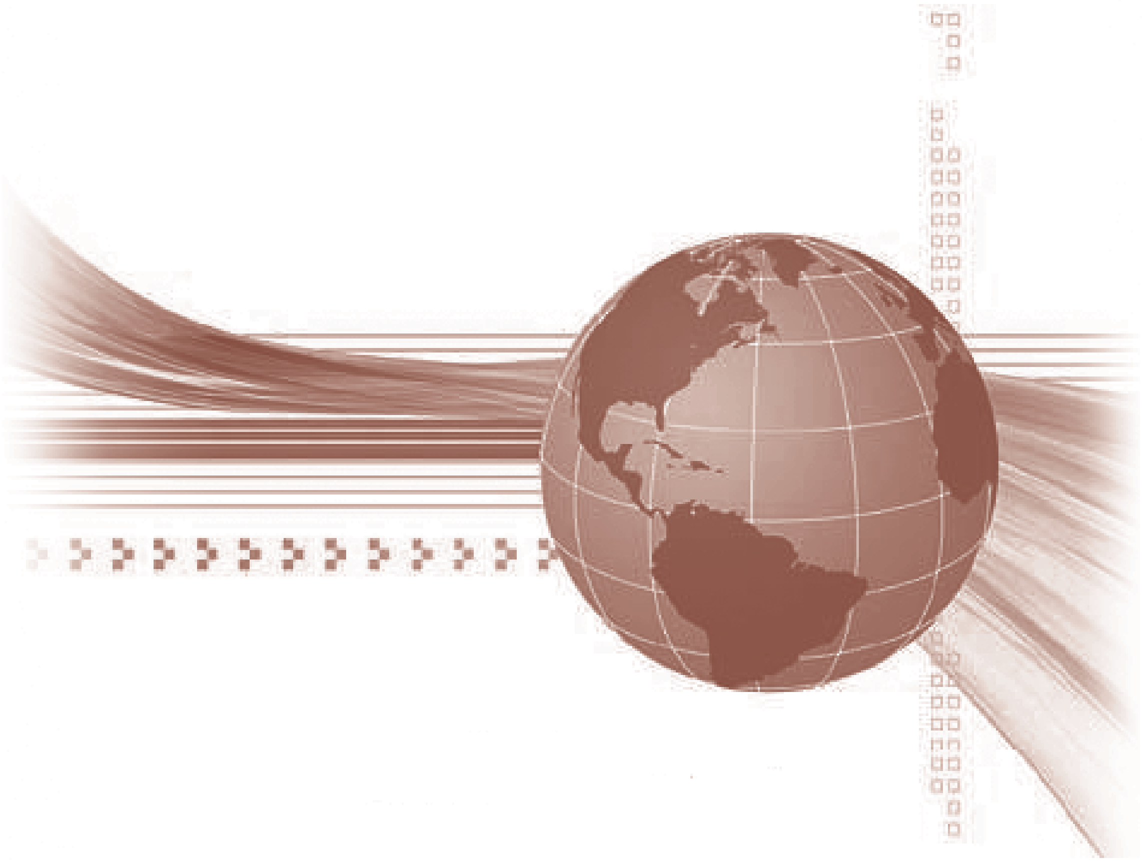
STUDIA UNIVERSITATIS
BABEŞ-BOLYAI



INFORMATICA

2/2021

STUDIA

UNIVERSITATIS BABEŞ-BOLYAI
INFORMATICA

No. 2/2021

July - December

EDITORIAL BOARD

YEAR Volume 66 (LXVI) 2021

MONTH **DECEMBER**

ISSUE **2**

STUDIA
UNIVERSITATIS BABEŞ-BOLYAI
INFORMATICA

2

EDITORIAL OFFICE: M. Kogălniceanu 1 • 400084 Cluj-Napoca • Tel: 0264.405300

SUMAR – CONTENTS – SOMMAIRE

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXVI**, Number 2, 2021
DOI: 10.24193/subbi.2021.2.01

COMPARISON OF GRADIENT-BASED EDGE DETECTORS APPLIED ON MAMMOGRAMS

CRISTIANA MOROZ-DUBENCO

Abstract. Breast cancer is one of the most common types of cancer amongst women, but it is also one of the most frequently cured cancers. Because of this, early detection is crucial, and this can be done through mammography screening. With the increasing need of an automated interpretation system, a lot of methods have been proposed so far and, regardless of the algorithms, they all share a step: pre-processing. That is, identifying the image orientation, detecting the breast and eliminating irrelevant parts.

This paper aims to describe, analyze, compare and evaluate six of the most commonly used edge detection operators: Sobel, Roberts Cross, Prewitt, Farid and Simoncelli, Scharr and Canny. We detail the algorithms, their implementations and the metrics used for evaluation and continue by comparing the operators both visually and numerically, finally concluding that Canny best suit our needs.

1. Introduction

According to Bray et. al [2], in 2018 breast cancer was the second leading type of cancer that caused death amongst women worldwide. It is also the most frequently diagnosed type of cancer when it comes to women, with a percentage of 25% of all types of cancer. Moreover, about 10% of women develop breast cancer [13].

Fortunately, early detection along with proper treatment can lead to curing. Mammography is one of the most used screening methods, which proved to be very effective and reliable in detecting cancer in early stages. Given the fact that it is recommended for mammographies to be performed at regular intervals and keeping in mind that mammogram interpretation is both difficult and time-consuming, the need for an automated interpretation system becomes imminent.

Received by the editors: 6 June 2021.

2010 *Mathematics Subject Classification.* 68U10.

1998 *CR Categories and Descriptors.* I.4.6 [**Image Processing and Computer Vision**]: Segmentation – *Edge and feature detection.*

Key words and phrases. Image Processing, Computer Vision, Edge Detection.

Although there are many different methods for mammography interpretation proposed in literature, most of them follow the same steps, as stated in Es-salhi et al. [6], Ramani et al. [18], Duque et al. [5], Sharma and Sharma [20] and Desai et al. [4], to name just a few:

(1) *pre-processing:* identification of the image orientation, detection of the breast, elimination of the background and detection and elimination of the pectoral muscle in case of medio-lateral orientation;
(2) *segmentation:* detection of possible lesion;
(3) *classification:* labeling the segmented area as either benign or malignant.

In this paper we are focusing on the first step and, more exactly, on breast detection. In order to perform an accurate segmentation, only the breast should be analyzed, whilst the black background and, especially, the artifacts, which are high-intensity areas that can affect the quality of the segmentation process, should be removed. In order to achieve this goal, we compared the performance of various existing edge detection operators when applied to mammograms - that is, we described each operator, analyzed their outputs, compared them both to the original images and to one another and applied a few well-known image quality assessment metrics on these results.

Whilst many comparative studies regarding the performance of popular edge detection techniques have been conducted so far, by the time we conducted this study we could not find any paper that compares the techniques when applied strictly to mammographic images which would either back the results by numerical values (namely, by computing image quality assessment metrics on the outputs) or apply those techniques on a large dataset.

In order to achieve these objectives, we applied existing edge detection methods on the mini-MIAS database [22].

2. Scientific Problem

Edge detection is a fundamental tool in image processing, used as a preprocessing step to feature detection, feature extraction and image segmentation. Usually, edges outline object boundaries, boundaries between objects and the background of the image, therefore allowing the extraction of the region of interest for further processing. In our case, we expect that the edge detection operation correctly and completely identifies the boundary between the breast and the background.

Mammographies are obtained by using a low-dose x-ray system and have some special features, which can lead to a rather difficult process of edge detection:

- they are grey-scale images;
- they can contain weak boundaries;

- they exhibit Gaussian noise;
- they can contain a different type of background noise: artifacts, such as medical labels;
- they can have low quality, low contrast and poor illumination, depending on the machine used.

Edge detection aims to identify points at which the brightness is slightly different. These points are organized into a set of curved line segments, namely edges. Rosenfeld and Kak [19] defined edges as abrupt changes in gray level or texture at the intersection of two different regions, while Park and Murphey [16] differentiated between edge points and edge fragments, stating that edge points are pixels where the local intensity changes significantly and edge fragments represent the edge points along with their orientation. Therefore, edges are defined as pixels that have discontinuities in intensity. There are four different edge types:

(1) *step edges:* ideal type of edges that occur when the intensity changes significantly from one side to the other;
(2) *line edges:* edges that occur when the intensity changes significantly, remains at the new value for a number of pixels, then returns to the original value;
(3) *ramp edges:* step edges where the intensity does not change instantaneously, but the change occurs over a finite distance;
(4) *roof edges:* line edges where the intensity does not change instantaneously, but the change occurs over a finite distance.

Since edges are pixels that have abrupt changes in intensity, the derivatives of the image intensity function can be used to measure these discontinuities, either by thresholding the first derivative values or by searching for zero-crossings in the second derivative of the image function.

Edge detection techniques based on the derivatives of the image intensity functions are widely used and can be split into two categories: search based methods and zero-crossing based methods. Search based methods use a first-order derivative expression in order to compute a measure of edge strength and then search for local directional maxima of the gradient magnitude, while zero-crossing based methods search for zero crossings in a second-order derivative expression.

Following, we are going to analyze several search based edge detection techniques. These methods, also called gradient-based methods, detect the edges by computing the gradient of local intensity at each point in the image and associate the local peaks in the first derivative with edges in the original image.

Considering $f(x, y)$ an image with (x, y) denoting the coordinates of a point, the two-dimensional gradient is a vector with two elements:

$$G = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix}$$

where G_x and G_y are measures for changes in pixel values in the horizontal and vertical directions, respectively.

Usually, gradient-based edge detection operators follow three steps:

(1) *smoothing:* pre-processing step for reducing the noise;
(2) *differentiation:* convolving the image with the two masks, G_x and G_y, for computing the local gradient;
(3) *detection:* detecting edge points based on the local gradients.

In this paper, we are focusing on six well-known gradient-based edge detectors: Sobel [23], Prewitt [1], Roberts Cross [10], Scharr [21], Farid and Simoncelli [7] and Canny [3].

3. Related Work

In Ramani et al. [18], four filters, namely mean filter, median filter, adaptive median filter and wiener filter, are both visually and numerically compared on mammographies. The filters are applied on three images from the mini-MIAS database [22]. On each image, three different types of noise are applied: salt & pepper noise, speckle noise and Gaussian noise. Afterwards, the images are reconstructed using the above-mentioned filters and the results are compared in terms of Mean Squared Error, Peak Signal to Noise Ratio, Structural Content and Normalized Absolute Error, concluding that the adaptive median filter yields the best results. Moreover, the numerical results are backed by the visual results obtained for one of the three images used for evaluation.

In paper [10], Maitra et al. propose a two-phase edge detection algorithm for breast detection in mammographies: homogenizing the image and detecting the edges. For the first phase, a novel method, which adjusts the intensity of the image in order to obtain a normalized intensity level, is presented. For the second phase, the input image is scanned in both horizontal and vertical direction and two edge maps are constructed, which are then merged to obtain the edge map of the image. The results obtained for five mammograms from the mini-MIAS database are visually compared to the results obtained with classical edge detection operators, namely Roberts Cross, Prewitt, Sobel, Kirsch and Laplacian of Gaussian, concluding that the proposed method produces the best results.

In Mirzaalian et al. [15], a new algorithm for breast contour detection is described, tested and compared to two other existing methodologies, presented by Ferrari et al. [8] and Wirth [25]. The proposed method is composed of multiple steps: normalizing the mammograms by histogram equalization, convolving with a mask, removing small noises through morphological operations, removing larger noises through labeling and convolving the top of the mammogram with a mask to overcome the problem of inaccurate border detection due to low contrast between the breast region and the background. When comparing the proposed method with the ones proposed by Ferrari et al. and Wirth for 20 mammograms in terms of Haudorff Distance Measure and Mean of Absolute Error Distance Measure, the proposed method outperformed the other two methods.

On the other hand, a number of comparative studies of edge detectors have been conducted so far. Harun et al. [9] compare five methods - Sobel, Canny, Roberts Cross, Canny and Sobel combination and Canny and Roberts Cross combination - in order to determine the vessel wall elasticity, by applying to operators on ten B-mode images and computing Mean Squared Error and Peak Signal to Noise Ratio on the output image, leading to the conclusion that the combination between the Canny and the Roberts Cross operators yields the most satisfactory results.

Kumar et al. [12] present a comparison of Prewitt, Sobel, Roberts Cross, Canny, Laplacian of Gaussian and Zero Crossing applied on three biometric images: the image of an iris, the image of a thumbprint and the image of a face. The numerical results are obtained by computing Mean Squared Error and Peak Signal to Noise Ratio, using, for each operator, the outputs of the other operators as ground truth. This paper concludes that Canny is the best fitted edge detection operator for biometric images.

Poobathy and Chezian [17] compare the performance of Canny, Sobel, Laplacian of Gaussian, Roberts Cross and Prewitt operators, by applying them on set of four universally standardized test images and computing Mean Squared Error and Peak Signal to Noise Ratio against the ground truth image. The authors achieve the conclusion that the Canny operator outperforms the other ones.

4. Proposed Approach

In order to compare the edge detection operators, we used a simple program that loads an image, pre-proccesses it and then applies the operator. For smoothing the image, we used the *GaussianBlur* method from the *opencv-python* library, while for edge detection, we used the *scikit-image* library in Python.

To evaluate the performances of the presented edge detection operators, we first visually compared their results, using three test images. For this goal, we applied each of the operators on the selected images and compared their outputs both to the original images and to one another. The resulting images are to be presented in the following section.

Secondly, we applied three quality assessment metrics on the entire dataset, as follows:

(1) *Mean Squared Error*

The Mean Squared Error (MSE) [12] is a measure for the average squared difference between the estimated values and the actual values - that is, the square difference between the compared images. It is a risk function which corresponds to the expected value of the squared error loss.

