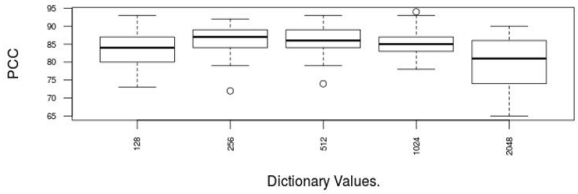


Fig. 3. Performance variations of SMOOpC technique. The y-axis represent performance and x-axis is size of dictionary.

The results with the dictionary size 256 have shown that the SVMOpC technique had the best performance in the yeast identification of colorless images, viable. The confusion matrix was obtained of image wherein the SVMOpC technical presented the best performance, since we used 121 images from sub-sample from database images, that show better result, for show the confusion matrix. In Table 1 is showed the confusion matrix, which there was 82 images identified as background, 16 images identified as viable yeast and 2 images identified as non-viable yeast.

TABLE I
Matrix of confusion showing the yeasts classification.

a	***b***	***c***	
82	1	12	a = background
3	2	1	b = non-viable
4	0	16	c = viable

V. Discussion

Two yeasts classified as viable, but it is a wrong identification, because image 6b is a non-viable yeast shown in Figure 4. This is one of the problems encountered, where two images were confused because of both yeasts have the same shape, but the color of the central region of each yeast is the factor to rank them. As we have no control over the regions detected by Color Opponent algorithm, then the interest points can be identified in any region of the image. This means that we do not have the spatial information, and this is one of the main problems found in the histograms.

As shown in Figure 5 both images were classified as viable yeast, but Figure 5b is a background. This is an example where

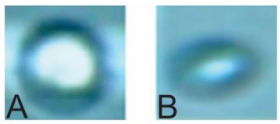


Fig. 4. Both images were classified as viable. A) Viable yeast. B) Non-viable yeast.

the classifier error, since the background color contrasts with the color of the center of viable yeast. This is an example where the shape is a factor that best distinguish viable yeast from image background. As we are working with the Color Opponent algorithm, we look for local changes in every image, leaving the feature concerning the form, which is an important feature in the image identification.

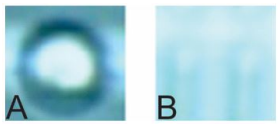


Fig. 5. Both images were classified as viable. A) Viable Yeast. B) Background.

The results showed that the color information added to BoVW algorithm improves the results in the yeasts identification. Although some problems were found in the identification, the SVMOpC technique with dictionary size of equal 256 showed good results when related to [1]. Our technique is invariant to image rotation and with a performance above 85% while a performance of 80% was reported by [1]. In relation to the [22], our technique is better to viable yeast identification.

VI. Conclusion

The counting activities and sorting of viable and non-viable yeast through the blue dye methylene are crucial for the ethanol production guarantee. One way to automate this process is to use the computer vision. In this research we analyzed the BoVW algorithm with some techniques that capture the color information for extracting features that have been used with a combination of classifiers.

The results showed that the opponent color feature extraction with SVM classifier achieved the best results. The metric used was the percentage of correct classification. ANOVA p-value were $2e^{-16}$ with both hypothesis tests. At confusion matrix the SVMOpC technique with 256 size dictionary identified the best viable yeast and the background, even with errors in the non-viable yeasts identification , the technical SVMOpC had the best performance than the others analyzed techniques. This work can be extended through the application in 3D images, thus increasing the amount of information of the image and guaranteed more real images.

Acknowledgment

This work was carried out with the financial support of the Coordination of Improvement of Higher Education Personnel - Brazil (CAPES) - Financing Code 001, National Council for Scientific and Technological Development (CNPq), Foundation for Support for the Development of Education, Science and Technology from the State of Mato Grosso do Sul (FUNDECT), Federal University of Mato Grosso do Sul (UFMS) and Catholic University Don Bosco (UCDB).

References

[1] Quinta, L.N.B., Queiroz, J.H.F.S., Souza, K.P., Pistori, H., Cereda, M.P. 2010. Classificao de Leveduras para o Controle Microbiano em Processos de Produo de Etanol. VI Workshop de Viso Computacional, p. 90-94.

[2] Boinot, M. 1939. Process of alcoholic fermentation with re-use os the yeass. The internatioal Sugar Journal.

[3] Stratford, M., 1996. Yeast flocculation: restructuring the theories in line with recent research. Belgian J. Brew. Biotechnol.

[4] Ceccato-Antonini, S.R., 2011. Microbiologia da fermentao alcolica: a importncia do monitoramento microbiolgico em destilarias. So Carlos: Universidade Federal de So Carlos, p. 105.

[5] Van Weijer, J., Khan, F. S. 2013. Fusing color and shape for bag-of-words based object recognition. In Computational Color Imaging, Springer, p. 2534.

[6] Pass, G., Zabih, R., Miller, J., 1997. Comparing images using color coherence vectors, in: Proceedings of the Fourth ACM International Conference on Multimedia. p. 6573.

[7] Bahri, A., Zouaki, H., 2013. A SURF-color moments for images retrieval based on bag-of features. Eur. J. Comput. Sci. Inf. Technol. 1, 1122.

[8] Wengert, C., Douze, M., Jgou, H., 2011. Bag-of-colors for improved image search, in: Proceedings of the 19th ACM International Conference on Multimedia. p. 14371440.

[9] van de Sande, K.E.A., Gevers, T., Snoek, C.G.M., 2008. Color descriptors for object category recognition, in: Conference on Colour in Graphics, Imaging, and Vision. p. 378381.

[10] Chan, L.L., Lyettefi, E.J., Pirani, A., Smith, T., Qiu, J., Lin, B., 2011. Direct concentration and viability measurement of yeast in corn mash using a novel imaging cytometry method. J. Ind. Microbiol. Biotechnol. 38, p. 11091115. doi: 10.1007/s10295-010-0890-7.

[11] Zhang, T., Fang, H.H.P., 2004. Quantification of Saccharomyces cerevisiae viability using BacLight. Biotechnol. Lett. 26, 989992.

[12] Chan, L.L., Kury, A., Wilkinson, A., Berkes, C., Pirani, A., 2012. Novel image cytometric method for detection of physiological and metabolic changes in Saccharomyces cerevisiae. J. Ind. Microbiol. Biotechnol. 39, p. 16151623. doi: 10.1007/s10295-012-1177-y.

[13] Saldi, S., Driscoll, D., Kuksin, D., Chan, L.L.-Y., 2014. Image-based cytometric analysis of fluorescent viability and vitality staining methods for ale and lager fermentation yeast. J. Am. Soc. Brew. Chem. 72, 253260. doi: https://doi.org/10.1094/ASBCJ-2014-1015-01.

[14] Versari, C., Stoma, S., Batmanov, K., Llamosi, A., Mroz, F., Kaczmarek, A., Deyell, M., Lhoussaine, C., Hersen, P., Batt, G., 2017. Long-term tracking of budding yeast cells in brightfield microscopy: CellStar and the Evaluation Platform. J. R. Soc. Interface 14. doi: 10.1098/rsif.2016.0705.

[15] Feizi, A., Zhang, Y., Greenbaum, A., Guziak, A., Luong, M., Chan, R., Berg, B., Ozkan, H., Luo, W., Wu, M., others, 2017a. Lensfree on-chip microscopy achieves accurate measurement of yeast cell viability and concentration using machine learning, in: CLEO: Applications and Technology. p. ATh4B–4.

[16] Feizi, A., Zhang, Y., Greenbaum, A., Guziak, A., Luong, M., Chan, R.Y.L., Berg, B., Ozkan, H., Luo, W., Wu, M., others, 2017b. Yeast viability and concentration analysis using lens-free computational microscopy and machine learning, in: Optics and Biophotonics in Low-Resource Settings III. p. 1005508.

[17] Hong, D., Lee, G., Jung, N.C., Jeon, M., 2013. Fast automated yeast cell counting algorithm using bright-field and fluorescence microscopic images. Biol. Proced. Online 15, 13. doi: 10.1186/1480-9222-15-13.

[18] Wei, N., Flaschel, E., Friehs, K., Nattkemper, T.W., 2008. A machine vision system for automated non-invasive assessment of cell viability via dark field microscopy, wavelet feature selection and classification. BMC Bioinformatics 9, 449. doi: 10.1186/1471-2105-9-449.

[19] Yu, B.Y., Elbuken, C., Ren, C.L., Huissoon, J.P., 2011. Image processing and classification algorithm for yeast cell morphology in a microfluidic chip. J. Biomed. Opt. 16, 66008. doi: 10.1117/1.3589100.

[20] Yu, B.Y., Elbuken, C., Shen, C., Huissoon, J.P., Ren, C.L., 2018. An integrated microfluidic device for the sorting of yeast cells using image processing. Sci. Rep. 8, 3550. doi:10.1038/s41598-018-21833-9.

[21] Mas, S., Ghommidh, C., 2001. On-line size measurement of yeast aggregates using image analysis. Biotechnol. Bioeng. 76, p. 9198. doi: 10.1002/bit.1148.

[22] Mongelo, A.I., Da Silva, D.S., Quinta, L.I.A.N.B., Pistoti, H., Cereda, M.P., 2011. Validao de mtodo baseado em viso computacional para automao de contagem de viabilidade de leveduras em indstrias alcooleiras, in: VIII Congresso Brasileiro de Agroinformtica SBIAGRO.

Analysis of futsal matches using a single-camera computer vision system

Heloiza M Paulichen*, Kallil M C Zielinski†, Dalcimar Casanova‡ and Pablo G Cavalcanti§
Universidade Tecnológica Federal do Paraná
Pato Branco, Paraná, Brasil
*Email: *paulichen@alunos.utfpr.edu.br, †kallil@alunos.utfpr.edu.br, ‡dalcimar@utfpr.edu.br, §pgcavalcanti@utfpr.edu.br*

***Resumo*—The use of computer systems in sports has increased significantly in the last decade. Consequently, systems have been developed to help each athlete or team quantify their performance, such as distances traveled, speeds attained, and positions where each athlete was on the court or field. In this work, a method based on computer vision is proposed to analyse futsal matches. Videos were acquired using a single camera with a wide-angle lens, which facilitates the installation and calibration process in different matches and arenas. The approach is illustrated through video recordings of Pato Futsal team, from which the athletes were detected, their positions projected from pixels to real world coordinates and their trajectories estimated. The generated data visualization aims to help coaches in their physical and tactical analysis.**

1. Introdução

Assim como o futebol, o futsal vem conquistando seu espaço no cenário mundial e no Brasil [1]. Esse crescimento recente tem levado a um aumento da demanda por informações estatísticas relacionadas as partidas e aos seus jogadores de forma individual e coletiva. Algumas dessas estatísticas são tradicionalmente anotadas manualmente pelos mesários, como tempo de bola rolando, tempo de cada jogador em quadra, etc. Por outro lado existe uma série de dados que são muito complicados para se obter de forma manual, tal como a distância percorrida por cada jogador, a região da quadra de mais atuação e até as iterações entre os jogadores (e.g. roubadas de bola, marcação, etc). O estudo dessas interações entre os atletas é de especial importância em esportes coletivos, como futebol e futsal [2].

Esse tipo de informação estatística mais detalhada é importante subsídio para elevar o desempenho geral da equipe, pois possibilita à equipe técnica evidenciar o padrão tático utilizado por seus adversários assim como planejar o treinamento físico e tático de seus atletas [3]. Permite também que o público geral compreenda melhor esse esporte em suas qualidades e complexidades, melhorando a qualidade das transmissões e análises pós-jogo [4].

Todavia, a aquisição desse tipo de dado demanda um sistema automatizado, que pode ser concebido por uma série de sensores ou através da aquisição de imagens. A abordagem por imagens é, sob um determinado ponto de vista, menos complexa, pois dispensa que cada jogador tenha de carregar um dispositivo e também dispensa a instalação de uma série de antenas para captura dos sinais. A tecnologia de medição baseada na utilização de câmeras e vídeos é conhecida como videogrametria [5] e pode ser combinada com outras técnicas de visão computacional.

Entretanto, a abordagem por visão computacional no futsal ainda é considerada escassa em estudos científicos [4]. Nesta linha, avanços recentes dos métodos têm motivado um aumento no interesse de tal abordagem.

Neste trabalho, um método foi desenvolvido para segmentação e, posterior análise estatística, dos jogadores de linha de uma equipe de Futsal. Tanto para o treinamento quanto para avaliação do método, foram utilizados vídeos de partidas da equipe Pato Futsal, gravados no ginásio Dolivar Lavarda em Pato Branco - PR. Em uma análise inicial, foi medida a distância percorrida de cada jogador da equipe Pato Futsal e o respectivo mapa de calor gerado pela posição do jogador na quadra ao longo do tempo da partida.

Adicionalmente, optou-se por utilizar uma única câmera de lente grande angular para captura dos vídeos. Essa abordagem facilita a aquisição da imagem, uma vez que quase toda a quadra está dentro do campo de visão do dispositivo único de captura, algo que não seria possível com lentes convencionais, dado o espaço disponível dentro da maioria dos ginásios de futsal.

2. Trabalhos relacionados

Outros trabalhos já abordaram a análise de partidas de futsal por meio de processamento de imagens e visão computacional, e nesta seção eles são apresentados.

Em [6], o autor utilizou três câmeras para obter as gravações de uma partida de futsal. Após a obtenção dos vídeos, foi necessário realizar uma sincronização para que fossem processados quadros em um mesmo instante de tempo. A calibração foi realizada manualmente, dependendo de um usuário para reconhecer pontos comuns aos dois planos e marcá-los como equivalentes. A partir da calibração, matrizes homográficas foram geradas para calcular a correspondência entre os planos. Finalmente, para a detecção dos jogadores, um Modelo de Mistura de Gaussianas (MMG) foi utilizado para segmentação, que busca modelar o fundo a partir de um conjunto de imagens passadas.

Outra abordagem por [7] também utilizou um MMG para a remoção de fundo, mas com apenas uma câmera

estática posicionada no topo da quadra. O objetivo era analisar a movimentação dos jogadores de futsal usando um filtro de particulas preditivo, que usa informações do estado atual de um objeto para inferir seu estado no próximo instante de tempo. Os resultados das movimentações foram mostrados em gráficos onde o plano de fundo era a própria quadra de futsal e a trajetória dos jogadores era destacada por linhas.

Uma análise sobre a correção de distorção radial em câmeras de lente grande angular no rastreamento de jogadores de futsal é feita por [8]. Quatro câmeras foram usadas para gravar uma partida das quartas de final da Liga de Futsal de São Paulo em 2013, três câmeras possuíam lentes convencionais e uma câmera era de lente grande angular. Para corrigir a distorção radial, foi aplicada a transformação de Hough para identificar linhas distorcidas nas imagens. A calibração de câmera foi feita baseada em 23 pontos de controle em uma superfície de corte, com as distâncias reais previamente medidas. A trajetória dos jogadores foi obtida pelo software DVideo. Uma comparação entre os resultados para as câmeras de lentes convencionais e a câmera de lente grande angular é mostrada e os resultados indicam que o uso de uma câmera de lente grande angular com correção de distorção radial é similar às câmeras de lente convencionais, mas com a vantagem de que o campo de visão é maior, tornando possível a captura da quadra inteira.

3. Referencial Teórico

3.1. Regras do Futsal

O futsal é uma modalidade esportiva coletiva, em que as disputas ocorrem entre equipes com cinco integrantes de cada lado. A quadra deve ser retangular e as dimensões oficiais para partidas nacionais são de, no mínimo, 25m x 16m e, no máximo 42m x 25m. Já para partidas internacionais, as dimensões são de, no mínimo, 38m x 20m e, no máximo, 42m x 25m [9].

3.2. Detecção de jogadores

Para a detecção dos jogadores na quadra, o método baseado em rede neural YOLOv3 (*You Only Look Once*) foi escolhido. Esse modelo de rede apresenta um sistema de detecção de objetos em tempo real que vem apresentando bons resultados na literatura. Em [10], a rede YOLOv3 apresentou uma precisão média de detecções (mAP) de 57,9% no conjunto de dados MS COCO (*Microsoft Common Objects in COntext*, processando a 30 quadros por segundo.

Para a extração de características, essa rede usa camadas convolucionais consecutivas de 3x3 e 1x1, possuindo no total 53 camadas convolucionais. Essa estrutura é denominada de Darknet-53 e é apresentada na Tabela 1.

Na arquitetura Darknet-53, cada camada convolucional é seguida de uma camada de normalização em lote e pela ativação do Leaky ReLU. Nenhuma forma de *pooling* é usada e uma camada convolucional com *stride* 2 é usada

	Tipo	Filtros	Tamanho	Saída
	Convolucional	32	3 x 3	256 x 256
	Convolucional	64	3 x 3 / 2	128 x 128
1x	Convolucional	32	1 x 1	
	Convolucional	64	3 x 3	
	Residual			128 x 128
	Convolucional	128	3 x 3 / 2	32 x 32
2x	Convolucional	64	1 x 1	
	Convolucional	128	3 x 3	
	Residual			64 x 64
	Convolucional	256	3 x 3 / 2	32 x 32
8x	Convolucional	128	1 x 1	
	Convolucional	256	3 x 3	
	Residual			32 x 32
	Convolucional	512	3 x 3 / 2	16 x 16
8x	Convolucional	256	1 x 1	
	Convolucional	512	3 x 3	
	Residual			16 x 16
	Convolucional	1024	3 x 3 / 2	8 x 8
4x	Convolucional	512	1 x 1	
	Convolucional	1024	3 x 3	
	Residual			8 x 8
	Média de Pooling		Global	
	Densa		1000	
	Softmax			

Tabela 1: Estrutura do modelo Darknet-53.

Adaptado de: [10].

para reduzir a amostragem dos mapas de características. Isso ajuda a impedir a perda de características de baixo nível geralmente atribuídos ao *pooling*. Para uma imagem de entrada com dimensão 416 x 416, é feita uma grade de 13 x 13 células. Cada célula da grade deve ser responsável por encontrar o local exato e a categoria à qual o objeto pertence. O Darknet-53 usa três escalas para detectar objetos grandes, objetos médios e objetos pequenos. Os três tamanhos relativos dos mapas de características resultantes são 13 x 13, 26 x 26 e 52 x 52 [11].

3.3. Remoção de Fundo

Para que a detecção de jogadores seja mais eficiente, um pré-processamento de remoção de fundo foi realizado nas imagens do vídeo. Este pré-processamento tem por objetivo retirar uma grande quantidade de objetos estáticos e semiestáticos que não são de interesse da análise proposta (e.g. as traves, as linhas da quadra, publicidades, etc).

De acordo com [12], uma das abordagens mais simples para a detecção de mudanças entre dois quadros de imagem $f(x, y, t_i)$ e $f(x, y, t_j)$, tomados nos momentos t_i e t_j, respectivamente, é comparar as duas imagens pixel por pixel. Uma forma de fazer isso é criar uma imagem que represente a diferença entre a imagem referência e outra imagem subsequente na mesma cena, em que a imagem referência contém apenas componentes estáticos. Ao compará-las, a imagem diferença elimina os elementos fixos em ambas as imagens e mostra os componentes em movimento. Mais detalhes sobre a imagem de referência utilizada neste estudo são apresentados na Seção 4.

3.4. Correção de lentes e transformação projetiva

Como neste trabalho optou-se por utilizar aquisição dos vídeos com um dispositivo de lente grande angular (popularmente conhecidas como "olho de peixe"), é necessária uma correção da distorção gerada por essas lentes. Esta correção é responsável por determinar a posição de cada pixel em um espaço não distorcido a partir da imagem original, obtendo assim uma imagem retilínea [13].

Após a correção da distorção, é necessário aplicar transformações projetivas para realizar o mapeamento dos pontos da dimensão 3D da quadra (x, y, z) em pixels da imagem 2D (x', y'). Possibilitando realizar medições em cada quadro dos vídeos das partidas.

Segundo [14], dados dois espaços projetivos de dimensões m e n, as transformações $T : \mathbb{RP}^m \longrightarrow \mathbb{RP}^n$ são dadas por transformações $T : \mathbb{R}^{m+1} \longrightarrow \mathbb{R}^{n+1}$, chamadas transformações projetivas.

Para este trabalho, foi utilizada a homografia, que é uma transformação projetiva planar, ou seja, mapeia pontos de um plano π para outro plano π', como mostra a Figura 1. Em aspectos práticos, um plano da homografia é representado pelos quadros dos vídeos e o plano resultante definido foi a vista superior da quadra.

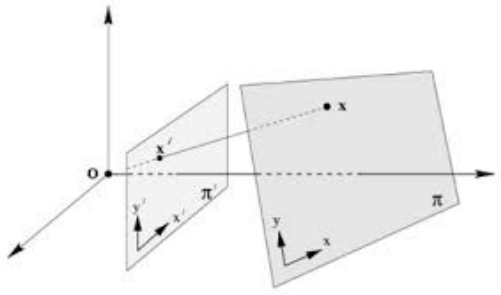


Figura 1: Mapeamento entre planos utilizando homografia. Fonte: [15]

3.5. Rastreamento e otimização da trajetória

Conforme descrito anteriormente, os jogadores são detectados em cada quadro do vídeo. Para que eles sejam associados entre um quadro e outro, ou seja, para rastrear os jogadores, uma posição no centro inferior de cada *bounding box* foi estimada, representando os pés dos jogadores. Foi estabelecido um ponto inicial (x_0, y_0) onde um jogador foi detectado em um determinado quadro e, então, houve uma comparação com o quadro seguinte. O jogador detectado em uma posição (x_1, y_1), que estava mais próximo à posição inicial (x_0, y_0), foi considerado o mesmo atleta do quadro anterior.

Entretanto, este processo de rastreamento pode acarretar em erros, seja por oclusão de jogadores em determinados quadros ou mesmo por imprecisões do algoritmo de detecção. Para minimizar esse efeito, foi utilizado o filtro de Kalman.

O filtro de Kalman possui inúmeras aplicações, sendo comum para orientação, navegação e controle de veículos, tais como aeronaves e navios [16]. Além disso, o filtro de Kalman tem seu conceito amplamente difundido em análises de série temporais usadas no campo de processamento de sinais. Os filtros Kalman também são um dos principais tópicos no controle de movimento robótico, e às vezes são incluídos na otimização de trajetória [17].

O filtro de Kalman assume que o estado real em um tempo k é proveniente de um estado em $(k-1)$ de acordo com:

$$x_k = F_k x_{k-1} + B_k u_k + w_k \tag{1}$$

em que F_k é o modelo de transição de estado aplicado ao estado anterior em x_{k-1}, B_k é o modelo de controle de entrada aplicado ao vetor u_k, e w_k é o ruído do processo, o qual se supõe ser extraído de uma distribuição normal multivariada de média zero, $\mathcal{N}$, com covariância Q_k: $w_k \sim \mathcal{N}(0, Q_k)$.

Em k uma medida z_k do estado real x_k é feita de acordo com:

$$z_k = H_k x_k + v_k \tag{2}$$

em que H_k é o modelo de observação que mapeia o espaço de estado real no espaço observado e v_k é o ruído de observação que é assumido como sendo um ruído branco gaussiano de média zero com covariância R_k: $v_k \sim \mathcal{N}(0, R_k)$.

O estado inicial e os vetores de ruído a cada passo $\{x_0, w_1, ..., w_k, v_1 ... v_k\}$ são assumidos como sendo mutuamente independentes.

4. Materiais e método proposto

A câmera utilizada para este projeto é a GoPro HERO 4 Black Edition. Os vídeos foram gravados com 30 quadros por segundo (fps) e resolução de 1080p SuperView - esta opção utiliza o formato grande angular de lentes, permitindo um campo de visão maior e, por consequência, quase que a totalidade da quadra é visível nas imagens (ver Figura 2). Apesar das redes de proteção da quadra estarem visíveis em partes da imagem, elas não interferiram nos resultados obtidos.

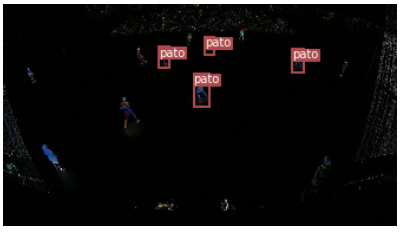
Os testes para o método proposto foram realizados a partir de dois vídeos, sendo cada um de 40 segundos e totalizando 1200 quadros. Aleatoriamente, 50 quadros foram separados para o treinamento da rede YOLOv3 e os demais utilizados para testes.

Inicialmente foi gerada uma imagem referência utilizando a mediana dos canais RGB de 500 quadros de um video controle. A Figura 3a apresenta esta imagem, a qual foi utilizada como referência para remoção de fundo. As imagens utilizadas no processo de detecção dos jogadores são, portanto, resultados da subtração desta imagem referência a partir das imagens originais.

Figura 2: Posicionamento da câmera durante as partidas.

Para o treinamento da rede YOLOv3, foram marcados 4 diferentes *bounding boxes* em cada quadro de teste, identificando apenas os jogadores de linha, pois a movimentação do goleiro foi ignorada em nossos experimentos. Portanto, 200 *bounding boxes* referentes aos jogadores da equipe em estudo são utilizados. A partir dos pesos do modelo Darknet-53 pré-treinados para o conjunto de dados MS COCO, após 100 épocas, a rede se mostrou capaz de detectar objetos de uma única classe, aqui denominada *pato*. A Figura 3b apresenta um resultado da aplicação do YOLOv3 após este treinamento. É possível notar os 4 *bounding boxes* correspondentes às posições dos 4 jogadores da equipe em estudo durante o quadro em questão.