The mean squared error between two images can be expressed as:

$$MSE = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}(I_1(x,y) - I_2(x,y))^2$$

where M and N are the number of rows and columns respectively in the input images (which need to be equal in order to obtain valid results) and $I_1(x,y)$ and $I_2(x,y)$ represent the value of the pixel having the coordinates (x,y), in each image respectively.

The MSE is always equal or greater than zero, with better values being closer to zero.

(2) *Peak Signal to Noise Ratio*

The Peak Signal to Noise Ratio (PSNR) [11] indicates the level of losses or signals integrity. It is measured in decibels and is frequently used to measure the quality of compressed images in comparison to the original ones, with higher PSNR values representing a better quality of the modified image.

The peak signal to noise ratio is computed using the formula:

$$PSNR = 10log_{10}(\frac{R^2}{MSE})$$

where R is the dynamic range of pixel values in the input image (which is 255 for gray level images where pixel values are represented as 8-bit integers) and MSE is the mean squared error between the two images.

(3) *Structural Similarity Index Measure*

The Structural Similarity Index Measure (SSIM) [24] is more related to the human visual system, extracting information as contrast, structure and luminance. It aims to address the limitations of the MSE in terms of perceived similarity by taking texture into account.

The SSIM index is computed on multiple windows of an image. For two windows p and q of size SxS, we have the following comparison functions, for luminance, contrast and structure:

$$l(p,q) = \frac{2\mu_p\mu_q + C_1}{\mu_p^2 + \mu_q^2 + C_1}, \quad c(p,q) = \frac{2\sigma_p\sigma_q + C_2}{\sigma_p^2 + \sigma_q^2 + C_2}, \quad s(p,q) = \frac{\sigma_{pq} + C_3}{\sigma_p\sigma_q + C_3}$$

where μ_p and μ_q are the averages of p and q respectively:

$$\mu_p = \frac{1}{S}\sum_{i=1}^{N} p_i \quad \text{and} \quad \mu_q = \frac{1}{S}\sum_{i=1}^{N} q_i$$

σ_p and σ_q are the variances of p and q respectively:

$$\sigma_p = \left(\frac{1}{S-1}\sum_{i=1}^{S}(p_i - \mu_p)^2\right)^{\frac{1}{2}} \quad \text{and} \quad \sigma_q = \left(\frac{1}{S-1}\sum_{i=1}^{S}(q_i - \mu_q)^2\right)^{\frac{1}{2}}$$

σ_{pq} is the covariance of p and q:

$$\sigma_{pq} = \frac{1}{S-1}\sum_{i=1}^{S}(p_i - \mu_p)(q_i - \mu_q)$$

and C_1, C_2 and C_3 are constants, defined as follows:

$$C_1 = (K_1R)^2, \quad \text{where} \quad K_1 << 1 \quad \text{is a small constant}$$

$$C_2 = (K_2R)^2, \quad \text{where} \quad K_2 << 1 \quad \text{is a small constant}$$

$$C_3 = \frac{C_2}{2}$$

By combining the three comparison functions presented above, we obtain the formula for the structural similarity index measure:

$$SSIM(p,q) = [l(p,q)]^\alpha \cdot [c(p,q)]^\beta \cdot [s(p,q)]^\gamma$$

with l, c and s denoting the luminance, contrast and structure respectively, and $\alpha > 0$, $\beta > 0$ and $\gamma > 0$ being the parameters used for adjusting the relative importance of the components. By setting $\alpha = \beta = \gamma = 1$, we obtain a specific form of the index, which we are going to use in this paper:

$$SSIM(p,q) = \frac{(2\mu_p\mu_q + C_1)(2\sigma_{pq} + C_2)}{(\mu_p^2 + \mu_q^2 + C_1)(\sigma_p^2 + \sigma_q^2 + C_2)}$$

In order to measure the quality of the output image from the edge detection operation as a whole, rather than the quality of particular windows of the image, we are going to use a mean structural similarity index measure:

$$MMSIM(P, Q) = \frac{1}{W} \sum_{i=1}^{M} SSIM(p_i, q_i)$$

where P is the original image, Q is the image obtained after applying an edge detection operator, W is the number of windows in the image and p_i and q_i are the image contents at the i-th window.

Because of the fact that the dataset we used for experiments contains neither ground truth images for reference, nor any relevant information regarding the breast contour, we followed the methodology presented by Poobathy and Chezian [17], Mat Harun et. al [14] and Kumar et. al [12] and compared the results of the edge detection operators to the original image and to the outputs of the other operators. That is, we applied each metric on the output of each operator, first against the original image and afterwards against the outputs of the other operators. To get a numerical value for the entire dataset, we computed the average of the results of each individual metric for each image.

Normally, when comparing a processed image to the ground truth, MSE values closer to 0 indicate a better result. However, in our case, when using the original image as ground truth, as stated in Poobathy and Chezian [17], we are interested in higher values for MSE. MSE values closer to zero indicate an output very similar to the original image, which means that the operator was not able to properly detect the edges. Since we aim to determine which operator best detects the contour, we are looking for the output image that is the most different than the original one. Thus, higher values for MSE mean an output containing almost only the contour of the breast, with both the background and the inside of the breast being considered errors.

On the other hand, for PSNR we are looking for values closer to zero. Following the same logic as for MSE and taking into consideration the fact that the peak signal to noise ratio represents the measure of the peak error, the operators yielding lower results for PSNR better detect the contour.

Finally, we are using MSSIM to compute the similarity level between the outputs of the edge detectors and the original images. Because MSSIM is used to determine the structural similarity, we aim for values closer to zero, which would indicate that, from a structural point of view, the output is no longer similar to the original image.

When comparing the operators to one another, according to Kumar et. al [12], higher value of dissimilarities between one operator and the others indicate a better performing edge detection operator. In order to properly compare the operators, we consider the average value for each of the metrics for every operator and look for lower PSNR and MSSIM values and higher MSE values.

For analyzing and comparing the performance of the edge detection operators, we use the mini-MIAS database [22], provided by the *Mammographic Image Analysis Society*, which contains 322 mammograms, digitized and reduced to 200 micro pixel edge, at a size of 1024x1024 pixels. The mammograms can be divided into three main categories: normal, containing benign abnormality and containing malignant abnormality. For visually comparing existing edge detection techniques we use three mammograms from the mini-MIAS database, one from each category, chosen randomly, where:

- *mdb014* does not contain abnormalities;
- *mdb080* contains a well-defined/circumscribed benign abnormality centered at (432, 149) coordinates with a radius of 20 pixels;
- *mdb184* contains a spiculated malignant abnormality centered at (352, 624) coordinates with a radius of 114 pixels.

For numerical evaluation of the operators, we use all the images in the database.

5. Experimental Results

5.1. **Visual Results.** The results of applying the operators onto the original images are shown in Figure 1 as follows:

- the first column contains the original images,
- the second column contains the result obtained for the Sobel operator,
- the third column contains the result obtained for the Prewitt operator,
- the fourth contains the result obtained for the Roberts Cross operator,
- the fifth column contains the result obtained for the Scharr operator,
- the sixth column contains the result obtained for the Farid and Simoncelli operator,
- the seventh contains the result obtained for the Canny operator, with the threshold values chosen experimentally to $t_{low} = 3$ and $t_{high} = 10$.

For a better visualization of the results, we converted the edge detection results into binary images, where all the pixels detected as edges, regardless of their intensity, are shown in white, and the background is shown in black. The binary results are presented in Figure 2.

Given these visual comparisons, we consider that the best results were obtained using the Canny filter, although the Sobel and Roberts filters also yielded good results in terms of breast detection, but their results contain much more noise than Canny's.

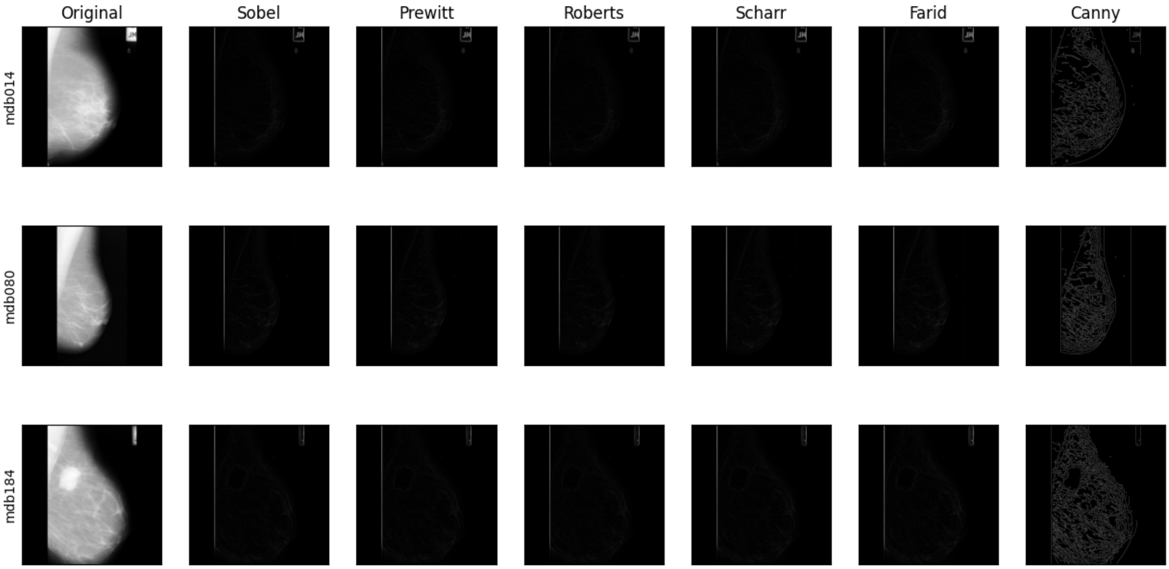


Figure 1. Comparison of edge detectors

5.2. **Numerical Results.** For comparing the operators numerically, we applied each operator on every image of the mini-MIAS dataset, then applied metrics as follows:

(1) Use the original image as ground truth and compute MSE, PSNM, MSSME for every image against the output image of every operator;
(2) Use each operator's output image as ground truth and compute MSE, PSNM, MSSME for every image against the output image of every operator.

The results presented in Table 1, Table 2, Table 3 and Table 4 represent the average of the results obtained across all 322 images in the mini-MIAS dataset.

Table 1 shows that the MSE values for all the operators are very close and, at the same time, very high. That means that the operators detected the edges and none of them produced an output close to the original image. The highest MSE value was obtained for the Canny operator, with a difference of approximately 0.6 from Farid, which produced the second highest value. The same table presents also the values obtained for PSNR, with Canny yielding

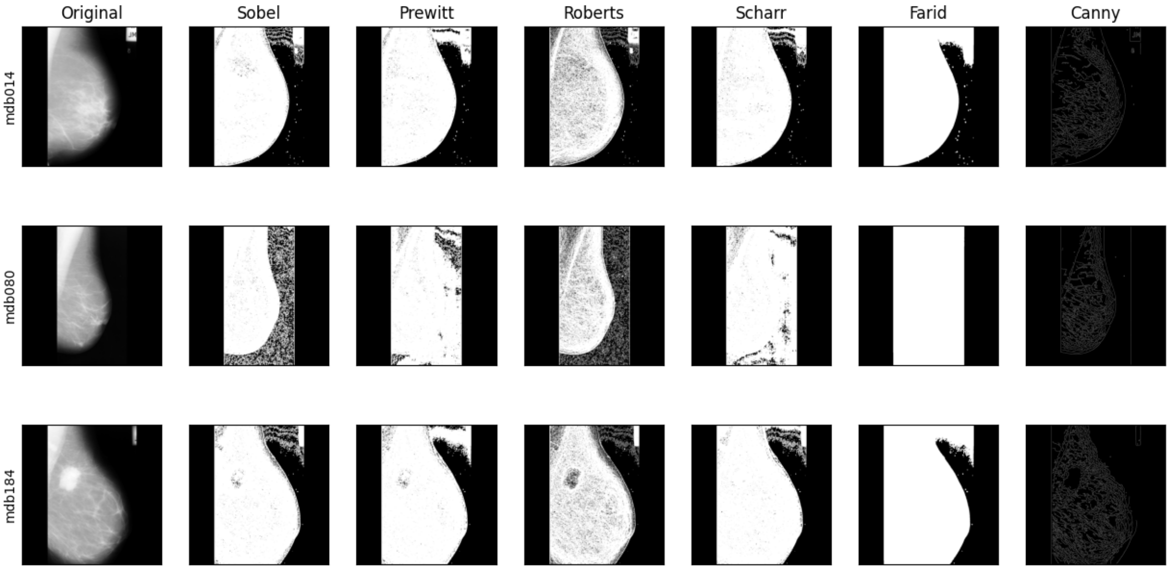


Figure 2. Comparison of edge detectors

	MSE	PSNR	MSSMI
Sobel	8417.05056	9.17966	0.51232
Prewitt	8417.07461	9.17965	0.51232
Roberts	8417.16912	9.17960	0.512333
Scharr	8417.02115	9.17968	0.51223
Farid	8417.83910	9.17952	0.51223
Canny	8418.43014	9.17893	0.51275

Table 1. MSE, PSNR and MSSMI computed using the original image as ground truth

the lowest value, followed by Scharr, Sobel and Prewitt. As for MSSIM, all the operators produced similar values of around 0.5, meaning that the structure of the mammography has somehow changed. In this case as well, Canny raised the best result.