(a) Fundo



(b) Detecção dos jogadores sem o fundo

Figura 3: Detecção dos jogadores com remoção de fundo.

Identificado os objetos de interesse, a correção da distorção da lente foi realizada utilizando o método proposto em [8], o qual é capaz de corrigir as distorções a partir da identificação automática das linhas da quadra a partir da transformada de Hough [18].

Em sequência, para a transformação projetiva, foi definido um plano com dimensões 400 x 200 pixels, o qual representa a vista superior da quadra. Esta dimensão foi escolhida para estabilizar uma escala de 10:1 com a quadra, cujas dimensões são 40m x 20m, tornando mais fácil a conversão de medidas e cálculos como distância percorrida. Aplicando a técnica de homografia na imagem, foi possível obter a representação da vista superior da quadra. Observa-se que os cantos inferiores desta representação é indefinida, em razão de estarem fora do campo de visão da câmera.

Figura 4: Vista superior da quadra obtida por homografia.

5. Resultados

Para ambos os vídeos, foi avaliada a qualidade da detecção dos jogadores dado pelos *bounding boxes* resultantes da inferência e mensurados a distância percorrida e a posição dos jogadores na quadra. Conforme anteriormente descrito, cada jogador é rastreado quadro-a-quadro e suas trajetórias corrigidas por filtros de Kalman.

Através de mapas de calor (coluna 1 das imagens 6 e 8), apresenta-se visualizações relativas a todas posições de quadra que cada jogador esteve ao longo do vídeo. Onde, cores mais quentes refletem partes da quadra em que o jogador esteve mais tempo presente. Já com gráficos de linha (coluna 2 das imagens 6 e 8), é possível analisar a trajetória completa da movimentação do jogador. Por sua vez, a coluna 3 apresenta uma relação entre distância percorrida ao longo do tempo decorrido.

5.1. Análise vídeo I

No primeiro vídeo, ao aplicar a rede YOLOv3 para a detecção dos jogadores, foram detectados quatro *bounding boxes* para todos os quadros do vídeo, como é possível notar na Figura 5. Pode-se notar que os quatro jogadores do Pato Futsal foram corretamente identificados e estão representados na figura com a cor roxa.

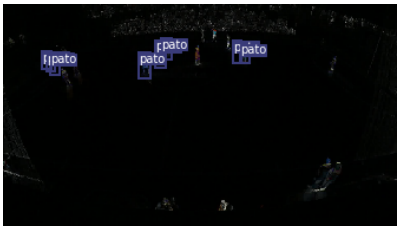
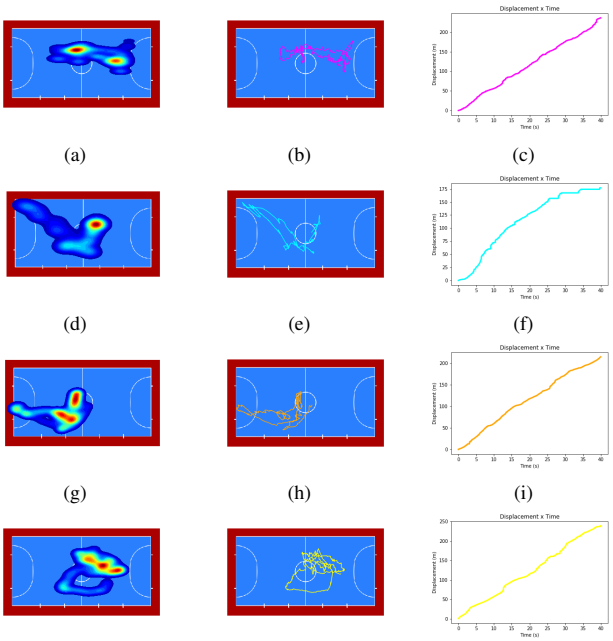


Figura 5: Quadro do Vídeo I sendo analisado pelo YOLOv3.

Após a detecção dos jogadores, os processos de rastreamento, correção de distorção e transformação projetiva, como mencionados na Seção 3 deste trabalho, foram aplicados para, então, coletar os dados sobre as movimentações individuais dos jogadores e apresentá-las como visualizações. O mapa de calor, gráfico de movimentação e distância percorrida pelo tempo são mostrados na Figura 6.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

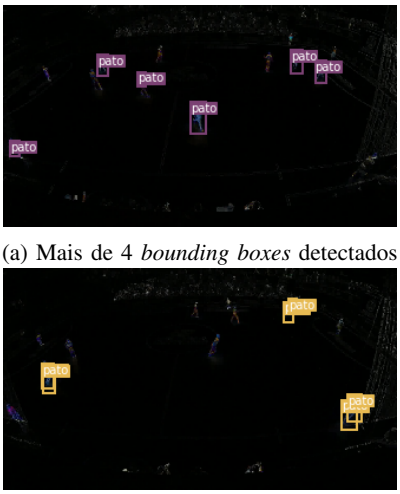
Figura 6: Visualização de dados para o Vídeo I.

Pode-se observar nessa figura que os jogadores representados pela cor amarela e magenta tiveram suas movimentações mais concentradas na metade direita da quadra (neste momento da partida, respectiva a defesa), enquanto os jogadores representados pelas cores laranja e ciano chegaram algumas vezes próximos ao gol do adversário e suas movimentações se deram na parte esquerda da quadra (região de ataque). Quanto aos gráficos de movimentação, é possível notar que o jogador representado pela cor ciano teve um pico de velocidade dos 5 aos 25 segundos de vídeo, enquanto os demais se movimentaram de forma mais uniforme.

5.2. Análise vídeo II

Aplicando o YOLOv3 para o vídeo II, em alguns casos foram detectados mais de quatro *bounding boxes*, como mostra a Figura 7a. O processo de rastreamento é o responsável por eliminar os sobressalentes, ou seja, aqueles mais distantes das posições dos quatro jogadores no quadro anterior. Neste vídeo, em 94.7% dos quadros foram detectados 4 ou mais *bounding boxes*. Porém, também houve casos em que a rede captou menos de quatro *bounding boxes*, caso representado na Figura 7b. Nestes casos, optamos por repetir a coordenada anterior do jogador que não foi associado pelo processo de rastreamento. Com essas correções, garantiu-se que as coordenadas de cada jogador seriam sempre estimadas e seus movimentos mapeados.

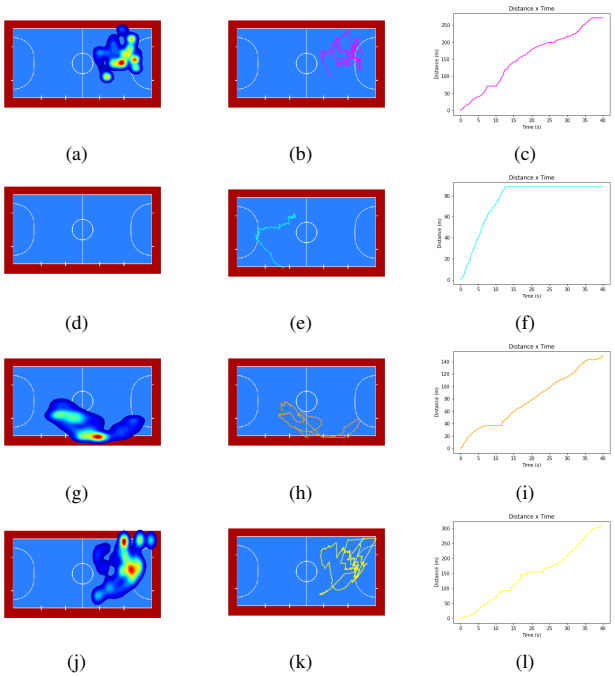
As visualizações geradas para o Vídeo II são apresentadas na Figura 8.



(a) Mais de 4 *bounding boxes* detectados

(b) Menos de 4 *bounding boxes* detectados

Figura 7: Quadros do Vídeo II sendo analisado pelo YOLOv3.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Figura 8: Visualização de dados para o Vídeo II.

Para esta análise, a detecção do jogador representado pela cor ciano passou a ser trocada pela posição da árbitra, fazendo com que sua referência fosse perdida. Uma possível explicação para isso é que a cor do uniforme da árbitra, após a aplicação da remoção de fundo, ficou similar com a cor do uniforme dos jogadores da equipe Pato Futsal e isso fez com que a detecção se equivocasse. Também é possível identificar na Figura 8 que está acontecendo uma jogada de defesa para a equipe Pato Futsal, pois a movimentação

dos jogadores se concentrou no campo de defesa. Além disso, em alguns momentos os jogadores ficaram parados em suas posições, como é possível observar nos gráficos de movimentação em função do tempo.

6. Considerações finais

Neste artigo, foi proposto um método para análise de partidas de futsal utilizando métodos de visão computacional. Para tal, foram utilizadas etapas de aquisição de imagens, subtração de fundo, detecção de objetos, correção de distorção, transformação projetiva e suavização de erros no processo.

Os resultados obtidos foram satisfatórios, com baixo erro na etapa de detecção de objetos e que foram possíveis de correção. Destaca-se a utilização de câmera única como dispositivo de aquisição, sem necessidade de integração, sincronização ou registro de imagens de diferentes fontes.

A transformação projetiva por homografia é de especial destaque, pois permite que se projete o campo de visão da câmera para uma vista superior da quadra, tornando mais fácil a análise técnica de jogadas e posicionamento do time.

A visualização obtida por mapas de calor permitiu analisar o local da quadra com maior concentração de movimento de cada um dos jogadores individualmente. Essa análise pode gerar subsídios para análise tática do jogo de forma precisa, possivelmente melhorando o desempenho da equipe. Também gera subsídios para avaliar o desempenho físico individual de cada jogador, medido aqui pela distância percorrida em relação ao tempo.

Trabalhos futuros como a identificação do goleiro e dos jogadores adversários, permitindo uma análise mais profunda perante determinados tipos de jogadas dos times adversários, poderão ser realizados a partir deste apresentado.

Com avanço da técnica, é possível criar uma ferramenta para analisar o comportamento e a movimentação dos jogadores durante as partidas, possibilitando consequentemente corrigir os respectivos erros em tempo real.

Referências

[1] B. H. Soares and H. Tourinho Filho, "Análise da distância e intensidade dos deslocamentos, numa partida de futsal, nas diferentes posições de jogo," *Revista Brasileira de Educação Física e Esporte*, vol. 20, no. 2, pp. 93–101, 2006.

[2] Z. Niu, X. Gao, and Q. Tian, "Tactic analysis based on real-world ball trajectory in soccer video," *Pattern Recognition*, vol. 45, no. 5, p. 1937–1947, 2012.

[3] G. Zhu, C. Xu, Y. Rui, S. Jiang, W. Gao, and H. Yao, "Trajectory based event tactics analysis in broadcast sports video," *Proceedings of the 15th ACM international conference on Multimedia*, vol. 5, pp. 58–67, 2007.

[4] R. Moore, S. Bullough, S. Goldsmith, and L. Edmonson, "A systematic review of futsal literature," *American Journal of Sports Science and Medicine*, vol. 2, no. 3, pp. 108–116, 2014.

[5] P. J. Figueroa, N. J. Leite, and R. M. Barros, "Tracking soccer players aiming their kinematical motion analysis," *Computer Vision and Image Understanding*, vol. 101, no. 2, pp. 122–135, 2006.

[6] M. B. de Oliveira, "Detecção Automática de Jogadores de Futsal Baseada em Visão Computacional," 2017, monografia - Curso de Engenharia de Computação, Centro Federal de Educação Tecnológica de Minas Gerais. Belo Horizonte.

[7] P. H. C. d. Pádua, F. L. C. Pádua, and M. T. D. Sousa, "Particle filter-based predictive tracking of futsal players from a single stationary camera," *Brazilian Symposium of Computer Graphic and Image Processing*, vol. 2015-Octob, no. August, p. 134–141, 2015.

[8] L. H. P. Vieira, E. A. Pagnoca, F. Milioni, R. A. Barbieri, R. P. Menezes, L. Alvarez, L. G. Déniz, D. Santana-Cedrés, and P. R. P. Santiago, "Tracking futsal players with a wide-angle lens camera: accuracy analysis of the radial distortion correction based on an improved hough transform algorithm," *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, vol. 5, no. 3, p. 221–231, 2017.

[9] FIFA, *Futsal laws of the game 2014/2015*. FIFA-Strasse 20, P.O. Box, 8044 Zurich, Switzerland: FIFA, 2014.

[10] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv*, 2018.

[11] J. Redmon, "Darknet: Open source neural networks in c," 2016. [Online]. Available: http://pjreddie.com/darknet/

[12] R. C. Gonzalez and R. E. Woods, *Processamento Digital de Imagens*. Av. Ermano Marchetti, 1435 CEP: 05038-001 – São Paulo – SP: Pearson, 2009.

[13] A. C. Valente, "Análise Comparativa de Compressão de Imagens Fisheye, Retilíneas e Panorâmicas," 2010, monografia - Curso de Engenharia Elétrica, Universidade de Brasília, Faculdade de Tecnologia. Brasília.

[14] C. C. dos Santos Cavalcanti Marques, "Um Sistema de Calibração de Câmeras," 2007, dissertação - Mestrado da área de concentração de Computação Gráfica, Universidade Federal de Alagoas, Programa de Pós Graduação em Matemática. Maceió.

[15] M. C. dos Santos, "Revisão de Conceitos em Projeção, Homografia, Calibração de Câmera, Geometria Epipolar, Mapas de Profundidade e Varredura de Planos," Unicamp, Campinas, Tech. Rep., 2012.

[16] P. Zarchan and H. Musoff, *Fundamentals of Kalman filtering: a practical approach*. American Institute of Aeronautics and Astronautics, Inc., 2013.

[17] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[18] A. Shehata, S. Mohammad, M. Abdallah, and M. Ragab, "A survey on hough transform, theory, techniques and applications," 02 2015.

Analysis of color feature extraction techniques for Fish Species Identification

1st Uéliton de Paula Freitas
Faculdade de Computação - FACOM
Universidade Federal de Mato Grosso do Sul
Campo Grande, Brazil
freitas.ueliton@gmail.com

2nd Marcio Carneiro Brito Pache
Campus Aquidauana
Instituto Federal de Mato Grosso do Sul
Aquidauana, Brazil
marcio.pache@ifms.edu.br

3rd Wesley Nunes Gonçalves
Inovisão
Universidade Federal de Mato Grosso do Sul
Campo Grande, Brazil
wesley.goncalves@ufms.br

4th Edson Takashi Matsubara
Faculdade de Computação - FACOM
Universidade Federal de Mato Grosso do Sul
Campo Grande, Brazil
edsontm@facom.ufms.br

5thJosé Sabino
Universidade Anhaguera - Uniderp
Campo Grande, Brazil
sabino-jose@uol.com.br

6th Diego André Sant'Ana
Campus Aquidauana
Instituto Federal de Mato Grosso do Sul
Aquidauana, Brazil
diego.santana@ifms.edu.br

7th Hemerson Pistori
Inovisão
Universidade Católica Dom Bosco
Campo Grande, Brazil
pistori@ucdb.br

***Abstract*—Color recognition is an important step for computer vision to be able to recognize objects in the most different environmental conditions. Classifying objects by color using computer vision is a good alternative for different color conditions such as the aquarium. In which it is possible to use resources of a smartphone with real-time image classification applications. This paper presents some experimental results regarding the use of five different feature extraction techniques to the problem of fish species identification. The feature extractors tested are the Bag of Visual Words (BoVW), the Bag of Colors (BoC), the Bag of Features and Colors (BoFC), the Bag of Colored Words (BoCW), and the histograms HSV and RGB color spaces. The experiments were performed using a dataset, which is also a contribution of this work, containing 1120 images from fishes of 28 different species. The feature extractors were tested under three different supervised learning setups based on Decision Trees, K-Nearest Neighbors, and Support Vector Machine. From the attribute extraction techniques described, the best performance was BoC using the Support Vector Machines as a classifier with an F-Measure of 0.90 and AUC of 0.983348 with a dictionary size of 2048.**

***Index Terms*—Aquarium Dataset, Fish Image Classification, Machine Learning, Point of Interest, Color Descriptor.**

I. Introduction

The Brazilian fauna and flora stand out worldwide for their diversity, thus cataloging species of animals and plants is a complicated and arduous task. With such diversity, given a particular animal, even using books and digital databases, it is laborious to identify it. It takes years of study by a professional to specialize in a particular animal. Thus, areas of science, such as artificial intelligence, aim to facilitate these tasks.

Techniques such as the Speeded-up Robust Features (SURF) [1] and Scale-Invariant Feature Transform (SIFT) [2], which describe the points of interest of the images are widely used. This way has been used the Bag Of Visual Words - BoVW [3] in which describing the images based on the regions of interest generated by the SURF. BoVW is a technique based on the Bag of Words, mainly used in the description of texts. As well as techniques that use colors along with other information, the Bag of Features and Colors - BoFC [4] uses BoVW-based concepts by adding color information areas of interest generated by SURF.

Another technique called Bag of Colored Words - BoCW [5], [6] has also been implemented and combines the description of the images supplied by SURF and color information provided by the histogram of color in the HSV space.

Moreover, a technique that uses just color to describe the images, the Wengert's Color Histogram [7] (Bag of Colors - BoC), which when used as a global descriptor uses only color information, creating signatures images and the use of histograms in HSV [8] and RGB [9] colors. Once the image description techniques have been defined, we can learn to classify new images into their respective species.

This paper presents an analysis of computer vision and machine learning techniques to classify fish images through experiments done with attribute extraction techniques in colored images of twenty-eight species of fish on a new dataset totaling 1120 images[1]. The images were obtained employing photographic cameras and smartphone cameras. Initially, there were several fish in the image, but, for the extraction of each species' characteristics, the images have been cropped for just

[1]Available at: https://www.dropbox.com/sh/d8yl5vmuz8ocol2/AADIeJP_edVyKIx31_YDrzJIa?dl=0

one fish per image.

In order to identify the fish species, we evaluate the classifiers based on Decision Trees (C4.5) [10], Support Vector Machine (SVM) [11] and K-Nearest Neighbors (KNN) [12]. The Decision Tree was chosen due to the simplicity of the model used in the classification. The KNN is widely used to classify images and was used in the performance evaluations of the BoC and BoFC. Now, SVM is also popular in object classification, and an example of their use is found in Rova et al. [13].

Therefore, this paper provides results for the classification of aquarium fish images in which two metrics were used to measure the performance of the classifiers and extractors of attributes. This experiment used the F-Measure (F-Scores) criterion of the best parameters of the classifiers and the Area Under the Curve (AUC) as an auxiliary metric and tiebreaker criterion in the choice of parameters.

II. Related Work

Some studies have been found in the literature concerning the classification of fishes. Nery et al. [14] reports that fish classification is not an easy task; according to the same, fish have 47 characteristics that define them, such as color, width, and length. Besides, the images in aquariums are usually obtained with different illuminations making it even more challenging to classify. Using a bayesian classifier and vectors of attributes provided with the mentioned characteristics, the authors presented a classification accuracy higher than 90% using six species of fish.

Rodrigues [15] used an automatic system based on Principal Component Analysis [16] and the Scale-Invariant Feature Transform (SIFT) for the parameterization of shape, appearance, and movement of species. He used two artificial immunological systems (Artificial Immune Network and Adaptive Radius Immune Algorithm) to group the species' characteristics. It obtained 92% accuracy using PCA and Adaptive Radius Immune Algorithm with KNN classifier in nine species of fish.

Matai et al. (2010) [17] developed research in order to automate the process of detection and recognition of the Scythe butterflyfish (Prognathodes falcifer) and flag rockfish (Sebastes rubrivinctus). For the detection process, the Viola and Jones-VJ algorithm based in haar-like features was used to make background subtraction reaching 90% of correct hit ratio for butterflyfish and 49% to rockfish. Principal Component Analysis-PCA and Scale Invariant Feature Transform-SIFT were used for classification, reaching 100% of hit ratio, although with just a small part of the dataset.

Researchers in [18] has developed a set-based approach to fish species identification to video captured from uncontrolled underwater environments. The approach consists of the tracking for separation of categories of species during the training and, new images were tracking in the test, and the system got an overall accuracy of 94.6%.

A machine learning approach was developed by Sengar et al. (2017) [19] for the identification of fish quality through Region of Interest – ROI segmentation of fish eyes and pupil after pesticide exposure. The dataset has 144 images of Indian Rohu (L. rohita) fish in which the proportion of 50% contains cypermethrin pesticide and 50% not. From that 144 images, 80 was separated for training consists of 40 samples with pesticide The Random Forest classifier got the best performance of 96.87% for fish pupil and 93.75% for the fisheye.

Rathi et al. (2017) [20] got good results using Convolutional Neural Networks-CNN and Image Processing for fish image classification in an underwater environment achieving an accuracy of 96.29% tested in the Fish4Knowledge dataset containing 27,142 images.

Allken et al. (2018) [21] also applied CNN for the identification of fish specimens but training with synthetic data from Blue Whiting, Atlantic Herring, and Atlantic Mackerel. The automatic system has a classification accuracy of 94% for synthetic data and 67.2 for real images.

Another CNN-based approach was developed by [22] for fish detection from a computer vision system embedded in an Autonomous Underwater Vehicle (AUV). Since original images were not enough, Data Augmentation was adopted to outperform the training step and algorithm optimization. It was required for time reduction in real-time operation. Thus in the three performed experiments: with Data Augmentation, the better average confidence was 0.65; with the Dropout training loss function, they reached 0.28, and with the refined sum-squared loss function, the prediction reaches a convergent point of 0.27 at 650 iteration time.

Recently, [23] proposed a Deep CNN for automatic fish species identification based on the AlexNet model using four convolutional layers and two fully connected layers. The proposed model was applied for freshwater fish farming images from six species, resulting after data augmentation by zooming, rotation, and flipping in a total of 1334 images. So, they obtained a testing accuracy of 90.48%.

Rauf [24] also take advantage of Deep CNN for automatic identification of fishes species available at the Fish-Pak dataset. It has 941 fish images from six species subdivided into morphological features such as head region, body shape, and fin rays. The best results for a learning rate, 0.001 and momentum 0.9, were 95.73%, 96.02%, and 96.94% of accuracy, respectively, for the head region, body shape, and fin rays. However, it is essential to mention that the Fish-Pak images were preprocessed, removing the background, and the model uses a small amount of data for deep learning techniques.

Yusup [25] developed a deep learning system in order to identify reefs fishes during real-time operations. The dataset consists of 24 species of reef fishes with a total of 9734 images and the labeling was done with Labeling software. The Yolo has been used for species identification and the best results were reach by Pomacanthus imperator with 90.70% in the testing accuracy.

III. Experiments

For each extractor, were generated dictionaries with different sizes that describe each species of fish. In the case of the HSV and RGB histograms, what was varied was the number of bands in which each color channel was divided. The experiments were done with implementations of the classifiers provided by Scikit-learn.

All the classifiers' parameters were varied to obtain the best results in the classification using cross-validation with ten folds. Therefore, the description of the provided images of each technique was submitted to the classifiers with all parameter variations described below.

A. Variations of Classifier Parameters

The parameters of the classifiers were varied as follows:

- SVM: linear and RBF kernels were used. According to Chang et al. [26], the linear core performs best for a large set of attributes, fitting into the context of this work. The RBF core can adapt training sets with non-linear attributes that were even inserted into the experiments. The values of C and γ were varied in the space logarithmic correlation corresponding to the values: $\log_2^{-5}$ to $\log_2^{15}$ for C and $\log_2^{-15}$ to $\log_2^{3}$ for γ.
- KNN: the value of K was varied from 1 to 500, increasing by one. The metrics used to calculate the distance of the points also varied.
- Decision Tree: as the strategy of dividing each node of the tree, it was utilized the best division or a random one. The division criterion was also varied between entropy and Gini impurity.

All parameter permutations were applied to the dictionary variations of the techniques based on the BoW and the variations of histogram ranges.

B. Determination of Parameters of Attribute Extractors

To determine the dictionary's size, that best represents each species' characteristics according to the mentioned extractors, and dictionaries were generated with sizes: 32, 64, 128, 256, 512, 1024, 2048, 4096, and 8192 for BoVW, BoC, BoFC, and BoCW. Because they describe images differently compared to extractors based on BoVW, the number of attributes that describe an image is different in the HSV and RGB histograms.

The variation occurs because the histograms divide each channel from the color space into tracks ranging from 8, 16, 32. The minimum number of tracks has been set in 8 (512 attributes) because for smaller values, the loss of information of color is considerable. The maximum number of tracks is 32 (32,768 attributes) due to hardware limitations found in the computers used in the experiments. The experiments were executed with 10-fold cross-validation, and ten replicates with all the mentioned parameters in Section A.

IV. Results and Discussion

The results of the best parameter settings are shown in the following tables.

1) Experiments with BoVW: In Table I, it is possible to observe that the dictionary of size 4096 obtained better F-Measure and AUC with the SVM classifier. The SVM parameters to achieve this value were as follows: linear core and C = 24826.608981569752.

TABLE I
Decision Tree with BoVW.

Dictionary Sizes	SVM		KNN		Decision Tree	
	F-Measure	AUC	F-Measure	AUC	F-Measure	AUC
32	0.16	0.822183	0.15	0.646302	0.18	0.571267
64	0.17	0.735667	0.14	0.714605	0.22	0.598646
128	0.21	0.803140	0.16	0.705625	0.21	0.6
256	0.31	0.833187	0.19	0.710104	0.25	0.619818
512	0.38	0.874362	0.20	0.750654	0.25	0.638622
1024	0.49	0.902782	**0.23**	**0.739506**	0.26	0.617659
2048	0.57	0.927132	0.21	0.680064	**0.26**	**0.635244**
4096	**0.59**	**0.944135**	0.20	0.650344	0.24	0.634283
8192	0.56	0.933591	0.14	0.560329	0.22	0.634658

2) Experiments with BoFC: The dictionary size that stood out over the others was the size 2048 with F-Measure equal to 0.8. The result was obtained using the SVM with C = 8.86516908684, γ = 8.08386864682e-05 and RBF kernels. Table II illustrates the results of F-Measure for variations of dictionary sizes.