MSE	Sobel	Roberts	Prewitt	Farid	Scharr	Canny
Sobel	0	4.74E-05	3.61E-08	0.00010	2.08E-08	0.03303
Roberts	4.74E-05	0	4.72E-05	4.00E-05	4.76E-05	0.03338
Prewitt	3.61E-08	4.72E-05	0	0.00010	1.11E-07	0.03303
Farid	0.00010	4.00E-05	0.00010	0	0.00010	0.03373
Scharr	2.08E-08	4.76E-05	1.11E-07	0.00010	0	0.03302
Canny	0.03303	0.03338	0.03303	0.03373	0.03302	0

Table 2. MSE computed using the operators' outputs as ground truth

PSNR	Sobel	Roberts	Prewitt	Farid	Scharr	Canny
Sobel	inf	43.64106	74.64643	40.22099	77.03130	15.01364
Roberts	43.641068	inf	43.65951	44.36988	43.62168	14.96674
Prewitt	74.64643	43.65951	inf	40.24118	69.74624	15.01292
Farid	40.22099	44.36988	40.24118	inf	40.20274	14.92039
Scharr	77.03130	43.62168	69.74624	40.20274	inf	15.01421
Canny	15.01364	14.96674	15.01292	14.92039	15.01421	inf

Table 3. PSNR computed using the operators' outputs as ground truth

MSSIM	Sobel	Roberts	Prewitt	Farid	Scharr	Canny
Sobel	1	0.99341	0.99998	0.97973	0.99999	0.73985
Roberts	0.99341	1	0.99355	0.99199	0.99328	0.74337
Prewitt	0.99998	0.99355	1	0.98006	0.99996	0.73999
Farid	0.97973	0.99199	0.98006	1	0.97943	0.74648
Scharr	0.99999	0.99328	0.99996	0.97943	1	0.739703
Canny	0.71237	0.72285	0.71283	0.73521	0.71190	1

Table 4. MSSIM computed using the operators' outputs as ground truth

The results of computing MSE using each of the operator's outputs as ground truth are presented in Table 2. It is easily detectable that Canny produced the higher MSE values when computed using any other operator's output as reference. Also, by looking at the row for Canny in Table 3, one can tell that there are the lowest PSNR values. From table 4, we can tell that Sobel and Scharr produced almost identical results, while Canny's results are the most different from a structural point of view.

6. Conclusion and future work

Suming up the results presented in the previous section, we can conclude that the Canny operator yields the best results. However, it is worth mentioning that the Farid and Simoncelli and the Scharr operators also yielded satisfactory numerical results, close to the ones obtained by Canny.

We consider that we reached our goal. We described, analyzed, compared and evaluated six edge detection operators - namely, Sobel, Roberts Cross, Prewitt, Farid and Simoncelli, Scharr and Canny -, providing useful visual and numerical comparison results.

As future work, we intend to compare the operators from a qualitative point of view as well, with the help of a radiologist, in order to back up the numerical results provided in this paper.

7. Acknowledgments

This work was supported by a grant of the Romanian Ministry of Education and Research, CCCDI - UEFISCDI, project number PN-III-P2-2.1-PED-2019-2607, within PNCDI III.

References

[1] Ahmed, A. S. Comparative study among sobel, prewitt and canny edge detection operators used in image processing. *J. Theor. Appl. Inf. Technol 96*, 19 (2018), 6517–6525.

[2] Bray, F., Ferlay, J., Soerjomataram, I., Siegel, R. L., Torre, L. A., and Jemal, A. Global cancer statistics 2018: Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA: a cancer journal for clinicians 68*, 6 (2018), 394–424.

[3] Canny, J. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, 6 (1986), 679–698.

[4] Desai, S. D., Megha, G., Avinash, B., Sudhanva, K., Rasiya, S., and Linganagouda, K. Detection of microcalcification in digital mammograms by improved-mmgw segmentation algorithm. In *2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies* (2013), IEEE, pp. 213–218.

[5] Duque, A. E. R., Gómez, D. C. A., and Nieto, J. K. A. Breast lesions detection in digital mammography: An automated pre-diagnosis. In *2014 XIX Symposium on Image, Signal Processing and Artificial Vision* (2014), IEEE, pp. 1–5.

[6] Es-salhi, R., Daoudi, I., Tallal, S., Medromi, H., et al. A survey on segmentation techniques of mammogram images. In *International Symposium on Ubiquitous Networking* (2016), Springer, pp. 545–556.

[7] Farid, H., and Simoncelli, E. P. Optimally rotation-equivariant directional derivative kernels. In *International Conference on Computer Analysis of Images and Patterns* (1997), Springer, pp. 207–214.

[8] Ferrari, R., Frere, A., Rangayyan, R., Desautels, J., and Borges, R. Identification of the breast boundary in mammograms using active contour models. *Medical and Biological Engineering and Computing 42*, 2 (2004), 201–208.

[9] Harun, M., Izzah, M., Ibrahim, N., and Aziz, N. S. Comparative study of edge detection algorithm: vessel wall elasticity measurement for deep vein thrombosis diagnosis. *ARPN Journal of Engineering and Applied Sciences 10*, 19 (2015), 8635–8641.

[10] Indra Kanta Maitra, Sanjay Nag, S. K. B. A novel edge detection algorithm for digital mammogram. *International Journal of Information and Communication Technology Research* (2012).

[11] Jose, A., Dixon, K. D. M., Joseph, N., George, E. S., and Anjitha, V. Performance study of edge detection operators. In *2014 International Conference on Embedded Systems (ICES)* (2014), IEEE, pp. 7–11.

[12] Kumar, S., Singh, M., and Shaw, D. Comparative analysis of various edge detection techniques in biometric application. *International Journal of Engineering and Technology (IJET) 8*, 6 (2016), 2452–2459.

[13] Lau, T.-K., and Bischof, W. F. Automated detection of breast tumors using the asymmetry approach. *Computers and biomedical research 24*, 3 (1991), 273–295.

[14] Mat Harun, N. H., Ibrahim, N., and Aziz, N. S. Comparative study of edge detection algorithm: vessel wall elasticity measurement for deep vein thrombosis diagnosis.

[15] Mirzaalian, H., Ahmadzadeh, M. R., Sadri, S., and Jafari, M. Pre-processing algorithms on digital mammograms. In *MVA* (2007), pp. 118–121.

[16] Park, J., and Murphey, Y. *Edge Detection in Grayscale, Color, and Range Images.* 04 2008.

[17] Poobathy, D., and Chezian, R. M. Edge detection operators: Peak signal to noise ratio based comparison. *IJ Image, Graphics and Signal Processing 10* (2014), 55–61.

[18] Ramani, R., Vanitha, N. S., and Valarmathy, S. The pre-processing techniques for breast cancer detection in mammography images. *International Journal of Image, Graphics and Signal Processing 5*, 5 (2013), 47.

[19] Rosenfeld, A., and Kak, A. Digital picture processing academic press. *New York* (1982), 242.

[20] Sharma, J., and Sharma, S. Mammogram image segmentation using watershed. *Int J Info Tech and Knowledge Management 4* (2011), 423–5.

[21] Sian, C., Jiye, W., Ru, Z., and Lizhi, Z. Cattle identification using muzzle print images based on feature fusion. In *IOP Conference Series: Materials Science and Engineering* (2020), vol. 853, IOP Publishing, p. 012051.

[22] SUCKLING J, P. The mammographic image analysis society digital mammogram database. *Digital Mammo* (1994), 375–386.

[23] Vairalkar, M. K., and Nimbhorkar, S. Edge detection of images using sobel operator. *International Journal of Emerging Technology and Advanced Engineering 2*, 1 (2012), 291–293.

[24] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing 13*, 4 (2004), 600–612.

[25] Wirth, M. A. *A nonrigid approach to medical image registration: matching images of the breast.* Royal Melbourne Institute of Technology, 1999.

Department of Computer Science, Faculty of Mathematics and Computer Science, Babeş-Bolyai University, Mihail Kogălniceanu 1, 400084, Cluj-Napoca, Romania

Email address: cristiana.moroz@ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXVI**, Number 2, 2021
DOI: 10.24193/subbi.2021.2.02

TOWARDS A SUPPORT SYSTEM FOR DIGITAL MAMMOGRAM CLASSIFICATION

ADÉL BAJCSI

Abstract. Cancer is the illness of the 21$^{\text{th}}$ century. With the development of technology some of these lesions became curable, if they are in an early stage. Researchers involved with image processing started to conduct experiments in the field of medical imaging, which contributed to the appearance of systems that can detect and/or diagnose illnesses in an early stage. This paper's aim is to create a similar system to help the detection of breast cancer. First, the region of interest is defined using filtering and two methods, Seeded Region Growing and Sliding Window Algorithm, to remove the pectoral muscle. The region of interest is segmented using k-means and further used together with the original image. Gray-Level Run-Length Matrix features (in four direction) are extracted from the image pairs. To filter the important features from resulting set Principal Component Analysis and a genetic algorithm based feature selection is used. For classification K-Nearest Neighbor, Support Vector Machine and Decision Tree classifiers are experimented. To train and test the system images of Mammographic Image Analysis Society are used. The best performance is achieved features for directions $\{45°, 90°, 135°\}$, applying GA feature selection and DT classification (with a maximum depth of 30). This paper presents a comprehensive analysis of the different combinations of the algorithms mentioned above, where the best performence repored is 100% and 59.2% to train and test accuracies respectively.

Received by the editors: 22 June 2021.

2010 *Mathematics Subject Classification.* 68T35.

1998 *CR Categories and Descriptors.* I.2.1 [**Artifical Intelligence**]: Applications and Expert Systems – *Medicine and science*; I.2.6 [**Artifical Intelligence**]: Learning – *Knowledge acquisition*; I.4.6 [**Image Processing and Computer Vision**]: Segmentation – *Pixel classification*; I.4.7 [**Image Processing and Computer Vision**]: Feature Measurement – *Feature representation*;

Key words and phrases. region growing, k-means, GLRLM feature extraction, GA feature selection, PCA, mammogram, classification, Decision Tree classification, Random Forest classification, MIAS.

1. Introduction

Computer-aided diagnosis (CADx) and -detection (CADe) systems are frequently researched by the scientists to help doctors and to give patients higher chance to recover from their illness. Based on a report by the European Cancer Information System [1] in 2020, amongst females breast cancer causes the the most deaths. To diagnose the abnormality of the breast tissue often digital mammograms are used. Creating a system to process mammograms and to decide whether it is normal or abnormal (also deciding if the lesion is benign or malignant) can be divided into two blocks: (1) including preprocessing and segmentation and the (2) classification block (including feature extraction, -selection and classification).

The focus of the current paper is on constructing a CADx system by implementing each of the steps mentioned above. As input data MLO (medio-lateral oblique) view mammograms are used. For the first two steps we propose the use of unsupervised learning methods (filters, k-means), then use the result of this first block (the segmented image) as input, together with the original image to the second block. For feature extraction we propose the use of Gray-Level Run-Length Matrices (GLRLM), for feature selection Principal Component Analysis (PCA) and a genetic feature selection method (GA) and for classification K-Nearest Neighbors (KNN), Support Vector Machines (SVM) and Decision Tree (DT) classifiers are used. The novelty of the research consists in the classification approach (using one classifier to separate normal, benign and malignant mammogram images) and also in the combination of methods (k-means, GLRLM, PCA/GA, KNN/SVM/DT).

This paper is arranged as follows. In section 2 methods from the literature are presented to the steps mentioned previously. The details of the current experiment are presented in section 3 followed by the numerical results in section 4. Section 5 includes the conclusions of the experiment and the future work. Our acknowledgement is expressed in the last section.

2. Related work

CADe systems for cancerous cells are essential, because with an early detection there is a higher chance for the patients survival. Thus, methods for the steps mentioned in section 1 have been widely investigated in the literature. In the following paragraphs solutions from recent studies are presented.

2.1. **Preprocessing.** In case of digital mammogram analysis consists of the definition of the ROI, which is the region of the breast without the pectoral muscle. First, the foreground (labels, breast region, other artifacts) and the

[1] `https://ecis.jrc.ec.europa.eu/explorer.php`

background has to be separated. Second, border between the pectoral muscle and the breast has to be defined.

2.1.1. *Remove label and image artifacts.* The solution for the first step in the literature is implemented using thresholding, which is one of the simplest and easiest methods proposed to segment the image into foreground and background. In some studies [25, 12, 22] simple binary thresholding is used while others [15, 24, 19, 6, 18] use Otsu's thresholding with filters to prevent removing less dense tissues. In the previous researches Rahimeto et al. [15] used Wiener filter [20] while Salam et al. [19] used median filtering [20].

2.1.2. *Remove pectoral muscle.* The pectoral muscle is part of the breast but its intensity and density is similar to the abnormal tissues. Therefore, it can cause misclassifications. Hence, we want to remove the achieve better results.

A large number of existing studies in the broader literature have examined the use of segmentation methods to remove the pectoral muscle from mammograms. In some studies unsupervised methods are used like region growing [12, 6, 24, 18], thresholding [15, 25, 24] and k-means [24]. Other used supervised methods [24].

Maitra et al. [12] focuses on a method to remove the pectoral muscle based on a seeded region growing algorithm where the seeds are selected from a line from the pectoral muscle. Compared to Maitra et al. [12], Esener et al. [6] used only a single seed. To enhance the results of the proposed method they applied a straight-line approximation to define the boundary of the pectoral muscle.

Rahimeto et al. [15] used multilevel (Otsu's) thresholding to segment the breast's tissue into three groups: background, less dense tissue and highly dense tissue. The pectoral muscle as well as the abnormality are taking part of the highly dense segment. Then they define the pectoral muscle by measuring the perimeter and the portion of the perimeter on the edges. Finally, they propose the use of quadratic polynomial curve fitting to smooth the boundary of the muscle.

Shrivastava et al. [25] presented another unsupervised method to remove the pectoral muscle using a sliding window algorithm. While the given conditions are satisfied (minimum total intensity, maximum difference) the pixels inside the window are set to 0.

Shinde and Rao [24] proposed using Support Vector Machines (SVM) [1]. To define a possible location of the pectoral muscle they applied three segmentation methods: k-means clustering, Otsu's thresholding and region growing (the seed is selected based on the assumed location of the pectoral muscle and the region's intensity). They used Gray Level Co-occurrence Matrix (GLCM) on each result to extract texture and statistical features. These feature were fed to an SVM which decided which one defines best the pectoral muscle.