TABLE II
Comparison of BoFC Dictionaries Sets

Dictionary Size	SVM		KNN		Decision Tree	
	F-Measure	AUC	F-Measure	AUC	F-Measure	AUC
32	0.25	0.823231	0.24	0.799302	0.29	0.602282
64	0.42	0.889544	0.39	0.833239	0.40	0.661901
128	0.56	0.929800	0.53	0.908964	0.47	0.727013
256	0.68	0.955215	0.58	0.860894	0.48	0.750654
512	0.76	0.967876	0.62	0.870247	0.53	0.751483
1024	0.77	0.969725	**0.63**	**0.784538**	0.55	0.748871
2048	**0.80**	**0.972607**	0.56	0.754933	**0.56**	**0.748999**
4096	0.78	0.974841	0.49	0.727618	0.53	0.758953
8192	0.79	0.968803	0.37	0.608176	0.49	0.733589

3) Experiments with BoC: The dictionary with the best performance with F-Measure were those of sizes 2048 and 8192. Using the AUC's tiebreaking criterion, the dictionary that best describes the species is size 2048. The SVM parameter settings were the following for the best result found: C = 1635.68097512, = 0.0471890060599, and core RBF. Table III illustrates the results of F-Measure for size variations of dictionaries.

TABLE III
Comparison of BoC Dictionaries Sizes

Dictionary Size	SVM		KNN		Decision Tree	
	F-Measure	AUC	F-Measure	AUC	F-Measure	AUC
32	0.77	0.966132	0.69	0.825876	0.59	0.753064
64	0.83	0.973133	0.75	0.857908	0.64	0.780888
128	0.84	0.974144	0.76	0.862319	0.62	0.800545
256	0.85	0.977770	0.78	0.879424	0.64	0.817035
512	0.89	0.985639	0.80	0.880192	0.65	0.835304
1024	0.88	0.983281	0.82	0.889093	0.69	0.810591
2048	**0.90**	**0.983348**	0.80	0.883555	0.68	0.840596
4096	0.88	0.979725	0.77	0.869325	0.71	0.852780
8192	**0.90**	**0.981404**	**0.90**	**0.864816**	**0.77**	**0.857722**

4) Experiments with BoCW: Because BoCw is a BoVW-derived technique histogram, the number of attributes that each image describes vary in the size of the dictionary used by BoVW and the total bands of the histogram. Thus, for each

dictionary size, the number of tracks ranges from 2, 4, 8, 16, and 32. The 512 and 1024 size dictionaries, both with 32 tracks, obtained the same F-Measure score of 0.88. However, taking the AUC as a tiebreaker, the dictionary of size 1024 was chosen. The best results were obtained by a track range of 32, as shown in Table IV.

TABLE IV
COMPARISON OF BOCW DICTIONARY SIZES USING 32 TRACKS.

Dictionary Size	SVM		KNN		Decision Tree	
	F-Measure	AUC	F-Measure	AUC	F-Measure	AUC
32	0.87	0.978157	0.66	0.772911	0.71	0.835395
64	0.87	0.978598	**0.68**	**0.762312**	**0.72**	0.825212
128	0.86	0.961626	0.66	0.738020	**0.72**	**0.838937**
256	0.86	0.971717	0.65	0.743238	0.71	0.822060
512	**0.88**	**0.972709**	0.65	0.730703	0.72	0.840050
1024	**0.88**	**0.973656**	0.63	0.697975	0.71	0.821125
2048	0.87	0.972574	0.59	0.650194	0.71	0.824874
4096	0.72	0.970887	0.27	0.669057	0.71	0.819003
8192	0.68	0.967433	0.19	0.590258	0.69	0.816841

5) Experiments with the HSV Color Histogram: Table V presents the comparison of the results obtained using the color histogram with HSV color space, the best F-Measure obtained was 0.89 along with an AUC equal to 0.981666 using 32 tracks with SVM classifier with RBF core, $C = 16833.4006851$ and $\gamma = 0.00210510528871$.

TABLE V
COMPARISON OF THE NUMBER OF HSV COLOR HISTOGRAM BANDS.

Tracks	SVM		KNN		Decision Tree	
	F-Measure	AUC	F-Measure	AUC	F-Measure	AUC
04	0.83	0.959969	0.80	0.898065	0.74	0.859452
08	0.86	0.967860	**0.86**	**0.913217**	**0.83**	**0.876275**
16	0.87	0.974043	0.77	0.883935	0.71	0.850113
32	**0.89**	**0.981666**	0.70	0.851477	0.69	0.838484

6) Experiments with the RGB Color Histogram: The best F-Measure obtained in the experiments using the RGB color histogram was 0.88 with 16 tracks, as shown in Table VI. The parameters of the SVM classifier were linear core and $C = 6.3540113525$.

TABLE VI
COMPARISON OF THE NUMBER OF TRACKS IN THE RGB COLOR HISTOGRAM.

Tracks	SVM		KNN		Decision Tree	
	F-Measure	AUC	F-Measure	AUC	F-Measure	AUC
04	0.86	0.951616	0.83	0.913880	0.73	0.876630
08	0.87	0.963213	**0.85**	**0.920641**	**0.76**	**0.863251**
16	**0.88**	**0.974300**	0.81	0.908237	0.75	0.854777
32	0.87	0.986589	0.72	0.865965	0.70	0.844429

The result obtained made use of the dictionaries with higher F-Measure and the optimized classifiers, and the attribute extractors were compared to find out the best classifier. For greater reliability of results, the Friedman hypothesis test [27] was applied to evidence the statistical difference of the results.

Table VII illustrates the performance of the SVM classifier concerning F-Measure and AUC of each attribute extractor, since it was the classifier that obtained the best results. Thus, the statistical tests were performed using the F-Measure of the dictionaries illustrated in Table VII. Analyzing the attribute extractors, BoC obtained the best result, followed by RGB and HSV histograms.

TABLE VII
COMPARISON OF THE F-MEASURE METRIC WITH THE USE OF THE SVM CLASSIFIER FOR EACH EXTRACTOR.

Extractor	Dictionary/Track	F-Measure	AUC
BoVW	4096	0.59	0.944135
BoC	**2048**	**0.90**	**0.983348**
BoFC	2048	0.80	0.972607
BoCw	1024/16 tracks	0.88	0.973656
Color Histogram - HSV	32 tracks	0.89	0.981666
Color Histogram - RGB	16 tracks	0.88	0.974300

Figure 1 illustrates the box diagram of the F-Measure values for each technique. It is possible to observe that the BoC attribute extractor and HSV and RGB histograms have F-Measure values around 0.9.

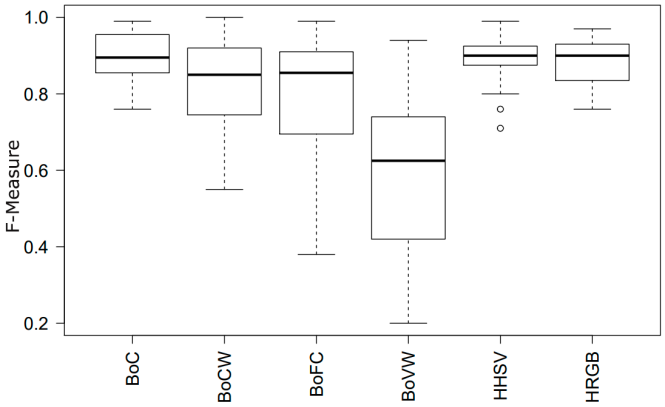


Fig. 1. Box plots of techniques using F-Measure.

The Friedman test obtained a *p-value* of 1.035×10^{-08} for F-Measure and 1.275×10^{-13} for AUC. Therefore, the *post-test* [28] was done, and the results are shown in Table VIII. According to the *post-test*, it is observed that the color-based extractors have a statistical difference with the BoVW, which does not use color to describe the images. As for the color-based classifiers, according to the Friedman test, they do not present statistical differences between them.

TABLE VIII
Post hoc TEST.

Extractors	BoFC	BoC	BoCW	Histogram HSV	Histogram RGB
BoVW	9.848803×10^{-03}	6.166622×10^{-08}	4.682090×10^{-08}	7.713548×10^{-09}	7.174929×10^{-07}
BoFC	•	1.578712×10^{-01}	1.029825×10^{-01}	7.800627×10^{-02}	3.238798×10^{-01}
BoC	•	•	9.999729×10^{-01}	9.997243×10^{-01}	9.991803×10^{-01}
BoCW	•	•	•	9.999979×10^{-01}	9.943114×10^{-01}
Histograma HSV	•	•	•	•	9.868104×10^{-01}

Figure 2 represents the Friedman *post-test* box diagram, obtained from the Table VIII. As shown, it is possible to observe that the extractors (represented by green boxes) more distant in relation to the y = 0 axis are those that present the greatest difference between the interquartile range.

The difference in the performance of BoVW about other techniques is noticeable. This fact is due to the exclusive use

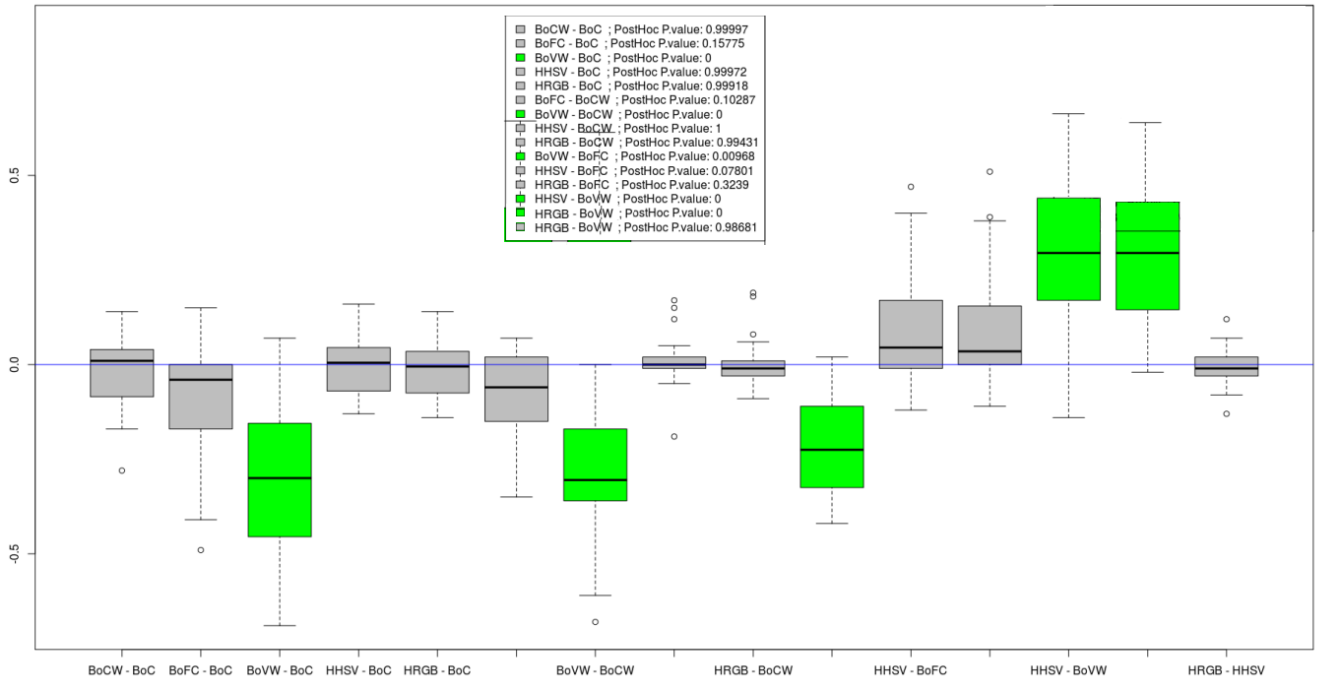


Fig. 2. Friedman post-test box diagram.

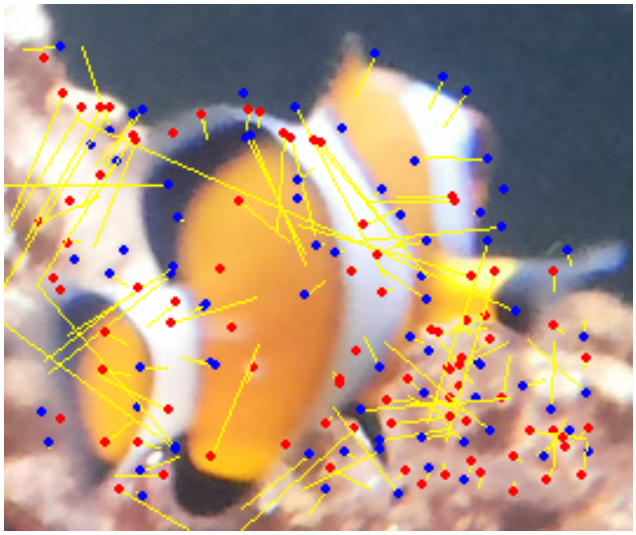


Fig. 3. Example of points of interest found in an image of the Clown fish.

of the cluster of points of interest without taking into account the other characteristics of the fish. The BoVW is based only on points of interest, and they can be found on the fish and/or the background of the image. An example of noise found in one of the images used in the experiments is illustrated in Figure 3, where it is possible to identify that most of the points of interest were found in the background of the image. Thus, much of the visual words formed are obtained from points that do not belong to the fish, impairing the description.

Nevertheless, the problem of the background of the image has no significant impact on the description of the images when the color is used. This is because, given a set of images relating to a species of fish belonging to a specific aquarium, the background of the images tends to have the same colors since the lighting and the aquarium are the same. Thus, in techniques such as BoC where colors are grouped, it is likely that color is formed corresponding to the average color of the image's background. In the case of BoC, a core representing a color signature corresponding to the background. Moreover, because the training images are from the same aquarium, the average color tends to have a uniform distribution in the description of the images, making it less discriminating in the classification.

The BoFC presented a good result compared to the techniques that use color in the description of the images. Although the same problem was found in the use of points of interest is found in the BoFC, inserting color into the description has attenuated the problem. The fact that the points of interest had color information in their description diminished the impact of the points found in the bottom of the image. For the same reason, cited in the BoC analysis, and because of this reason, the BoFC obtained better results than the BoVW.

V. Conclusions

In this paper, we proposed an aquarium fishes identification method based on point of interest feature extraction, especially color. The fact of grouping the colors of the background of the

image is evidenced mainly in the BoCw technique. Observing the Tables I to VI in Section IV, the AUC values from the color information are proportional to the F-Measure values. Initially, for a small number of attributes, the results are close to BoVW due to lower color information compared to the same variety of dictionary size (32, 64, ..., 8192). However, increasing the amount of bands, the values of F-Measure also increase, evidencing the importance of the color in the description. The color histograms also highlight the importance of color in the classification and the color grouping of the background of the image, since the division of color space into stripes is a way of grouping. For future experiments, we can use other types of attribute extraction techniques and also deep learning.

ACKNOWLEDGMENT

This work has received financial support from the Universidade Católica Dom Bosco, the Foundation for the Support and Development of Education, Science and Technology from the State of Mato Grosso do Sul - FUNDECT (131/2016) and this study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and CNPq (National Council for Scientific and Technological Development) through research grants (p. 314902/2018-0). Thanks to Nvidia Corporation for donating the GPU.

REFERENCES

[1] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008. [Online]. Available: http://dx.doi.org/10.1016/j.cviu.2007.09.014

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [Online]. Available: http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94

[3] L. F. J. W. G. Csurka, C. Dance and C. Bray, "Visual categorization with bags of keypoints," *Workshop on statistical learning in computer vision*, vol. 1, pp. 1–22, 2004.

[4] A. Bahri and H. Zouaki, "A surf-color moments for images retrieval based on bag-of-features," in *European Journal of Computer Science and Information Technology Vol.1 No.1, pp.11-22, June 2013*. Department of Mathematics and computer science, Faculty of Science, El Jadida, Morocco: EAJ, Jun. 2013, pp. 11–22. [Online]. Available: http://www.eajournals.org/journals/european-journal-of-computer-science-and-information-technology-ejcsit/vol-1-issue-1-june-2013/a-surf-color-moments-for-images-retrieval-based-on-bag-of-features/

[5] C. S. Joost van de Weijer, "Coloring local feature extraction," *European Conference on Computer Vision (ECCV '06)*, pp. 334–348, 2006.

[6] J. van de Weijer1 and F. S. Khan, "Fusing color and shape for bag-of-words based object recognition," *Computational Color Imaging. CCIW 2013. Lecture Notes in Computer Science*, vol. 7786, 2013.

[7] C. Wengert, M. Douze, and H. Jégou, "Bag-of-colors for improved image search," in *MM 2011 - 19th ACM International Conference on Multimedia*. Scottsdale, United States: ACM, Nov. 2011, pp. 1437–1440, qUAERO. [Online]. Available: http://hal.inria.fr/inria-00614523

[8] S. Sural, Gang Qian, and S. Pramanik, "Segmentation and histogram generation using the hsv color space for image retrieval," in *Proceedings. International Conference on Image Processing*, vol. 2, 2002, pp. II–II.

[9] R. Chakravarti and X. Meng, "A study of color histogram based image retrieval," in *2009 Sixth International Conference on Information Technology: New Generations*, 2009, pp. 1323–1328.

[10] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.

[11] V. N. Vapnik, "An overview of statistical learning theory," *Trans. Neur. Netw.*, vol. 10, no. 5, pp. 988–999, Sep. 1999. [Online]. Available: http://dx.doi.org/10.1109/72.788640

[12] N. Altman, "An introduction to kernel and nearest neighbor nonparametric regression," *The American Statistician*, vol. 46, pp. 175—-185, 1992.

[13] A. Rova, G. Mori, and L. M. Dill, "One fish, two fish, butterfish, trumpeter: Recognizing fish in underwater video," in *In IAPR Conference on Machine Vision Applications*, 2007.

[14] M. S. Nery, A. M. Machado, M. F. M. Campos, F. L. C. Pádua, R. Carceroni, and J. P. Queiroz-Neto, "Determining the appropriate feature set for fish classification tasks," *Graphics, Patterns and Images, SIBGRAPI Conference on*, vol. 0, pp. 173–180, 2005.

[15] M. T. A. Rodrigues, "Classifição automática de espécies de peixes baseada em técnicas robustas para extração de caracteriisticas e sistemas imunológicos artificiais," Mestrado, Belo Horizonte : Centro Federal de Educacao Tecnologica de Minas Gerais, 2009.

[16] I. Jolliffe, *Principal component analysis*, ser. Springer series in statistics. Springer-Verlang, 1986. [Online]. Available: http://books.google.com.br/books?id=cN5UAAAAYAAJ

[17] J. Matai, R. Kastner, G. R. Cutter, and D. A. Demer, "Automated techniques for detection and recognition of fishes using computer vision algorithms," 2012.

[18] F. Shafait, A. Mian, M. Shortis, B. Ghanem, P. F. Culverhouse, D. Edgington, D. Cline, M. Ravanbakhsh, J. Seager, and E. S. Harvey, "Fish identification from videos captured in uncontrolled underwater environments," *ICES Journal of Marine Science*, vol. 73, no. 10, pp. 2737–2746, 07 2016. [Online]. Available: https://doi.org/10.1093/icesjms/fsw106

[19] N. Sengar, M. K. Dutta, and B. Sarkar, "Computer vision based technique for identification of fish quality after pesticide exposure," *International Journal of Food Properties*, vol. 20, no. sup2, pp. 2192–2206, 2017. [Online]. Available: https://doi.org/10.1080/10942912.2017.1368553

[20] D. Rathi, S. Jain, and S. Indu, "Underwater fish species classification using convolutional neural network and deep learning," *2017 Ninth International Conference on Advances in Pattern Recognition (ICAPR)*, Dec 2017. [Online]. Available: http://dx.doi.org/10.1109/ICAPR.2017.8593044

[21] V. Allken, N. O. Handegard, S. Rosen, T. Schreyeck, T. Mahiout, and K. Malde, "Fish species identification using a convolutional neural network trained on synthetic data," *ICES Journal of Marine Science*, vol. 76, no. 1, pp. 342–349, 10 2018. [Online]. Available: https://doi.org/10.1093/icesjms/fsy147

[22] S. Cui, Y. Zhou, Y. Wang, and L. Zhai, "Fish detection using deep learning," *Applied Computational Intelligence and Soft Computing*, vol. 2020, p. 3738108, Jan 2020. [Online]. Available: https://doi.org/10.1155/2020/3738108

[23] M. A. Iqbal, Z. Wang, Z. A. Ali, and S. Riaz, "Automatic fish species classification using deep convolutional neural networks," *Wireless Personal Communications*, Aug 2019. [Online]. Available: https://doi.org/10.1007/s11277-019-06634-1

[24] S. Z. S. Z. H. S. A. U. R. S. A. C. B. Hafiz Tayyab Raufa, M. Ikram Ullah Lalia, "Visual features based automated identification of fish species using deep convolutional neural networks," *Computers and Electronics in Agriculture*, vol. 167, no. 105075, 2019.

[25] M. I. I M Yusup and I. Jaya, "Real-time reef fishes identification using deep learning," *IOP Conference Series: Earth and Environmental Science*, vol. 429, 2019.

[26] C.-J. Chang, Chih-Chung e Lin, *LIBSVM: A Library for Support Vector Machines*. ACM Transactions on Intelligent Systems and Technology, 2011.

[27] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *The Annals of Mathematical Statistics*, no. 1, pp. 86–92. [Online]. Available: http://www.jstor.org/stable/2235971

[28] M. Hollander and D. Wolfe, *Nonparametric Statistical Methods*, ser. A Wiley-Interscience publication. Wiley, 1999. [Online]. Available: http://books.google.com.br/books?id=RJAQAQAAIAAJ

Improving the network traffic classification using the Packet Vision approach

Rodrigo Moreira*†, Larissa Ferreira Rodrigues*
*Instituto de Ciências Exatas e Tecnológicas
Universidade Federal de Viçosa – UFV
Rio Paranaíba - MG - Brasil
Email: {rodrigo, larissa.f.rodrigues}@ufv.br

Pedro Frosi Rosa†, Flávio de Oliveira Silva†
†Faculdade de Computação - FACOM
Universidade Federal de Uberlândia – UFU
Uberlândia - MG - Brasil
Email: {rodrigo.moreira, pfrosi, flavio}@ufu.br

***Abstract*—The network traffic classification allows improving the management, and the network services offer taking into account the kind of application. The future network architectures, mainly mobile networks, foresee intelligent mechanisms in their architectural frameworks to deliver application-aware network requirements. The potential of convolutional neural networks capabilities, widely exploited in several contexts, can be used in network traffic classification. Thus, it is necessary to develop methods based on the content of packets transforming it into a suitable input for CNN technologies. Hence, we implemented and evaluated the Packet Vision, a method capable of building images from packets raw-data, considering both header and payload. Our approach excels those found in state-of-the-art by delivering security and privacy by transforming the raw-data packet into images. Therefore, we built a dataset with four traffic classes evaluating the performance of three CNNs architectures: AlexNet, ResNet-18, and SqueezeNet. Experiments showcase the Packet Vision combined with CNNs applicability and suitability as a promising approach to deliver outstanding performance in classifying network traffic.**

***Keywords*—Network traffic classification; convolutional neural networks; SDN; data augmentation; fine-tuning.**

I. Introduction

Classifying network traffic allows us to know the kind of application running on the network, benefiting the models for forecasting, capacity utilization, quality of service, security, and planning and management steps. Besides, in the frameworks of new communication, network architectures require intelligent entities to support resources management and operation. Traffic classification mechanisms are known and widely explored in the state-of-the-art, however, with the advent of convolutional neural networks (CNNs), new methods of training, validation, and classification are available, especially those based on images raising the opportunity to propose and evaluate mechanisms for network traffic classification [1] [2].

Among the known traffic classification mechanisms, we can categorize them as port-based, payload-based, machine-learning approaches based on statistics and deep learning [3]. In particular, CNNs demonstrate capabilities beyond its fields of action with highly accurate mechanisms for clustering and classifying medical images [4], biomolecular [5], environmental [6], and others contexts [7]. The success of CNNs is due to their ability to incorporate spatial context and weight sharing between pixels in order to extract high-level hierarchical representations of the data [8].

In this sense, we employ the capabilities of CNNs for processing packets of data communication networks. The graphics processing supported by GPU hardware surpasses the CPU-based processing because reducing the execution time [9]. Hence, the speedup of time-to-ready of traffic classification technologies is reducing [10], enabling faster classification.

Recent studies demonstrated effective results in network traffic classification using deep CNNs [1] [2] . However, these studies performed the classification splitting both header and payload of packets as a learning feature. In a real scenario, this approach may generate security and time issues, regarding the last one, it may increase the pre-processing time without guaranteeing gains in classification performance metrics.

In this paper, we proposed the Packet Vision: a method based on computer vision to generate images from both payload and packet header. Our main contribution relies on generating a single image representing all content of the network packet. Other approaches for traffic classification, such as those based on the packet signature [11], conflict with security, and privacy aspects, since information as the source and destination address, port, and transport protocol, to name a few are handling as plain text, making straightforward inference by malicious third-parties. Furthermore, a novel contribution of this paper is an evaluation of the performance of three state-of-the-art CNNs for the network traffic classification via training from scratch and fine-tuning.