The literature review conducted by Moghbel et al. [13] lists solutions (including the ones mentioned) presented for this step from the state-of-the-art.

2.2. **Segmentation.** Haty et al. [7] used Otsu's thresholding [3] to segment the breast tissue and in the last step kNN classification was used to decide if the breast is normal or abnormal. Sadeghi et. al [18] used a two step segmentation. In the first step they created a binary mask with a global threshold (relative maximums from the histogram) from the histogram normalized image. After applying the mask got from the first step they used morphological operations to enhance the texture of the remaining area. In the second step they go through the mammography image with two windows. Based on the average and the difference between the minimum and maximum intensity in the windows respectively they segment the tissue which is probably cancerous.

A recent study by Kim et al. [9] presented the use of convolutional neural networks (CNN) for unsupervised image segmentation. They alternatively predict the labels for each pixel and optimize the network's parameters until they become spatially continuous, or similar pixels are assigned to the same label and the number of labels is the highest. Li et. al [10] presented a dual CNN which segments the image and predicts the diagnosis simultaneously. The networks are running parallel and while the first one defines the semantic features the other one defines structural features. At the end they used the structural feature to segment the tumor and a fusion of features to decide if it is malignant or benign.

2.3. **Feature extraction.** Feature extraction has a key role in a CADe system. The extracted information consist of the input of the final classification. Mammograms are gray-scale images, therefore specific feature extraction methods need to be examined.

Vijayarajeswari et al. [28] applied Hough transform to the mammogram and then calculated statistical features like mean, variance, entropy and standard deviation. Similar features, are the wavelet features proposed by Rashed and Awad [16].

Chaieb and Kalti [5] conducted a literature survey on feature extraction from mammograms. They compared five statistical features (First Order Statistics, Gray-Level Co-occurrence Matrices, Gray-Level Difference Matrices, Tamura, GLRLM features) and concluded that the best result was achieved using GLRLM features.

Arora et al. [2] proposed the use of Convolutional Neural Networks (CNNs) for feature extraction. The ROI is embedded using five networks (AlexNet, VGG16, ResNet, GoogLeNet, InceptionResNet) and the resulting features are concatenated and fed for the classification module. The advantage of using CNNs is that with the supervised manner of feature extraction mitigates the problem of class imbalance in the dataset. On the other hand, a disadvantage of the proposed method is that the ROI is defined as a minimum bounding box containing the abnormality.

2.4. **Feature selection.** The result of extraction can result in many feature. This can have an impact on the classifier's complexity. Thus, feature selection methods are used to reduce the number of features without too much information loss.

Different unsupervised feature selection methods are compared in a review by Solorio-Fernández et al. [26]. The most widely used unsupervised method for dimensionality reduction is PCA [11]. It also appears in recent studies to filter features extracted from mammograms [5].

In the literature supervised feature selection methods are also investigated. The survey by Chaieb and Kalti [5] also discuses the problem of feature selection. Tabu search, genetic algorithm (GA), ReliefF algorithm, sequential forward selection and sequential backward selection are compared as feature selection methods. They concluded that using GA selection had the best performance.

2.5. **Classification.** Classification is the final step of a CAD system. It aims to distinguish different types of breast tissues (normal/benign/malignant).

Deep Learning (DL) is an extensively researched field of computer science and in recent studies [29] it is used for mammogram classification. Wang et al. [29] compared the performance of six deep learning models: AlexNet, VGG16 and ResNet, two classifiers presented by Shen [23] (one using VGG16 and another with ResNet) and an instance-based learning method using r-CNN. They concluded that the CNN classifiers have a good performance on the training data, but the model can not be generalized to unseen data (regardless of the model's structure).

Nurtanto Diaz et al. [14] used KNN to determine if a lesion is benign or malignant. They used first order features, extracted from the ROI, as input to the classification and achieved an accuracy of 91.8%. Vijayarajeswari et al. [28] proposed the use of SVM classifier and achieved an accuracy of 94%.

Decision trees (DTs) [1] for mammogram classification (benign/malignant) were proposed by Kamalakannan and Rajasekhara Babu [8]. The used input was extracted from the bounding box containing only the abnormality.

3. Proposed approach

The current paper focuses on defining an approach for each of the above identified steps in order to facilitate the creation of a support system for digital mammogram classification. In the following sections the implemented algorithms are presented.

3.1. **Preprocessing.**

Mammograms are X-ray images taken from the breast tissue. Most of the mammograms also contain informative labels: the view of the image (MLO or CC – cranio-caudal) and the side (L – left or R – right) from which the image was taken. Besides the labels, small numbers can appear on them as well. The preprocessing's aim is to separate a subset of pixels needed to define whether the breast tissue is ill or not from the rest of the image.

First, the region of the breast has to be defined. The labels and artifacts outside this region are not relevant for a cancer detecting system. Morphological opening and histogram equalization are used in the first step to emphasize image features and to remove noise. After applying simple binarization to the resulting image we select the biggest region (the region of the breast) [25]. The threshold for the binarization is set to 50 (pixels with intensity less than this value does not contain information related to the breast): pixels with value greater than the given threshold will result in 1, otherwise in 0.

Next, we want to remove noninformative regions, which are placed inside the breast's area. This means the removal of the pectoral muscle, a triangular shaped area on the mammogram. It has similar intensity as the lesions. Therefore, with a very high probability the pectoral muscle will be detected as abnormal tissue. Hence, we want to separate it.

First, we implemented a sliding window algorithm proposed by Shrivastava et al. [25]. The method consists in the traversal of the image with a 5×5 window. While (1) the total intensity of the window is greater than a given value ($total\_intensity$) and (2) the difference between the top left and lower right corner is less than another value ($max\_difference$) the content of the window in the resulting image will be set to 0. One drawback of this method is that the intensity of the pectoral muscle and the soft tissue near can vary. Thus, the definition of $total\_intensity$ and $max\_difference$ is not straightforward. To overcome this problem we defined $total\_intensity$ separately for each image based on the intensities in the first window.

The other implemented method is based on seeded region growing (SRG) and it is proposed by Maitra et al. [12]. First, the range, where the pectoral muscle can be located, is decreased. For this four lines are defined: (1) AB on the left-, (2) CD on the right side of the breast, (3) CO connects the top of

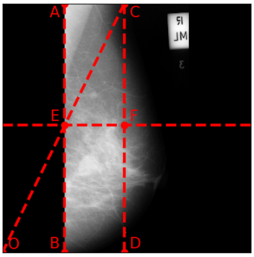


FIGURE 1. Guide lines used for SRG.

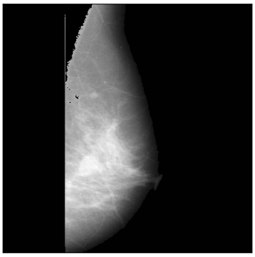


FIGURE 2. Result of SRG.

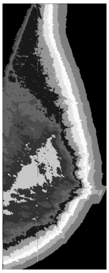


FIGURE 3. Segmentation of the ROI with k-means in 11 clusters.

the right side to the lower left corner and defines E (intersection of AB and CO) and (4) EF perpendicular to CO. The lines are marked in Figure 1. In most of the cases the pectoral muscle is inside triangle ACE. As recommended in the article seeds are selected from the the upper half of the first diagonal. After the seeds (S) are defined we calculate the average (S_{avg}) and maximum (S_{max}) intensity of the set. Next, we have to define which pixels to add to the region. For this we recalculate each pixel in triangle ACE by subtracting the average intensity and dividing it with the difference between the maximum and the average ($I' = \frac{I_{x\,y}-S_{avg}}{S_{max}-S_{avg}}$). The pixels with new value from interval $(0, 1]$ will be added to the region. The result will be the mask of the pectoral muscle. Figure 2 shows a mammogram after applying the got mask.

3.2. **Segmentation.** After defining the ROI, the next step towards a CAD system is the segmentation of the breast tissue. For this scope there are solutions in the literature both from the field of supervised and unsupervised learning. In this study we focus on the unsupervised segmentation methods.

We segmented each image with k-means algorithm [11] (see Figure 3). K-means is a clustering method that aims to split n observations into k clusters. The basic steps of the algorithm are (1) calculating the centroid of each cluster and (2) assigning each point from the input to the cluster with the closest centroid. The method stops when the difference between the new and the old cluster centroids falls below a given threshold. There are different researches in the literature on how to define the centroids in the first iteration ([21]). In this experiment we used random initialization. Also, the input for the algorithm in our case is a preprocessed mammogram image.

The application of the algorithm results in a segmentation of the mammogram (example in image Figure 3). We will use this result image together with the original one as input to the second block. Due to the use of clustered image in feature extraction the system will have more information about the shape of the cancerous tissue.

3.3. **Feature extraction.** Mammograms are gray-scale images. Hence, specific feature extraction is selected. Textural features are calculated from Gray-Level Run-Length Matrices (GLRLM) [5]. In case of a 2-dimensional images four GLRLMs can be calculated for directions: horizontal, vertical, first- and second diagonals. The result matrix will be a 2-dimensional array. As its name suggests the first axis corresponds to the intensity values and the second axis corresponds to the run length values. From each GLRLM 11 features can be derived, characterizing the distribution of short- and long runs in the input image in the specified direction. Chaieb and Kalti [5] included in their review the equations used to calculate these features. Consequently, for each image the descriptor contains $44 = 4 \times 11$ (4 directions and 11 features) elements.

3.4. **Feature selection.** The size of the result of the feature extraction can increase the computational cost, thus the classification progress can show a slow down. Therefore, filtering the best descriptive and independent features is the next step of our system. We propose PCA and a genetic algorithm (GA) based feature selection algorithm.

PCA [11] is the most used feature selection method. It consists in the calculation of covariance matrix from the input data and then applying eigen-decomposition on it. The results of the decomposition will be eigenvalue and -vector pairs, where the higher the eigenvalue the better the descriptiveness of the feature. The principal components are the first $n_{components}$ number eigenvectors with the highest eigenvalues. Using the calculated principal components the input data can be projected into a lower dimensional space.

GAs [11] are nature inspired, stochastic search algorithm. Its basic concept is to start with an initial population (usually randomly defines) and in each iteration create new individuals using genetic operations (selection, crossover and mutation). Each individual is evaluated and the best performing ones are added to the population. The algorithm stops when an individual exceeds a given threshold of fitting the result.

In case of feature selection individuals are binary vectors, where 1 means that the respective feature is selected and 0 otherwise. For the fitness function we selected a classifier (DT) and a performance measure (accuracy score). For each individual a DT is trained and its accuracy will correspond to the individual's fitness value. The method shown in the pseudocode in algorithm 1 represents the fitness function.

3.5. **Classification.** Classification is the progress of labeling observations based on examples. The built model extracts the characteristics of each class,

Algorithm 1 GA - genetic algorithm fitness function for feature selection

```
Require: $estimator, scorer, cv, individual$
Ensure: performance of the individual
BEGIN
$total\_metrics \leftarrow 0$
for $train\_indices, test\_indices \in cross\_validation\_folds(cv)$ do
    $X\_train, y\_train$ @ define train of features and ground truth based on $train\_indices$
    $X\_test, y\_test$ @ define train of features and ground truth based on $test\_indices$
    $current\_model \leftarrow estimator.fit(X\_train, y\_train)$
    $prediction \leftarrow current\_model.predict(X\_test)$
    $total\_metrics \leftarrow total\_metrics + scorer(prediction, Y_test)$
end for
return $\frac{total\_metrics}{cv}$
END
```

thus it will be able to differentiate classes for new inputs. In our system there are three classes: normal, benign and malignant.

In our experiments we considered Decision Trees (DT) [1] which are nature inspired classifiers. DTs are usually represented as binary trees where each leaf represents a predicted class, otherwise each node contains a condition.

The depth of a DT defines the performance of the classifier. However, there is a compromise between the performance and the chance of overfitting. The deeper the tree is the classification on the training data is more accurate, but this can cause poor performance on the test data (caused by overfitting).

K-nearest neighbors [1] is another classification method considered in out experiments. The class of a new observation is defined based on the majority class between its K nearest neighbors. The disadvantage of this method is that in case of high dimensional input the distance between any two points will be one (curse of dimensionality). Hence, feature selection is crucial for this type of classification.

Support Vector Machines (SVM) [1] are widely used classifiers based on statistical learning. The scope of the method is to define a hyperplane, which has the largest distance from the samples of each class (functional margin). For this the support vectors are used (perpendicular sections from the sample point to the plane). The advantage of the this approach is that it works well in high dimension.

4. Experiments

At the beginning of this section we describe the used dataset. In the second part the ran experiments and the achieved results are presented.

4.1. **MIAS.** Mammographic Image Analysis Society contains 161 pair (322) MLO mammograms. From the samples 207 are from healthy breast tissues, 64 are from benign lesions and 54 are from malignant cancerous tissues. This highlights the imbalance of the dataset. It is included a ground truth file

		True class	
		Positive	Negative
Predicted class	Positive	TP	FP
	Negative	FN	TN

Table 1. Contingency table, where TP and TN denote true positive/negative samples, while FP/FN denote false positive/negative samples.

in the dataset defined by radiologists. This file specifies the class of each mammogram (normal/benign/malignant).

We used simple train and test split to train our classifiers. Both in the training- and test sets the ration of the classes will be preserved. Due to this property of the split overfitting caused by missing classes from the set is prevented. The split is randomly defined by assigning 75% (241) of the data to the training set and the remaining 25% (81) to the test set.

4.2. **Metrics.** To evaluate the performance of the first block (preprocessing – pectoral muscle removal and segmentation) precision, recall and quality are used. For the second block instead of quality accuracy and f1-score are used. To calculate these values the elements of the contingency table (see table 1) are used, which contains the relation between the ground truth and the prediction.

4.3. **Results.** In this section we discuss the performance of each part in the constructed CAD system. In the experiment we implemented multiple methods, but in the previous sections we discussed the best performing one. In the following paragraphs we present the result of all the used methods.

4.3.1. *Pectoral muscle removal.* For the evaluation of the removal of the pectoral muscle first we need a ground truth. Maitra et al. [12] segmented the pixels in the ACE triangle (Figure 1) and selected the cluster corresponding to the pectoral muscle. These clusters were validated by radiologists and were taken as ground truth. The achieved results are presented in Table 2 (columns 2-4). In our experiment we re-implemented the used methods, but without the validation from radiologists we could achieve the results presented in the same table's last three columns (Table 2 – columns 5-7). Besides SRG we implemented a sliding window algorithm proposed by Shrivastava et al. [25] and the results achieved by our implementation are presented in Table 2 – column 8. This algorithm has a bit better performance than our SRG, but it could not outperform the original method.

4.3.2. *Segmentation.* To evaluate the segmentation, first we have to select the cluster of the abnormality. For this we use the ground truth given in the dataset. The cluster of the lesion will be defined by the maximum number

	Original SRG[12]			SRG			SWA
	Fatty	Gladural	Dense	Fatty	Gladural	Dense	
Precision	0.963	0.978	0.991	0.8125	0.7378	0.7637	0.8125
Recall	0.971	0.975	0.994	0.8930	0.8978	0.8627	0.8930
Quality	0.936	0.954	0.985	0.7415	0.6762	0.6731	0.7415

Table 2. Results of the pectoral removal on MIAS

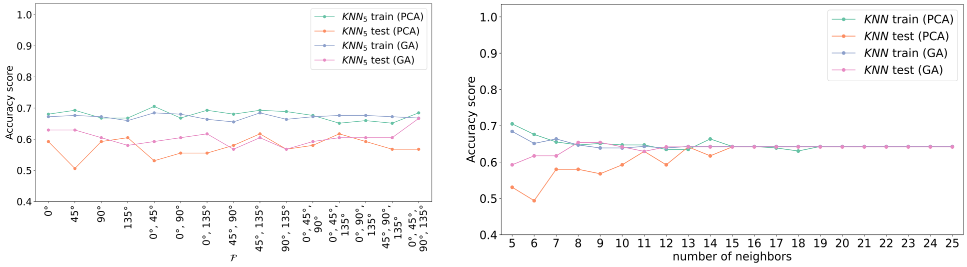
	k						
	4	8	10	12	14	16	18
Precision	0.8370	0.6522	0.5765	0.5230	0.4872	0.4531	0.4212
Recall	0.0771	0.1339	0.1464	0.1556	0.1646	0.1648	0.1763
Quality	0.0692	0.1043	0.1057	0.1068	0.1059	0.1021	0.0990

Table 3. Mean measures calculated from mammograms' segmentation containing tumors from MIAS for different number of clusters

of pixels overlapping with the ground truth. After defining this cluster we compare it with the ground truth and we calculate the measures mentioned in Section 4.2. The results are presented in table 3. These numbers are calculated from the mammograms that contain abnormality. As we can see the results are not satisfactory, and with the growth of the cluster number the precision drops faster compared to how recall increases. The cause of this phenomenon is the small number of clusters. With a low k value it is likely that the pixels in the ground truth are part of the same cluster (high precision value). On the other hand, other pixels outside of the ground truth are likely to be selected (low recall value).

4.3.3. *Classification.* In our research the input of the classification consists of the GLRLM features calculated from each image and its segmented version. GLRLM matrices are constructed in all four directions (0°, 45°, 90°, 135°). To reduce the dimensionality of the classification's input feature selection is applied. In the experiments the explained variance of the PCA is set to 99%. Consequently, the first two principal components are selected in general. For the evaluation of the feature selection we looked at the system's over all performance. In our approach, we built a single classifier with labels: normal, benign and malignant.

The result of the KNN classifier is shown in figure 4a and figure 4b. Figure 4a shows the performance of KNN classifier, where the five nearest neighbor is considered in the decision making. On x axis the different combinations of feature sets are visible. It shows that the KNN reaches its best performance with features $\{0°, 45°\}$ using PCA feature selection. The accuracy with GA feature selection is a bit below on the training set, but it's better on the test set. On the next figure (figure 4b) the feature set is fixed to $\{0°, 45°\}$ and the result of experiments with the number of neighbors is presented. We can see



(A) Number of neighbors fixed to 5 for all the possible combination of features.

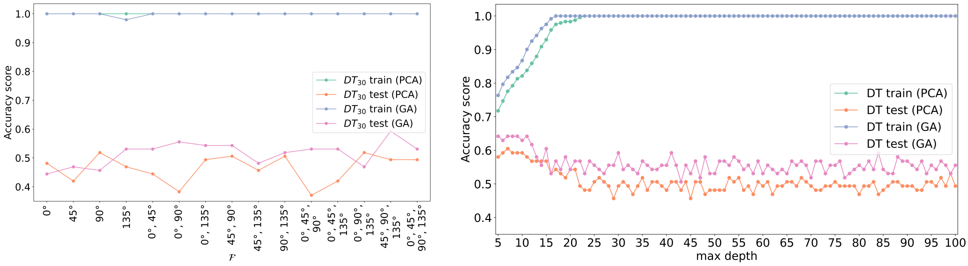
(B) Features fixed to $\{0°, 45°\}$ with different number of neighbors considered.

FIGURE 4. Performance of the KNN classifier.

that from higher than 15 for the number of neighbors in the classification the train an the test result is the same. This can be explained with the fact that the model predicts "normal" for 99% of the images and the ratio of the labels is the same in the train and the test set. Figure 4b shows that KNN on input filtered by GA has lower variance than on input filtered by PCA.

The same experiments were performed with the other classifiers. On figure 5a the performance of the DT model is shown using different feature combinations (using a maximum depth of 30). The result on the training set (with both selection function) is 100%, while on the test set using GA outperforms the results with PCA. The best test result is on set $\{45°, 90°, 135°\}$ using GA (59.2%). Figure 5b shows the result of changing the maximum depth of the DT classifier. It can be seen that with the increase of the maximum depth the accuracy calculated on the training set it is also increasing. On the other hand the metrics calculated on the test set are deceasing. This can be explained with the overfitting generated by the depth of the model.

Table 4 shows the results of the classifications. It can be seen that filtering features with GA and using DT clearly outperforms the rest of the classifiers on the training data. On the test set the difference in the metrics is smaller, and the combination PCA-KNN, GA-SVM have the highest accuracy. As mentioned above, this can be caused by overfitting in the DT because the best train accuracy is achieved when the depth of the tree is 30. The best performance, considering both the train and test results, was achieved by using GLRLM features for directions $\{45°, 90°, 135°\}$, GA feature selection with DT and accuracy used in its fitness function and 30 as the depth of the tree. With this setting the train accuracy and test accuracy was 100% and



(A) Maximum depth fixed to 30 for all the possible combination of features.

(B) Features fixed to $\{45°, 90°, 135°\}$ with different maximum depth considered.

FIGURE 5. Performance of the DT classifier.

		PCA				GA			
train	KNN	0.6556	0.6357	0.6556	0.5568	0.6722	0.7061	0.6722	0.5800
	SVM	0.6515	0.7316	0.7734	0.6197	0.6473	0.7296	0.7685	0.6089
	DT	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
test	KNN	0.6420	0.5211	0.6420	0.5410	0.6049	0.4866	0.7206	0.5809
	SVM	0.6173	0.4840	0.7353	0.5837	0.6420	0.6420	1.0000	0.7820
	DT	0.4938	0.5048	0.4938	0.4976	0.5926	0.5853	0.5926	0.5884

TABLE 4. Results of the feature selection and classification combinations for MIAS. From each feature selection group the columns correspond to accuracy, precision, recall and f1-score respectively.

59.2% respectively for classifying the mammograms into normal, benign and malignant classes.

4.4. **Discussions.** Based on the results shown above, from the investigated methods the combination of SRG, GA and DT achieved the best result for a breast cancer detecting CAD system. The reported test accuracy for classification in three classes (normal, benign and malignant) is 59.2%. In this section we compare the results of methods from the state-of-the-art using the same database, feature extraction, -selection or classification.

Srivastava et al. [27] proposed the use of GA feature selection on features extracted from mammograms in MIAS. They used histogram, texture, geometric, wavelet and Gabor features, and after feature selection an SVM-MLP is used to classify the data into normal and abnormal classes. They reported an accuracy of 87%. A combination of GA and RF was proposed by Rouhi et al. [17] on GLCM, shape and intensity histogram features. They achieved a result of 74.44% on classifying images in benign and malignant classes.

Recent researches using GLRLM features used different label in the classification or multiple classifiers compared to the proposed approach. Candra et al. [4] built three models to classify mammograms in normal, benign and malignant classes. They used SVM models with different kernels, but the best result (93.97%) was reported using polynomial kernel.

The mentioned result above can not be directly compared to the proposed approach because the aim of the classification is different. The deviation of the result metrics can be caused because the difference in the train test split or in the used dataset (MIAS or DDSM).

5. Conclusions and future work

The scope of the current paper was to construct a system that helps detecting breast cancer by classifying mammograms into normal, benign and malignant categories. This process involved the usage of preprocessing, segmentation, feature extraction and selection. Based on the exhaustive experiments conducted on methods for the steps of image classification it is concluded the best performance achieved is by using a combination of SRG, GLRLM $\{45°, 90°\}$, GA and DT with a 100% training accuracy and 59.2% test accuracy.

In the current approach MLO mammograms are used as input to the presented system. Therefore, it will not work with CC images. In future work other preprocessing methods and ROI definition are planned to be investigated. In future work, investigating pruning might improve the performance of DT classifiers by reducing overfitting. We will further investigate the use of other feature selection methods and classifiers (for instance neural networks). In the current paper, we built a single classifier. Hence, in a future work we can build two binary classifiers (one to decide if the breast tissue is normal or abnormal and a second classifier to decide if the lesion is benign or malignant) equivalent to the proposed one. Furthermore, cross validation is planned to be investigated instead of simple train test split. MIAS is an unbalanced dataset, therefore investigating different balancing techniques might increase the performance of the system. In addition, in furute work a detailed comparative analysis will be included between the methods presented in the literature and the proposed one.

Acknowledgements

This work was supported by a grant of the Romanian Ministry of Education and Research, CCCDI - UEFISCDI, project number PN-III-P2-2.1-PED-2019-2607, within PNCDI III.

References

[1] Aggarwal, C. C. *Data Classification: Algorithms and Applications*, 1st ed. Chapman & Hall/CRC, 2014.

[2] Arora, R., Rai, P. K., and Raman, B. Deep feature–based automatic classification of mammograms. *Medical & Biological Engineering & Computing* ***58***, 6 (June 2020), 1199–1211.

[3] Bali, A., and Singh, S. N. A review on the strategies and techniques of image segmentation. In *Proceedings of the 2015 Fifth International Conference on Advanced Computing & Communication Technologies* (USA, 2015), IEEE Computer Society, p. 113–120.

[4] Candra, D., Novitasari, R., Lubab, A., et al. Application of feature extraction for breast cancer using one order statistic, GLCM, GLRLM, and GLDM. *Advances in Science, Technology and Engineering Systems Journal* ***4***, 4 (2019), 115–120.

[5] Chaieb, R., and Kalti, K. Feature subset selection for classification of malignant and benign breast masses in digital mammography. *Pattern Analysis and Applications* ***22***, 3 (Aug. 2019), 803–829.

[6] Esener, I. I., Ergin, S., and Yuksel, T. A novel multistage system for the detection and removal of pectoral muscles in mammograms. *Turkish Journal of Electrical Engineering and Computer Sciences* ***26*** (2018), 35–49.

[7] Htay, T. T., and Maung, S. S. Early stage breast cancer detection system using glcm feature extraction and k-nearest neighbor (k-nn) on mammography image. In *18th International Symposium on Communications and Information Technologies* (2018), pp. 171–175.

[8] Kamalakannan, J., and Babu, M. R. Classification of breast abnormality using decision tree based on GLCM features in mammograms. *International Journal of Computer Aided Engineering and Technology* ***10***, 5 (2018), 504–512.

[9] Kim, W., Kanezaki, A., and Tanaka, M. Unsupervised learning of image segmentation based on differentiable feature clustering. *IEEE Transactions on Image Processing* ***29*** (2020), 8055–8068.

[10] Li, H., Chen, D., Nailon, W. H., et al. Dual convolutional neural networks for breast mass segmentation and diagnosis in mammography, 2020.

[11] Liu, H., and Motoda, H. *Computational Methods of Feature Selection*. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series. CRC Press, 2007.

[12] Maitra, I. K., Nag, S., and Bandyopadhyay, S. K. Technique for preprocessing of digital mammogram. *Computer Methods and Programs in Biomedicine* ***107***, 2 (2012), 175–188.

[13] Moghbel, M., Ooi, C. Y., Ismail, N., et al. A review of breast boundary and pectoral muscle segmentation methods in computer-aided detection/diagnosis of breast mammography. *Artificial Intelligence Review* ***53***, 3 (Mar. 2020), 1873–1918.

[14] Nurtanto Diaz, R. A., Nyoman Tria Swandewi, N., and Pradnyani Novianti, K. D. Malignancy determination breast cancer based on mammogram image with k-nearest neighbor. In *2019 1st International Conference on Cybernetics and Intelligent System* (2019), vol. 1, pp. 233–237.

[15] Rahimeto, S., Debelee, T. G., Yohannes, D., et al. Automatic pectoral muscle removal in mammograms. *Evolving Systems* (Nov. 2019).

[16] Rashed, E. A., and Awad, M. G. Neural networks approach for mammography diagnosis using wavelets features, 2020.

[17] Rouhi, R., Jafari, M., Kasaei, S., et al. Benign and malignant breast tumors classification based on region growing and CNN segmentation. *Expert Systems with Applications* ***42***, 3 (2015), 990–1002.