Our results showcase the suitability and performance score of Packet Vision in generating and classifying images of packets from communication networks considering from raw-data. Besides, we considered three classification technologies based on CNNs, applying a hypothesis test to judge the performance between them.

The remaining of this paper is organized as follows: Section II surveys related work. Section III presents our approach for network traffic classification. The CNNs evaluated in this paper, and the protocol used in the experiments are presented in Section IV. Section V presents and discusses the results. Finally, we provide concluding remarks and future work agenda in Section VI.

II. Related Work

Lim et al. [2], proposed a traffic classification mechanism aims to improve the quality of service for applications without

interference from the network operator. Its structure generates a dataset containing images of flows analyzed over time intervals. The approach uses CNN and Long short-term memory (LSTM) to train and evaluate the classification performance using the F1-score metric. The proposed architecture considers three layers, the lowest containing data switches, including switches and hosts that exchange data between their own, on top of previous the control including classification mechanisms and traffic entities. The topmost layer allows the implementation of specific network behaviors based on the type of traffic.

The image generation mechanism for the dataset comprises capturing the flow: a set of packets with similar characteristics (source and destination host, port, and transport protocol) in a specific time interval. Therefore, for each packet of a flow, extracts its payload and performs mathematical operation over a set of bits to transform it into a single numerical value, consequently a single pixel. Thus, a single figure, containing many pixels, is the set of packet representing an application's flow. This approach does not carry out cross-validation and disregard the entire package structure, requiring additional computation in the processing step that consists of extracting the payload of each package.

Vasan et al. [12] proposed an architecture of CNN and evaluates the virtual threats as malware close classification real-time. The construction of the dataset transforms the binary signature of malware, which is an 8-bit vector into an 8-bit array, afterward in a grayscale figure and then applying a 2-D color map. The classification performance evaluation takes into account approaches with data augmentation and fine-tuning. Unlike the present proposal, this article proposes cross-validation to avoid bias and over model adjustment, besides there is no need to transform the image of the 2-D color maps dataset, maintaining performance.

Chen et al. [13] presents an IP traffic classification framework based on CNNs named Seq2Img. This approach consists of capturing the packets of a flow and extracting its characteristics and behaviors. A probability distribution model called Reproducing Kernel Hilbert Space (RKHS) is mandatory to construct the figures for each traffic class, consisting of the network protocols and popular social networking applications. Accuracy was the performance metric held in the validation of the traffic classification model. Unlike the present paper, the authors did not validate the proposal with hold-out, and the data collection mechanism depends on a third non-open source application. On the other hand, our approach consists of an open-source collector and does not handle images as flows and does not require processing with complex mathematical models.

Wang et al. [1] proposed a framework for classifying malicious traffic in domestic environments through home-gateway equipment containing an embedded traffic prediction mechanism. The mechanism based on CNNs is similar to ours because they take into account the figure from each package as a data suitable for Machine learning models. However, different from us in the pre-processing stage, the ethernet header of the package is removed. Besides, to avoid bias and overfitting in the training model, we applied, according to a probability distribution, we shuffle the image pixels of each class. Thus, packages containing the same source and destination address do not keep standardized pixels in predefined locations. The dataset images built from a set of packet captures of typical Internet standard applications.

Other works are known in the state-of-the-art, proposing a network traffic classification targeting security, quality of service enhancement, management, and others [14], [15], [16]. They vary in terms of the learning and validation method, also differs between strategies based on port, payload, statistics, CNNs, flows, and others [17]. The Packet Vision innovates by drawing the packets entirely, considering header and payload, and by creating a deep learning model considering those images generated through packets raw data.

III. Packet Vision

The resource sharing turn up in different ways in the literature. The architecture of the operating systems, especially those for time-sharing processing, has been inspiring new formats of resource sharing, impacting computing resources, and network sharing. Sharing network resource relies on to assign part of general-purpose hardware to a specific user while safeguarding essential aspects of isolation and guarantees. In the context of mobile networks, especially in the 5G standardization, sharing took the form of network slicing, which provides logical networks with independent data and control plans for users to meet specific application requirements.

Therefore, among the network slicing approaches rising the Network and Slice Orchestrator (NASOR) [18] that implements the network slicing beyond the mobile network ecosystem, providing logical connectivity over the Internet data plane. The NASOR ecosystem includes interfaces that facilitate network slice management, called the Open Policy Interface (OPI). The OPI interface allows third-party mechanisms to support network slicing and management. Consequently, we propose a component that performs this interface, offering traffic classification to lead the NASOR path configuring agent, called Packet Vision.

The Packet Vision is a method, originated from the drawing-packet action, capable of receiving a network packet in the raw format and transforming it into images considering both the header and the payload. After generating images, it is possible to classify them according to the traffic class. Traffic classes range across the network according to the overlying application. This classification guides the network slicing agent as to the path that logical connectivity must take along Internet routers. We present Packet Vision as a method of building a dataset of network traffic class images to train and evaluate deep learning algorithms. Hence, Fig. 1 depicts Packet Vision as a method for creating Datasets.

The first step comprises collecting network packets carried over a network interface. The open-source application Wireshark and its extension libraries allow collecting packet from a network interface without affecting the application. The Packet Vision handles packets traces from four sources, collected through the open-source tool Wireshark, containing *pcap* files for each traffic class.

The first class of traffic is the standard of IoT Applications containing around 27 heterogeneous devices such as sensors and actuators [19]. The second packet trace comprises conventional Internet applications containing DNS and BitTorrent classes [20], also available in *pcap* format.

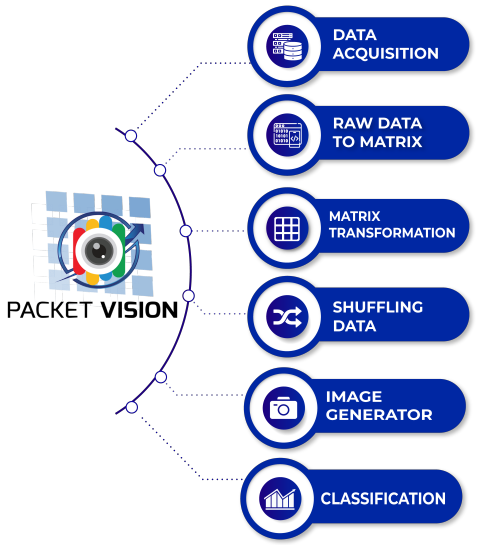


Fig. 1. Packet Vision proposed method.

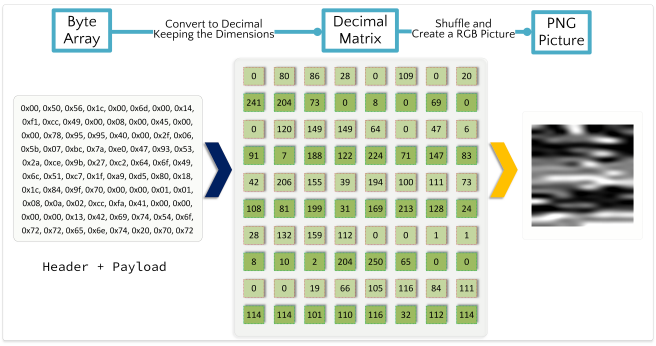


Fig. 2. Building Dataset.

The raw information of the packet available in this dataset had been processing in order to generate figures for each class. Finally, the third packet trace refers to network slice deployed through NASOR, considering three network domains [18]. Hence, a VoIP application providing communication between entities being in domain A targeting domain B communicating with voice chunks processed by codec G.711. Our method combines three packet traces making it possible to build a dataset of figures containing four traffic classes: BitTorrent, DNS, VoIP, and IoT.

The watcher captures the packets and presents it differently; bits is the conventional form of the physical layer. However, they are grouping in formats with semantic values, such as byte array, plain text, to name a few. Hence, the second step of the method consists of handling the data in raw format, distributed in an array of bytes, and transforming them into a matrix. In this sense, our method considers the data grouping model in the Array format, which presents the packet information in hexadecimal composition.

Regarding the second step, turn the hexadecimal byte array into a matrix whose size is $n \times 8$, where n represents the number of rows and 8 the number of columns according to Fig. 2. The matrix columns are 8 in size due to the native implementation of the Wireshark raw extractor library. The size of the packets, measured in bytes, varies among applications, so the method considers the number of columns fixed at 8, and the number of rows in the matrix is variable to accommodate the size of the packet in bytes. There are scenarios where the packet size in bytes is not $n \times 8$, requiring that bytes-padding appending at the end of the packet. We agree that bytes-padding is always *0xFF* for all traffic classes. Thus, when processing the matrix $n \times 8$ of hexadecimal and constructing the dataset organized in classes, these will contain figures of size $n \times 8$ pixels.

The third stage of the method considers as essential to convert the hexadecimal matrix, previously created, into decimal format. At the end of this step, the fourth shuffles the decimal values of the matrix to avoid bias and overfitting in the deep learning model. Shuffling is mandatory to change the fixed place of packet headers, such as source host, destination host, port, to name a few. Our shuffling method held Poisson probability distribution over the decimal matrix, handling the security and privacy lack in the state-of-the-art. The decimal values representing the header may remain at fixed locations in the matrix, regardless of the package content. Therefore, the fourth stage performs a shuffling of the values according to a Poison probability distribution.

The fifth step of the proposed method consists of adding RGB channels according to each decimal in the matrix, maintaining the color intensity for the three channels. The fifth step brings PNGs figures representing the contents of the packet, including the headers and the payload as an image texture. Headers are the addressing information essential to the entire packet deliver, and the payload is the information carried.

The information about the created dataset has been summarizing in both Table I and Fig. 3, where the last one depicts examples of how Packet Vision can draw packets categorizing them into classes. This dataset are available at <https://romoreira.github.io/packetvision/> under open-source license.

TABLE I. DISTRIBUTION OF IMAGES BY CLASSES.

Class	Samples
Bit Torrent	1217
DNS	1412
VoIP	1320
IoT	1848
Total	5797

The sixth step includes training and validating the deep learning mechanism that uses the properly labeled figures from the created dataset. Many convolutional neural network architectures are known, so it is necessary to evaluate the performance of some to identify the most suitable for this kind of problem. After training and validating the learning model based on the figures generated through the raw packets, the characteristic of the current traffic on the network may be collected from a given network channel, by sample or for a determined time.

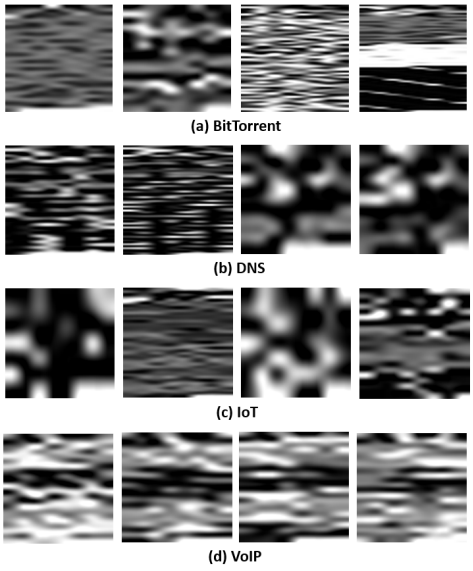


Fig. 3. Network packets samples generated from Packet Vision.

Other methods for building images from the packet are known [13], [21], [22], although they do not handle the complete packet structure. Alternatively, our method does not require the header and the packet payload separating in advance, causing additional processing. Besides, by shuffling the packet bytes in the matrix highlights our method regarding privacy, it is not straightforward to achieve the original semantics of the packet, including a source, destination, transport protocol port, and others from generated image.

IV. Classification Method

In this study, the classification was performed using CNNs, which uses multi-layer neural networks to learn features and classifiers in different layers, at running time, and does not require handcrafted feature extraction [23]. Three state-of-the-art CNN architectures were selected based on their past performance in image classification tasks: AlexNet [24], ResNet-18 [25], and SqueezeNet [26].

AlexNet [24] was the champion of ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 and is responsible for the recent popularity of neural networks. This CNN has five convolutional layers, three max-pooling layers, two fully connected layers with a final softmax. It was a breakthrough architecture since it was the first to employ non-saturating neurons and dropout connections to prevent overfitting.

ResNet, presented in [25], was the champion of ILSVRC 2015 [27] and has several variations with 18 to 152 layers. This network has a series of residual blocks, each composed of several stacked convolutional layers. This configuration allows accelerating the convergence of the deep layers without overfitting. In this study, we choose to work with the ResNet-18 for the sake of simplicity.

SqueezeNet [26] has a compact architecture with approximately 50 times fewer parameters than AlexNet. This CNN reduces parameters through 1×1 convolutions and eight fire modules, which performs the functions of fully connected and dense layers.