[18] Sadeghi, B., Karimi, M., and Mazaheri, S. Automatic suspicions lesions segmentation based on variable-size windows in mammography images. *Health and Technology* ***11***, 1 (Jan. 2021), 99–110.

[19] Salama, M. S., Eltrass, A. S., and Elkamchouchi, H. M. An improved approach for computer-aided diagnosis of breast cancer in digital mammography. In *IEEE International Symposium on Medical Measurements and Applications* (2018), pp. 1–5.

[20] Sarker, O., Akter, S., and Mishu, A. A. Review on the performance of different types of filter in the presence of various noises. *Engineering International* ***4***, 2 (dec 2016), 49–56.

[21] Saxena, A., Wang, J., and Sintunavarat, W. An empirical study on initializing centroid in k-means clustering for feature selection. *International Journal of Software Science and Computational Intelligence* ***13***, 1 (jan 2021), 1–16.

[22] Selvathi, D., and Aarthy Poornila, A. *Deep Learning Techniques for Breast Cancer Detection Using Medical Image Analysis.* Springer International Publishing, Cham, 2018, pp. 159–186.

[23] Shen, L., Margolies, L. R., Rothstein, J. H., et al. Deep learning to improve breast cancer detection on screening mammography. *Scientific Reports* ***9***, 1 (Aug. 2019).

[24] Shinde, V., and Thirumala Rao, B. Novel approach to segment the pectoral muscle in the mammograms. In *Cognitive Informatics and Soft Computing* (Singapore, 2019), pp. 227–237.

[25] Shrivastava, A., Chaudhary, A., Kulshreshtha, D., et al. Automated digital mammogram segmentation using dispersed region growing and sliding window algorithm. In *2nd International Conference on Image, Vision and Computing* (June 2017), pp. 366–370.

[26] Solorio-Fernández, S., Carrasco-Ochoa, J. A., and Martínez-Trinidad, J. F. A review of unsupervised feature selection methods. *Artificial Intelligence Review* ***53***, 2 (Feb. 2020), 907–948.

[27] Srivastava, S., Sharma, N., Singh, S., et al. Quantitative analysis of a general framework of a CAD tool for breast cancer detection from mammograms. *Journal of Medical Imaging and Health Informatics* ***4***, 5 (Oct. 2014), 654–674.

[28] Vijayarajeswari, R., Parthasarathy, P., Vivekanandan, S., et al. Classification of mammogram for early detection of breast cancer using SVM classifier and hough transform. *Measurement* ***146*** (2019), 800–805.

[29] Wang, X., Liang, G., Zhang, Y., et al. Inconsistent performance of deep learning models on mammogram classification. *Journal of the American College of Radiology* ***17***, 6 (2020), 796–803.

Babeş-Bolyai University, Faculty of Mathematics and Computer Science, 1 Mihail Kogălniceanu, Cluj-Napoca 400084, Romania

Email address: `adel.bajcsi@ubbcluj.ro`

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXVI**, Number 2, 2021
DOI: 10.24193/subbi.2021.2.03

A REVIEW AND ANALYSIS OF THE EXISTING LITERATURE ON MONOCHROMATIC PHOTOGRAPHY COLORIZATION USING DEEP LEARNING

ALEXANDRU MARIAN ADĂSCĂLIŢEI

Abstract. It is universally known that, through the process of colorization, one aims at converting a monochrome image into one of color, usually because it was taken by the limited technology of previous decades. Our work introduces the problem, summarizes the general deep learning solutions, and discusses the experimental results obtained from open-source repositories. Although the surveyed methods can be applied to other fields, solely the content of photography is being considered. Our contribution stands in the analysis of colorization in photography by examining used datasets and methodologies for evaluation, data processing activities, and the infrastructure demanded by these systems. We curated some of the most promising papers, published between 2016 and 2021, and centered our observations around software reliability, and key advancements in solutions employing Generative Adversarial Networks and Neural Networks.

1. Introduction

Photography colorization, in the context of this paper, represents the procedure of artificially reconstructing color information in a picture that has never been captured on a storage medium capable of recording color. In the absence of such research, we took the challenge of providing a comprehensive perspective on deep learning solutions. Approaches vary, with examples of discriminative networks [4, 52], generative networks [15, 53], and adversarial ones [1, 6]. In an area which have seen solutions as early as 1980, and modern ones began to appear around 2002, only the recent years brought methods to yield impressive results. A discussion was opened on 28 papers and 17 datasets, leading us through four main patterns, each with different models, computational demands, and time costs. Through patterns, a common set of

Received by the editors: TBA.

2010 *Mathematics Subject Classification.* 68T99.

1998 *CR Categories and Descriptors.* I.4.8 [**Image Processing and Computer Vision**]: Scene Annalysis – *Color*.

Key words and phrases. Photography Colorization, Deep Learning.

input values and processing steps are grouped under a common name, while the models encompass technically detailed networks. However, color reconstruction may yield structurally incoherent result, due to the lack of similar visual content in the training phase of the models. To better assess, experiments were conducted on a dataset of our own, managing to confirm a common trend regarding colorization performance.

On the one hand, our research methodology aimed to study the existing patterns, highlighting the main differences. On the other hand, we focused on the architecture, and the use of data. Aside from photography, domains such as communication protocols, medical imaging, and gaming could benefit from data compression, physiological highlights, and photo-realistic scene renderings, respectively. This paper contributes with a thorough analysis of the existing papers and datasets, process guided by the following research questions.

1.1. **Aims and Research Questions.** While our primary concern was the best possible coverage of the literature, software reliability and open access to the source code shaped our approach to a good extent, making us ask the following initial research questions `RQ 1` to `4`.

- `RQ 1` As for now, is automatic colorization achievable without deep learning?
- `RQ 2` What solving patterns and deep learning models are usually employed?
- `RQ 3` How well would these models perform in professional applications?

The rest of the paper analyzes the work of the previous half of a decade, with a five section structure, having the context and relevance of colorization described in Section 2, patterns and models of learning in Section 3, literature result analysis in Section 4, and the conclusions summarized in Section 5.

2. Context and Relevance

In explaining how light is stored on a computer, this section builds an intuitive reasoning path to understand why mathematical reconstruction formulas can not infer color directly from the grayscale image.

2.1. **Digital Representation.** When it comes to the pictures we store on our computers, they can be thought of as grids of numerical values, stacked upon each other. In this stack, each layer stores in those numeric values information regarding light absorption, by converting the electromagnetic wavelengths to a color standard computers can reproduce. The purpose of such structures is to mimic the eye's response to natural light but in the context of a screen. For example, transitioning from the wavelength measurements to the 1931

Commission Internationale de l'éclairage (CIE) standard [41] one uses three empirically determined weights, $\bar{x}_\lambda$, $\bar{y}_\lambda$, $\bar{z}_\lambda$, called tristimulus. As they only measure the color perception, historically defined by a group of human observers, we will use these values to determine the actual components of CIE, namely X, Y, and Z.

Color Space. The fundamental aspect of black and white photography, and the reason behind lacking mathematical formulas for color reconstruction `RQ1`, is that the technology capturing the visible light, either film rolls or digital sensors, only keeps the brightness, a weighted average of the mixed wavelengths we call colors. What was previously a rich source of information, now it became unable to tell what colors were in the scene. Using the Lab format, one may train various models to predict the chromaticity based on the luminance channel, pretending it was the black and white image, and then, using the discarded channels, measure the distance between the prediction and the reality. A candidate model makes the transition from black and white to color stacking two layers of chromaticity, which in the case of Lab one describes blue/yellow, and the other one orange/violet information.

2.2. **Progress and Relevance Over Time.** Colorization is a practice as old as photography itself, dating back to the eighteenth century, but recent technological breakthroughs endowed us with tools capable to supplement the monochrome record with a more appealing visual representation. Recording light on a physical medium had to overcome the technological limitations imposed by the early photographic materials which only captured shades of black and white. At the beginning of the nineteenth century, around the year 1930 [21], color photography gave people access to an unconscious understanding of the physical world through color. Nowadays, teams of artists such as Dynamichrome [11], are closing the gap, manually reconstructing black and white records. They are deciding how to approach the task, using their experience and intuition. Similarly, deep learning algorithms are calibrating their parameters until the model is behaving as intended.

Methods that predate the 1980s had more of a mechanical nature, using some of the early iterations of computers capable of graphical manipulation. Although little research has been made publicly available until 2002, Wilson Markle and Brian Hunt patented one of the earliest, if not the first attempt of movie colorization [34] using a computer, in 1988. For each scene of a black and white film pellicle, a color mask was applied to the frame. The adjacent frames were then similarly colored, taking the motion into consideration. Each of the areas indicated as having motion was processed by some adjacent pixels algorithm, while static areas were inheriting the previously applied colors.

3. Colorization Patterns and Learning Models

The following section is dedicated to answering RQ2 through a brief introduction into the patterns' general idea, and a discussion on the models' traits.

3.1. **Colorization Patterns.**

The source of data and the processing pattern can have a decisive performance impact on the model. Predicting color channels depends on the type of information sources one has at their disposal, whether it implies contextual hints, or large amounts of color images.

3.1.1. *Data-Driven Colorization.* Due to the fact that early algorithms heavily relied on human interventions, the work that followed completely removed preferences coming from the outside of the system. Su et al. [42] separated the image as a whole from the objects within the frame, colorized each patch using a network inherited from [52], and later fused the features while also avoiding the artifacts. Even if limitations may appear when the object instances are not well detected, it usually generates results with fine-tuned details without human interventions. Presumably, from the idea of finding the best local match, and balancing global coherence, improved approaches will be derived. All such approaches leverage large scale data and end-to-end training. Nevertheless, the decision of relying of fully autonomous processes was later reverted, and human preferences began to be taken into account under various forms that will be discussed in the following paragraphs.

3.1.2. *Human-in-the-Loop Colorization.* The shared knowledge of a community, historical documents, or reference images contain information that artists may access and use in their work, yet software methods are still unable to deal with such diversity and spread of information when transferring color. The following methods embrace the multi-modality of the problem, providing colorization results that differ when changes are iteratively introduced by a person. From such interactions, reinforcement learning may better predict what would be of interest for humans in color photographs. However, seldom is reinforcement learning present in the scene of colorization networks.

Based on Textual Descriptions. Notes were often placed on the back of legacy photography, and even nowadays, colorization associated with a language has rich sources of training data. Many social media platforms are improving their indexing systems based on the words and sentences associated with the visual content. Photography colorization based on captions conditions the nuances to fit color palettes associated with the present words, building on the idea that particular colors are associated with complex semantic concepts. One may imagine that a *cold evening* varies in nuances of blue, while the *golden* *hour* covers everything in warm colors. Regions that could not be matched from the text are then processed using a dominant color, such as *denim blue*.

Manjunatha et al. [33] concatenated units of text into every convolutional block of the baseline network - a fully convolutional neural network, obtaining a model that joins textual and visual feature maps at the cost of significant parameter demands. To address the issue of parameter efficiency, the authors employed a second approach to fuse the representations, using a feature-wise linear modulation - Perez et al. [37]. Training on the dataset presented in Lin et al. [31] yield unsatisfactory colorization due to image complexity and the set size limitation, although accounting for over $82 \cdot 10^3$ images. In these circumstances, the baseline network was pre-trained on ImageNet, then the two network variants presented in [33] were fine-tuned. The evaluation confirmed a better precision in the second model, although no significant differences were observed in both evaluation metrics or Turing tests. Both models performed well under caption changes, and they were able to change the colorization according to the updated sentences.

Image segmentation based on natural language expressions was approached by Hu et al. [17], then based on their framework Chen et al. [7] improved on features fusion, using a recurrent attentive module for deciding the number of text-to-image processing iterations. The framework matches image regions with the words describing them, employing the recurrent attentive fusion module that repeatedly reads the textual features maps until, through the attention mechanism, enough information was retrieved. A deconvolutional network later takes the fusion features map and up-scales it to the width and height of the final image, with a depth comprised of the number of classes resulted in segmentation and two chromaticity channels. Additionally, Chen et al. [7] introduced the first colorization results obtained on Oxford-102 Flowers dataset [36].

Bahng et al. [3] proposed two generative adversarial networks, one for text-to-palette generation, T, and another one, P, for palette-based colorization. The generator of T learns mappings between color palettes and sequences of words, while the discriminator distinguishes between real and fake palettes, using the Huber loss across the network. P operates on two sub-networks, a U-Net based colorization network, and a network that guides the colorization based on the color palette generated by the caption, whose output is passed to a Deep Convolutional Generative Adversarial Network (DCGAN) discriminator. Provided with rich textual resources, the model generates multiple color palettes, adapting to more than a couple of words, thus contrasting the limitations of small input volumes that previous work brought. Following this idea, a dataset of more than 10^3 mappings between sequences of words and palettes of five colors was introduced, and later applied on T's training.

The language itself makes a great difference in colorization, as English has eleven basic color categories, Russian twelve, and other languages might drastically differ, with the number of color terms reaching as low as three - white, dark, and red. Berlin and Kay theory addresses how various cultures share a basic understanding of color, even if they have various manifestations at the vocabulary level. Loreto et al. [32] presented a multi-agent simulation on how the use of a language influences color terms.

Based on Color Hints. Learning deep priors was not always the obvious path, and the early colorization approaches were envisioned to spread color strokes in correlation with the luminosity channel. Deep image priors represent a network's ability to obtain some knowledge about the world, and then use it in the actual task, where such knowledge comes in handy, and alone it is not enough to find the answer we seek. Data-driven processing turned the coin, making the process easier at the expense of user control. Might the best of both worlds be obtained, then a truly robust tool would be handed to the creatives ones, allowing for colorization preferences that would be difficult to include otherwise. In the following paragraph, we explain how one may combine user preferences and deep priors. Such user preferences come under the form of hues defined at a specific point on the digital canvas (tablet/monitor). Aside from the arbitrarily selected hues, the color hints get propagated through the network after specifying them on the user interface.

Zhang et al. [52] beautifully covered the specifics of this pattern, employing a CNN fusing low-level features extracted from clues with high-level semantic information. The main branch uses a U-Net architecture, which additionally absorbs the sparse color points through a Local Hints Network, L, and either the histograms or the average saturation levels using a Global Hints Network, G. Whenever the preference for a color is expressed on a drawing pad, a recommendation of nine colors is obtained through running a k-means clustering on G's final per-pixel distribution. In our experiments, this method ranked first when using their baseline model - the one with no hints provided. The user interface requires up to 8GB of RAM for the Docker image, but the experience is impressive. The codebase can be used without the graphical part, as the repository is very well documented and maintained. As an improvement, Xiao et al. [48] allowed for both global and local hints to be provided concurrently, in contrast to only one type of hint at a time as it was permitter in the previous approach [52].

Based on Reference Color Images. Transferring the chromaticity information from a semantically related color image to a target $\mathtt{T}$ monochromatic image is the main focus of this paradigm, and whether the user provides a reference $\mathtt{R}$ color image, or the system manages to retrieve the appropriate one, the idea is to allow for a multi-modal colorization, which neural networks prevent from happening using the dominant colors they have learned. One could imagine passing colors from cherry blossom to a black and white Californian coast image, obtaining synthetic, but artistic pink waves. Finding images with similar semantics and luminance as the input we want to process might prove as difficult as giving the right hints. Thus, in He et al. [16] an image query reaches to a gray-VGG-19 which in turn, based on its class, and the cosine similarity of the tuples $(\mathtt{R_i}, \mathtt{T})$ computed using the features $\mathtt{F}^5_{\mathtt{R_i},\mathtt{T}}$ and $\mathtt{F}^6_{\mathtt{T},\mathtt{R_i}}$ from network's last convolutional layer and first fully-connected layer, narrows down the top n images, generating a global ranking. Then, further pruning is realized using semantic and luminance similarities, which are denoted in Equation 1 as the sum's two terms.

$$\mathtt{score}(\mathtt{R_i}, \mathtt{T}) = \sum_{\mathtt{p}} (\mathtt{d}(\mathtt{F}^5_{\mathtt{T}}(\mathtt{p}), \mathtt{F}^5_{\mathtt{R_i}}(\mathtt{q})) + \beta \mathtt{d_H}(\mathtt{C_T}(\mathtt{p}), \mathtt{C_{R_i}}(\mathtt{q}))) \tag{1}$$

Where $\mathtt{i} = \overline{\mathtt{0}, \mathtt{n}}$, β has been empirically set to $\mathtt{0.25}$, and $\mathtt{T}$ is our grayscale image, for each point p from F^5_T the nearest neighbor q from $F^5_{R_i}$ is assigned so that the pair minimizes the cosine distance. Then, $\mathtt{C_T}(\mathtt{p})$ maps each point from the feature map $\mathtt{F}^5_{\mathtt{T}}$ to a grid cell from a down-scaled $\mathtt{16} \times \mathtt{16}$ resolution T, which in turn gets used in $\mathtt{d_H}$ to compute the luminance similarity. The semantic similarity directly applies the cosine similarity represented by $\mathtt{d}(\mathtt{x}, \mathtt{y})$. After the local ranking is determined, the reference retrieval algorithm yields the top-1 reference image. The visual attribute correspondence technique used is known as Deep Image Analogy, which is explained at length in the work of Liao et al. [30].

A general downside of this pattern are the unrelated spots that should be dealt with, thus He et al. [16] employs an end-to-end colorization subnetwork that simultaneously learns color sample selection, color propagation, and dominant color prediction on two sub-branches, one for chrominance, and one for perceptual correlation. While the chrominance branch propagates color samples extracted from the reference to the entire image, the perceptual branch makes a prediction for areas left uncolored by the reference, purely based on dominant colors learned from the large-scale training set. This pattern was also used in the work of He et al. [16], and Xu et al. [49].

3.2. **Deep Learning Models.** One would have to create a list of initial study sources, therefore we provided in Table 1 our recommendation in terms papers that would represent a good read. The 28 papers influenced our opinion on the matter, and although we did not refer directly to all of them, the manner we grouped and filtered may represent a valuable source of information. In addition, it would be fair to say that one approach does not account for all our expectations, thus we focused on their strengths at the network level, making small remarks visible on the fourth column.

Architecture	Datasets	Metrics	Strengths	Related studies
Convolutional Neural Networks	[5], [9], [13], [39], [44], [47], [55]	LPIPS, PSNR, SSIM	produces excelent predictions for first time encountered parts of an image	[4], [16], [18], [26], [27], [33], [42], [48], [50],[51], [52]
Network Refinement	[5], [14], [22], [36], [39], [46]		optimizes on conservative predictions	[2], [7], [8], [10], [15], [38], [40]
Transformer	[39]			[25]
Generative Adversarial Networks	[3], [23], [28], [39], [43], [47], [54], [55]		less artifacts, better skin nuances, reduced blue bias for clothing	[1], [3], [6], [12], [19], [20], [29], [35], [45]

Table 1. Literature recommendations with the codebase freely available on GitHub.

3.2.1. *Convolutional Neural Networks.* This class of models is known for the heavy use in computer vision tasks. In the larger scheme of discriminating or generating numerical values, starting from a 2D tensor representing the luminosity, and ending up with two chromaticity tensors, the network's layers, made out of convolutional kernels, are optimized and interconnected to improve the end result. When convolved with the input, these filters are generating the feature maps. In colorization, two important aspects must persist: the image ratio, which can be managed with padding, and that one should avoid image distortions, preferring a stride operation for pooling in the case of downsampling. The input image resolution ranges between 64 and 512 pixels, while some models have no restrictions on the input resolution, but they yield results within the previous boundaries. In the Lab format, the color values range between -128 and 128, and get later transformed so that they match the last layer activation (for example, for `tanh` would range between -1 and 1).

In general, the spatial information gets encoded, and lost, in exchange for learning more about the input image, procedure associated with an encoder. It is common to notice that additional features are added, fusing them into the output of the encoder, as they give us a stronger sense that we are in the possession of an improved solution. As an example, Baldassare et al. [4] used a pre-trained Inception-ResNet-v2 for features extraction alongside the encoder. Then, the model is upsampling the compact representation, using as much of the first layers as it needs to bring back spatial information. While different approaches leverage different parts of the network to their advantage, they have something in common in the way all approaches try to compress as many and insightful features together and to create the chromaticity channels out of them. In addition, hypercolumns are often used in this context (for example in Larsson et al. [27]), because the last layer gives information too coarse to precisely localize chromaticity descriptors in the pixels space, thus storing the activation values for a pixel increases prediction accuracy in the deeper layers.

Iizuka et al. [18] designed their approach based on Krizhevsky et al. [24], with four components: three networks thought for low, middle, and global features extraction, and a colorization network. A particularity of their work was that they allowed for input files of any resolution, global image priors, and colorization style transfer. When fusing global features with a purpose similar to that of priors into the local features, the environmental information influenced the colorization, avoiding, for example, green nuances for the water surface. The model was trained exclusively on 224×224 pixels images from [55], augmenting via cropping from an initial 256×256 pixels, and randomly flipping in the vertical orientation. According to the authors, results may be obtained on one of NVIDIA® Tesla® K80 GPU cores, with a batch size of 128, and 11 epochs (accounting for $2 \cdot 10^5$ iterations), in approximately 3 weeks time. As a comparison point, in the work of Baldassarre et al. [4] the same GPU unit completed the training stage in 23 hours, using a batch of 100, and $6 \cdot 10^4$ images, supporting the previous time estimation for training on the entire ImageNet dataset (which contains $14 \cdot 10^6$ pictures). For He at al. [16], training for 10 epochs, with a batch size of 256, took 2 days on eight Titan XP GPUs. Two days were also enough for Xiao et al. [48] to train their model using a batch size of 50 images, $4 \cdot 10^4$ iterations on NIVIDIA's GTX1080Ti GPU.

Larsson et al. [27] discarded the classification layer of a VGG-16 and transformed this fully convolutional network into a model in which each pixel had a probability distribution assigned over 313 ab pairs, a quantized color space that may vary in size from one implementation to another. While the idea was gaining traction due to existing progress documented in Zhang et al. [51], it later influenced the hint-based work of Zhang et al. [52], in which it was shaped into a pixel-level color recommendation. A VGG inspired network was also used in Zhang et al. [51], adding depth, dilated convolutions, and an improved loss function in the form of a classification loss to compare the probability distributions, while also making use of class rebalancing, without which desaturated colors would have dominated. A VGG-19 was used in both the similarity sub-network and the colorization one in the approach of He et al. [16]. Other networks were remarked, such as the GoogleNet, AlexNet, and Capsule Neural Networks, as well as those described in the work of Guadarrama et al. [15] and Zhao et al. [53] which use generative models, with a Pixel Convolutional Neural Network in the first approach, and a color distribution generator, coupled with a pixelated semantic generator in the latter.

3.2.2. *Generative Adversarial Networks.* Such networks, abbreviated GANs, share a fair amount of traits with the work presented in the previous subsections, consisting of two smaller networks. As the name denotes, the two networks compete, having a generator network produce images indistinguishable from ground truth, and a discriminator classify which pair of images contains the original color version. The training ends when the classification no longer distinguished between the two types of images, real, and colorized. The target is to avoid conservative predictions, and allow multiple colorization results by varying the noise, thus offering highly realistic results. Conditional GANs are most often employed, as the grayscale image represent part of the input and it could not be transformed into randomly generated noise as the traditional models would need. The generator takes the monochromatic image as a prior, and later allows for multi-modality through noise applied in the form of dropouts, or multi-layer noise coupled with multi-layer conditional information.

While the work of Nazeri et al. [35] had both the discriminator and the generator implemented after the U-Net architecture, the work of Cao et al. [6] envisioned an alternative to the encoder-decoder structure. One may image the encoder-decoder structure, where the middle part contains a U-Net architecture with skip connections between the layer $\mathtt{i}$ and $\mathtt{n} - \mathtt{i}$ to compensate for the bottleneck that prevents the low level information to reach the last layers. Such approaches tend to process the overall image information, which is suitable for transformations at the whole image scale, but in the case of colorization, it lacks local guidance. Nazeri et al. [35] embraced this method, and noticed, among other things, improved performance in the generator's encoder when leaky ReLU was applied. However, Cao et al. [6] preserves details at their location in space by using only convolutional layers in the generator. The noise gets attenuated when introduced early, hence it would be beneficial to introduce it in multiple layers. Complementing it, the multi-layer conditional information may be easily achieved, due to the fact that the network never used spatial transformations that would have complicated the process. Both [35] and [6] were inspired by the work of Isola et al. [19], given that the general idea of image-to-image translation has strong points that could be adapted from case to case.

An insightful read is the work of Antic et al. [1], called DeOldify. To the best of our knowledge, it remains the only competitive approach that was not associated with a research paper. Antic introduced a new breed of architecture, called NoGAN. Independently training the generator and the discriminator gives us most of the insights we need, then, GAN training addresses the issue of colorization realism. Shortening the GAN's training manages to avoid artifacts formation, while also closing the gap towards vivid colors. When the two networks are to be trained together, an inflection point in training will be noticed shortly, marking the moment when the critic managed to reach a learning threshold. When reached, the training must end, otherwise the quality varies drastically. Although not yet defined, the inflection point was determined by saving the checkpoints at each 0.1% of training data, and then manually inspecting whether the quality of the images did abruptly drop. This approach offers an artistic model, addressing details and color saturation, and a stable one, tailored for landscapes and portraits. In the same category of rarely visited ideas, we noticed the PatchGAN discriminator employed in the work of Victoria et al. [45]. Further exploration regarding pixel-level independence between two patches could offer an excellent penalty system in colorization.

4. Literature Results Analysis

Since the early '80s, the number of solutions proposed in literature remained small, in the two digits figure, and out of those, the human eye may be fooled by only a dozen of these algorithms. To further support research initiatives in legacy photography colorization, we have manually curated a 102-photograph dataset, shot on both film and digital mediums. Table 2 presents the results obtained from a variety of techniques, studying the context in which these models perform best, but also when they reach their limitations. For example, we often encountered models poorly selecting color distributions for landscape scenes, while at the same time, accurate color palettes for portraits. The results presented in this table were obtained from the open-source implementation made available by the authors of these papers on GitHub. The initial codebase was not changed in any manner.

In Table 2, the three columns denoting metrics, `LPIPS`, `PSNR` and `SSIM` rank the models by statistical means, and they will be introduced in Section 4.2. A number of factors contribute to a low score, such as patches left untouched, colors mappings without any real grounds, or spots leaking color into the immediate vicinity. Most models process landscapes and nature scenes well, while only particular portraits, urban events, and outdoor activities may deceive a person. Even if the work of Antic et al. [1] and Iizuka et al. [18] sometimes yields an unconvincing version of reality, it is impressive how those colors can, at the same time, provide a starting point for artists, and a bridge to the past for the general public. The last column summarizes the type of images we believe, based on the experiments, that would optimally be colorized. We aligned our results with those obtained in He et al. [16], Su et al. [42], and Zhang et al. [52], thereby agreeing with the general trend.

An improved performance can be observed on the generative models' side. The first column ranks the performance starting from the lowest score, while the other two columns rank in the opposite order. The metrics may have specific ranges of values, yet it remains a problem specific issue. The colorization has, as for the moment, no testing methodology, and this state of development leaves an opportunity for further research initiatives. To answer `RQ3`, the existing methods can deliver when used in professional photography tasks, being integrated into products targeting the general public. One example is the work of Zhang et al. [52] that was included in Photoshop Elements 2020.