We consider two strategies of training: from scratch and fine-tuning. In training from scratch, we initialize all parameters randomly, and during the training, the values of the parameters were learned directly from the dataset in all layers [23], achieving better results when compared with training based on fine-tuning since the CNN learns specific features [6] [4]. The fine-tuning strategy was performed over models pre-trained on the ImageNet dataset and consists of fine-tuning the parameters in the deeper layers [8]. For both training strategies, the dimension of the last fully connected layer was four, according to the number of classes.

In order to compare the CNN architectures, we trained and tested using stratified k-fold cross-validation method [28]. The cross-validation was repeated five times, for each iteration, one of the training folds is chosen for the test and the others for training. Also, was taken the average accuracy, precision, recall, and f1-score, measured from the confusion matrix [29].

V. Results and Discussion

All experiments were performed on a machine with an Intel i5 3.00 GHz processor, 16 GB RAM, and a GPU NVIDIA GeForce GTX Titan Xp with 12 GB memory. The experiments were programmed using Python (version 3.6) and PyTorch [30] (version 1.4) deep learning framework.

We trained the CNN architectures using Stochastic Gradient Descent (SGD) [31] optimizer, with a learning rate of 0.001, the momentum of 0.9, batch size of 32, and 50 epochs for both, training from scratch and fine-tuning. All images were resized to 224×224 pixels to adapt for the input of the CNNs evaluated. The training images had augmented through vertical and horizontal flips, rotating images around its center through randomly chosen angles of between 0° and 360°.

Our experiments aim to answer the following questions:

1) What is the highest classification performance among three evaluated CNNs?
2) Considering accuracy, training from scratch, and fine-tuning, what is the most suitable training method for this dataset?
3) Is the performance of pre-trained CNNs statistically equivalent?

To assess the impact of the training from scratch and fine-tuning, we analyze the classification performance of each CNN architecture according to metrics of accuracy, precision, recall, and f1-score. Regarding the classification performance, the Tables II and III presents the average 5-fold cross-validation for each CNN considering training from scratch and fine-tuning, respectively. As shown, the best performance results are achieving with the training from scratch. Consequently, the best result among the three has been obtaining by the AlexNet architecture, especially the strategy which use from scratch training.

Although the fine-tuning technique did not improve the performance indices compared to training from scratch, this approach requires less time to train the unfrozen layers and could be suitable in real scenarios (see Table IV). Thus, we

TABLE II. 5-FOLD AVERAGE VALUES OF THE PERFORMANCE INDICES FOR EACH CNN ARCHITECTURE TRAINING FROM SCRATCH.

CNN	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
AlexNet	**100.00**	**100.00**	**100.00**	**100.00**
ResNet-18	99.80	100.00	100.00	100.00
SqueezeNet	99.60	99.80	99.60	99.60

TABLE III. 5-FOLD AVERAGE VALUES OF THE PERFORMANCE INDICES FOR EACH CNN ARCHITECTURE TRAINING WITH FINE-TUNING.

CNN	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
AlexNet	95.40	96.00	95.80	95.80
ResNet-18	96.40	96.40	96.80	96.40
SqueezeNet	**97.60**	**97.80**	**97.40**	**97.60**

compared only the pre-trained CNNs in order to identify the best model.

TABLE IV. AVERAGE TRAINING TIME FOR EACH CNN ARCHITECTURE CONSIDERING BOTH TRAINING STRATEGIES.

	Training Time (minutes)	
CNN	**From-scratch**	**Fine-tuning**
AlexNet	16.21	06.21
ResNet-18	37.00	14.13
SqueezeNet	27.41	12.29

According to the results presented in Table IV, although training from scratch achieves high accuracy, the most suitable for this dataset considering the impact of computational cost is the SqueezeNet architecture trained with fine-tuning. Since, in real network traffic classification scenarios, approaches with lower computational cost are more appropriate.

To assess the performance, we carried Z-Test with 95% of confidence over samples of Table V, which contains accuracy obtained from each test set. Thus, considering AlexNet and ResNet-18, we raise the following hypotheses: H_0 – the performance of AlexNet is equal to or less than ResNet-18. On the other hand, H_a – the performance of AlexNet is higher than ResNet-18. Considering the sample space of size five, we can infer the observed $Z_{obs.}$ is lower than $Z_{crit.}$, leading us to accept H_0, implying that the performance of AlexNet is equal to or less than ResNet-18.

TABLE V. 5-FOLD TEST ACCURACY FOR EACH CNN ARCHITECTURE TRAINING WITH FINE-TUNING.

Fold	*AlexNet (%)*	*ResNet-18 (%)*	*SqueezeNet (%)*
1	93.00	95.00	96.00
2	97.00	97.00	98.00
3	98.00	98.00	98.00
4	93.00	95.00	97.00
5	96.00	97.00	99.00

Besides, we infer the performance of ResNet-18 and SqueezeNet, raising two hypotheses, namely H_0 – the performance of ResNet-18 is less than or equal to SqueezeNet. At the same time, H_a – the performance of ResNet-18 is higher than SqueezeNet. Considering a sample space with size five, and a normal distribution, the observed $Z_{obs.}$ is outside the critical region, which leads us to accept H_0, implying that ResNet-18 is less than or equal to SqueezeNet.

Hence, SqueezeNet architecture pre-trained with ImageNet had been performed better than or equal to its peers. These results suggest the suitability of Packet Vision to act as a traffic classifier mechanism and, eventually, enabling its embodiment on low-cost hardware such as Raspberry Pi.

Finally, considering the best result for each training strategy (from-scratch and fine-tuning), the charts in Fig. 4 show how each CNN architecture behaved during the training stage, considering the average loss and accuracy of the 5-folds. The results show that CNNs maintained the generalization property.

Fig. 4. Average 5-fold training loss and accuracy considering the best training strategy. (a) AlexNet training from-scratch; and (b) SqueezeNet training with fine-tuning.

VI. CONCLUDING REMARKS

This paper presents the Packet Vision method for building and evaluating datasets representing traffic on communication networks through CNNs. This method allows representing the raw-data of network packets in images for training and classification in a deep learning mechanism. The image creation mechanism considering the header and the payload advances the state-of-the-art since its peers consider only the payload, among other approaches such as the semantic and statistical representation of flows. Besides, our approach is suitable for classifying traffic with similar characteristics implying in challenging tasks, achieving excellent performances according to state-of-the-art metrics, and its implementation in the network being direct by handling the packets as they are.

Carried experiments showcase that SqueezeNet performance is at least equal or higher against AlexNet and ResNet-18 trained with fine-tuning, enabling us to answer questions about the quality of CNNs performance. Besides, we point out training approaches suitability for this problem, including a statistical test seeking possible performance equivalence. Also, unlike the approaches found in the state-of-the-art, the Packet Vision shuffling step enhances the privacy claim upon packets, avoiding fixed fields of the packets at the same pixel location, avoids rebuilding the original packet from the image.

We believe that Packet Vision is a robust application for the traffic network classification with a significant degree of innovation stemming from computer vision techniques applying to generate images from packets raw-data. Moreover, the Packet Vision seems suitable for future networks, such as 5G and beyond, whose take into account the security, privacy, and application-aware as a baseline.

As future work, we intend to exploit the Packet Vision approach to generate other traffic classes related to distinct applications, such as Remote Desktop Protocol (RDP), SSH, and social media. We are also planning to evaluate other CNN architectures, data augmentation strategies, and hyperparameter optimization.

ACKNOWLEDGMENTS

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the TITAN Xp GPU used for this research. And also, this study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

REFERENCES

[1] P. Wang, F. Ye, X. Chen, and Y. Qian. Datanet: Deep learning based encrypted network traffic classification in sdn home gateway. *IEEE Access*, 6:55380–55391, 2018.

[2] Hyun-Kyo Lim, Ju-Bong Kim, Kwihoon Kim, Yong-Geun Hong, and Youn-Hee Han. Payload-based traffic classification using multi-layer lstm in software defined networks. *Applied Sciences*, 9(12):2550, 2019.

[3] T. T. T. Nguyen and G. Armitage. A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys Tutorials*, 10(4):56–76, 2008.

[4] Larissa Ferreira Rodrigues, Murilo Coelho Naldi, and João Fernando Mari. Comparing convolutional neural networks and preprocessing techniques for hep-2 cell classification in immunofluorescence images. *Computers in Biology and Medicine*, 116:103542, 2020.

[5] Yukiko Nagao, Mika Sakamoto, Takumi Chinen, Yasushi Okada, and Daisuke Takao. Robust classification of cell cycle phase and biological feature extraction by image-based deep learning. *Molecular Biology of the Cell*, 31(13):1346–1354, 2020. PMID: 32320349.

[6] Keiller Nogueira, Otávio A.B. Penatti, and Jefersson A. [dos Santos]. Towards better exploiting convolutional neural networks for remote sensing scene classification. *Pattern Recognition*, 61:539 – 556, 2017.

[7] Yanming Guo, Yu Liu, Ard Oerlemans, Songyang Lao, Song Wu, and Michael S. Lew. Deep learning for visual understanding: A review. *Neurocomputing*, 187:27 – 48, 2016. Recent Developments on Deep Big Vision.

[8] M. A. Ponti, L. S. F. Ribeiro, T. S. Nazare, T. Bui, and J. Collomosse. Everything you wanted to know about deep learning for computer vision but were afraid to ask. In *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, pages 17–41, Oct 2017.

[9] S. Potluri, A. Fasih, L. K. Vutukuru, F. A. Machot, and K. Kyamakya. Cnn based high performance computing for real time image processing on gpu. In *Proceedings of the Joint INDS'11 ISTET'11*, pages 1–7, 2011.

[10] S. Shi, Q. Wang, P. Xu, and X. Chu. Benchmarking state-of-the-art deep learning software tools. In *2016 7th International Conference on Cloud Computing and Big Data (CCBD)*, pages 99–104, 2016.

[11] Jeffrey Erman, Martin Arlitt, and Anirban Mahanti. Traffic classification using clustering algorithms. In *Proceedings of the 2006 SIGCOMM Workshop on Mining Network Data*, MineNet '06, page 281–286, New York, NY, USA, 2006. Association for Computing Machinery.

[12] Danish Vasan, Mamoun Alazab, Sobia Wassan, Hamad Naeem, Babak Safaei, and Qin Zheng. Imcfn: Image-based malware classification using fine-tuned convolutional neural network architecture. *Computer Networks*, 171:107138, 2020.

[13] Z. Chen, K. He, J. Li, and Y. Geng. Seq2img: A sequence-to-image based approach towards ip traffic classification using convolutional neural networks. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1271–1276, 2017.

[14] S. Rezaei and X. Liu. Deep learning for encrypted traffic classification: An overview. *IEEE Communications Magazine*, 57(5):76–81, 2019.

[15] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret. Network traffic classifier with convolutional and recurrent neural networks for internet of things. *IEEE Access*, 5:18042–18050, 2017.

[16] F. Al-Obaidy, S. Momtahen, M. F. Hossain, and F. Mohammadi. Encrypted traffic classification based ml for identifying different social media applications. In *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, pages 1–5, 2019.

[17] Murat Soysal and Ece Guran Schmidt. Machine learning algorithms for accurate flow-based network traffic classification: Evaluation and comparison. *Performance Evaluation*, 67(6):451 – 467, 2010.

[18] Rodrigo Moreira, Pedro Frosi Rosa, Rui Luis Andrade Aguiar, and Flávio de Oliveira Silva. Enabling multi-domain and end-to-end slice orchestration for virtualization everything functions (vxfs). In Leonard Barolli, Flora Amato, Francesco Moscato, Tomoya Enokido, and Makoto Takizawa, editors, *Advanced Information Networking and Applications*, pages 830–844, Cham, 2020. Springer International Publishing.

[19] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. Sadeghi, and S. Tarkoma. Iot sentinel: Automated device-type identification for security enforcement in iot. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 2177–2184, 2017.

[20] Valentín Carela-Español, Tomasz Bujlow, and Pere Barlet-Ros. Is our ground-truth for traffic classification reliable? In Michalis Faloutsos and Aleksandar Kuzmanovic, editors, *Passive and Active Measurement*, pages 98–108, Cham, 2014. Springer International Publishing.

[21] T. Shapira and Y. Shavitt. Flowpic: Encrypted internet traffic classification is as easy as image recognition. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 680–687, 2019.

[22] L. Xu, X. Zhou, Y. Ren, and Y. Qin. A traffic classification method based on packet transport layer payload by ensemble learning. In *2019 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6, 2019.

[23] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[26] Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and ¡0.5mb model size, 2016.

[27] J. Deng, W. Dong, R. Socher, L. J. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.

[28] Pierre A. Devijver and Josef Kittler. *Pattern Recognition: A Statistical Approach.* Prentice-Hall, 1982.

[29] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2Nd Edition)*. Wiley-Interscience, New York, NY, USA, 2000.

[30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc., 2019.

[31] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov 1998.