Paper	Colorization Metrics						Recommended types of images
	↓ LPIPS	σ	↑ PSNR	σ	↑ SSIM	σ	
Zhang et al. [52]	0.11678	0.04927	18.69112	3.41512	0.88102	0.08394	all
Iizuka et al. [18]	0.18068	0.06863	15.80264	3.94617	0.77813	0.12155	events, portraits, landscapes
Antic et al. [1]	0.18389	0.08614	13.36557	3.55204	0.73828	0.12560	all
Zhang et al. [51]	0.22174	0.08790	13.60779	4.01649	0.77388	0.11998	landscapes
Kumar et al. [25]	0.30766	0.07357	11.22693	3.14602	0.53996	0.15731	close-up portraits, landscapes

Table 2. Performance evaluation made on a 102-image dataset (`github.com/alexdarie/color/images`) containing urban landscapes and events, objects, and portraits.

4.1. **Datasets Challenges.** The main disadvantage when solving this task is the training data, as we encountered only a hand full of datasets specifically designed for the task, as for example the Palette-and-Text dataset [3], or the Chinese Youth Subculture dataset [29]. Aside from these, the existing solutions inherited the most popular computer vision training sources. An overview can be found in Table 1. Often, images from other tasks are either semantically too simple, too small resolution-wise, or they lack descriptors (textual or color clues), thereby partially preventing the learning process. Although they might seem numerous, the existing sets lack diversity present in consistent amounts. Such a balanced dataset would take some of the time spent on adapting to data, and move it towards learning from it.

4.2. **Evaluation Metrics.** Three metrics are most often used to assess the results, namely Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), and the Learned Perceptual Image Patch Similarity (LPIPS), yet they might neglect the human intuition with respect to the goal. Notable about the first two would be that the PSNR centers around the MSE, while SSIM is defined using three factors: luminance, contrast, and structural similarity. In the case of LPIPS, it learns the similarity using deep neural network activation function values.

An alternative to these metrics, recently highlighted, is the use of the Patch-based Contrast Quality Index (PCQI), and the Underwater Image Quality Measure (UIQM). Nevertheless, when the human intuition is the next in line, our recommendation is to have a prior empirical study, and an open mind, as they are designed to address colorization efficiency, and not data compression loss. PCQI accounts for the mean luminosity, change in contrast, and structural distortion, while UIQM requires no reference image, and measures sharpness, colorfulness, and contrast.

Despite all the effort, having a person assessing the colorization results remains the golden standard at the moment, as mathematical observations may miss important aspects. A test involves a number of correspondents answering whether they think that the photography they see was colorized or is the original one. Out of the total amount of trials, a fooling rate is determined, accompanied by the probability that an observation occurred by chance.

5. Conclusions and future work

The work presented in this paper sets the grounds for further colorization initiatives. We initially explored whether data driven colorization may achieve human level accuracy, and discovered that there are cases when it is possible. Even alone, the fact that colorization optimizes time costs, and reduces manual labor allows the general public to relive moments from their collection of old photographs. Moreover, the tasks deriving from colorization have even wider implications. Even if this challenge is governed by the absence of a dedicated dataset, and the tendency to borrow techniques from image compression, the generative models, and even the more straight forward convolutional neural network can achieve impressive results.

The gap formed by the semantically complex images, will, in time, be closed through optimizations specific to computational photography. The work of Antic et al. [1], and Zhang et al. [52] would be our recommendation as a model development gateway. As for solving the open problems, enough room was left for improvement in areas such as color leaks, color normalization, conservative predictions, as well as the resolution constraints. Making the colorization models more accessible to the general public, and improving on the existing approaches are the milestones we set for ourselves in the future.

Acknowledgments

The project development relied on the High Performance Computing Center from the Babeş-Bolyai University, Cluj-Napoca, Romania.

References

[1] Antic, J. jantic/deoldify: A deep learning based project for colorizing and restoring old images (and video!). `github.com/jantic/DeOldify`, accessed on Dec 4, 2020.

[2] Ardizzone, L., Lüth, C., Kruse, J., Rother, C., and Köthe, U. Guided image generation with conditional invertible neural networks, 2019.

[3] Bahng, H., Yoo, S., Cho, W., Park, D. K., Wu, Z., Ma, X., and Choo, J. Coloring with words: Guiding image colorization through text-based palette generation, 2018.

[4] Baldassarre, F., Morín, D. G., and Rodés-Guirao, L. Deep koalarization: Image colorization using cnns and inception-resnet-v2, 2017.

[5] Caesar, H., Uijlings, J., and Ferrari, V. Coco-stuff: Thing and stuff classes in context, 2018.

[6] Cao, Y., Zhou, Z., Zhang, W., and Yu, Y. Unsupervised diverse colorization via generative adversarial networks, 2017.

[7] Chen, J., Shen, Y., Gao, J., Liu, J., and Liu, X. Language-based image editing with recurrent attentive models, 2018.

[8] Cheng, Z., Yang, Q., and Sheng, B. Deep colorization, 2016.

[9] Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. The cityscapes dataset for semantic urban scene understanding, 2016.

[10] Deshpande, A., Lu, J., Yeh, M.-C., Chong, M. J., and Forsyth, D. Learning diverse image colorization. `github.com/aditya12agd5/divcolor`, 2017.

[11] Dynamichrome. Showcase. `dynamichrome.com`, accessed on Dec 4, 2020.

[12] El Helou, M., and Süsstrunk, S. BIGPrior: Towards decoupling learned prior hallucination and data fidelity in image restoration. *arXiv preprint arXiv:2011.01406* (2020).

[13] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results, 2012.

[14] Gross, R., Matthews, I., Cohn, J., Kanade, T., and Baker, S. Multi-pie. In *2008 8th IEEE International Conference on Automatic Face Gesture Recognition* (2008), pp. 1–8.

[15] Guadarrama, S., Dahl, R., Bieber, D., Norouzi, M., Shlens, J., and Murphy, K. Pixcolor: Pixel recursive colorization, 2017.

[16] He, M., Chen, D., Liao, J., Sander, P. V., and Yuan, L. Deep exemplar-based colorization, 2018.

[17] Hu, R., Rohrbach, M., and Darrell, T. Segmentation from natural language expressions, 2016.

[18] Iizuka, S., Simo-Serra, E., and Ishikawa, H. Let there be color!: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. *ACM Transactions on Graphics 35* (07 2016), 1–11.

[19] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. Image-to-image translation with conditional adversarial networks, 2018.

[20] Kiani, L., Saeed, M., and Nezamabadi-pour, H. Image colorization using generative adversarial networks and transfer learning. In *2020 International Conference on Machine Vision and Image Processing (MVIP)* (2020), pp. 1–6.

[21] Kodak. Chronology of film. `www.kodak.com/en/motion/page/chronology-of-film`.

[22] Krizhevsky, A. Learning multiple layers of features from tiny images. Tech. rep., 2009.

[23] Krizhevsky, A., Nair, V., and Hinton, G. Cifar-10 dataset. `https://www.cs.toronto.edu/~kriz/cifar.html`.

[24] Krizhevsky, A., Sutskever, I., and Hinton, G. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems 25* (01 2012).

[25] Kumar, M., Weissenborn, D., and Kalchbrenner, N. Colorization transformer. `github.com/google-research/google-research/tree/master/coltran`, 2021.

[26] Larsson, G., Maire, M., and Shakhnarovich, G. Colorization as a proxy task for visual understanding. `github.com/gustavla/self-supervision`, 2017.

[27] Larsson, G., Maire, M., and Shakhnarovich, G. Learning representations for automatic colorization. `github.com/gustavla/autocolorize`, 2017.

[28] Li, Y., Zhuo, J., Fan, L., and Wang, H. J. Cys: Chinese youth subculture dataset. `https://github.com/tezignlab/subculture-colorization/tree/main/CYS_dataset`.

[29] Li, Y., Zhuo, J., Fan, L., and Wang, H. J. Culture-inspired multi-modal color palette generation and colorization: A chinese youth subculture case, 2021.

[30] Liao, J., Yao, Y., Yuan, L., Hua, G., and Kang, S. B. Visual attribute transfer through deep image analogy, 2017.

[31] Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., and Dollár, P. Microsoft coco: Common objects in context, 2015.

[32] Loreto, V., Mukherjee, A., and Tria, F. On the origin of the hierarchy of color names. *Proceedings of the National Academy of Sciences of the United States of America 109* (04 2012), 6819–24.

[33] Manjunatha, V., Iyyer, M., Boyd-Graber, J., and Davis, L. Learning to color from language. `github.com/superhans/colorfromlanguage`, 2018.

[34] Markle, W., and Hunt, B. Coloring black and white signal using motion detection. *Canadian Patent Nr 1291260* (01 1988).

[35] Nazeri, K., Ng, E., and Ebrahimi, M. Image colorization using generative adversarial networks. *Lecture Notes in Computer Science* (2018), 85–94.

[36] Nilsback, M.-E., and Zisserman, A. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing* (Dec 2008).

[37] Perez, E., Strub, F., de Vries, H., Dumoulin, V., and Courville, A. Film: Visual reasoning with a general conditioning layer, 2017.

[38] Royer, A., Kolesnikov, A., and Lampert, C. H. Probabilistic image colorization. `github.com/ameroyer/PIC`, 2017.

[39] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. Imagenet large scale visual recognition challenge, 2015.

[40] Santhanam, V., Morariu, V. I., and Davis, L. S. Generalized deep image to image regression. `github.com/venkai/RBDN`, 2016.

[41] Schanda, J. *CIE 1931 and 1964 Standard Colorimetric Observers: History, Data, and Recent Assessments*. Springer New York, New York, NY, 2016, pp. 125–129.

[42] Su, J.-W., Chu, H.-K., and Huang, J.-B. Instance-aware image colorization. `github.com/ericsujw/InstColorization`, 2020.

[43] Timofte, R., Agustsson, E., Gu, S., Wu, J., Ignatov, A., and Gool, L. V. Div2k dataset: Diverse 2k resolution high quality images.

[44] Tyleček, R., and Šára, R. Spatial pattern templates for recognition of objects with regular structure. In *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, pp. 364–374.

[45] Vitoria, P., Raad, L., and Ballester, C. Chromagan: Adversarial picture colorization with semantic class distribution, 2020.

[46] Wang, L., Guo, S., Huang, W., Xiong, Y., and Qiao, Y. Knowledge guided disambiguation for large-scale scene classification with multi-resolution cnns. *IEEE Transactions on Image Processing 26*, 4 (Apr 2017), 2055–2068.

[47] Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., and Torralba, A. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2010), pp. 3485–3492.

[48] Xiao, Y., Zhou, P., and Zheng, Y. Interactive deep colorization with simultaneous global and local inputs, 2018.

[49] Xu, Z., Wang, T., Fang, F., Sheng, Y., and Zhang, G. Stylization-based architecture for fast deep exemplar colorization. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 9360–9369.

[50] Yang, Z., Liu, H., and Cai, D. On the diversity of realistic image synthesis. `github.com/ZJULearning/diverse_image_synthesis`, 2017.

[51] Zhang, R., Isola, P., and Efros, A. A. Colorful image colorization, 2016.

[52] Zhang, R., Zhu, J.-Y., Isola, P., Geng, X., Lin, A. S., Yu, T., and Efros, A. A. Real-time user-guided image colorization with learned deep priors, 2017.

[53] Zhao, J., Han, J., Shao, L., and Snoek, C. G. M. Pixelated semantic colorization, 2019.

[54] Zhou, B., Khosla, A., Lapedriza, A., Torralba, A., and Oliva, A. Places: An image database for deep scene understanding, 2016.

[55] Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., and Oliva, A. Learning deep features for scene recognition using places database. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1* (Cambridge, MA, USA, 2014), NIPS'14, MIT Press, p. 487–495.

Department of Computer Science,, Faculty of Mathematics and Computer Science,, Babeş-Bolyai University, Kogălniceanu no. 1

Email address: `aaic2261@scs.ubbcluj.ro`

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXVI**, Number 2, 2021
DOI: 10.24193/subbi.2021.2.04

DEEP REINFORCEMENT LEARNING FROM SELF-PLAY IN NO-LIMIT TEXAS HOLD'EM POKER

TIDOR-VLAD PRICOPE

Abstract. Imperfect information games describe many practical applications found in the real world as the information space is rarely fully available. This particular set of problems is challenging due to the random factor that makes even adaptive methods fail to correctly model the problem and find the best solution. Neural Fictitious Self Play (NFSP) is a powerful algorithm for learning approximate Nash equilibrium of imperfect-information games from self-play. However, it uses only crude data as input and its most successful experiment was on the in-limit version of Texas Hold'em Poker. In this paper, we develop a new variant of NFSP that combines the established fictitious self-play with neural gradient play in an attempt to improve the performance on large-scale zero-sum imperfect-information games and to solve the more complex no-limit version of Texas Hold'em Poker using powerful handcrafted metrics and heuristics alongside crude, raw data. When applied to no-limit Hold'em Poker, the agents trained through self-play outperformed the ones that used fictitious play with a normal-form single-step approach to the game. Moreover, we showed that our algorithm converges close to a Nash equilibrium within the limited training process of our agents with very limited hardware. Finally, our best self-play-based agent learnt a strategy that rivals expert human level.

1. Introduction

Learning by interacting with a certain environment (or emulator) has its roots in the way human brain evolved, or how natural intelligence advances [1]. We can consider a game as a simulation of our real world with its own set of rules and features. Some games resemble real-world problems on a smaller scale which means that solutions can provide an intuition for tackling real applications such as financial trading, traffic control, airport and network

Received by the editors: 1 June 2021.
2010 *Mathematics Subject Classification.* 68T05 .
1998 *CR Categories and Descriptors.* I.2.1 [**Artificial Intelligence**]: Learning – *Applications and Expert Systems - Games.*
Key words and phrases. Artificial Intelligence, Computer Poker, Adaptive Learning, Fictitious Play, Self-Play, Deep Reinforcement Learning, Neural Networks.

security, routing ([2], [3], [4]). Most of these real-world games involve decision making with imperfect information and high-dimensional information state spaces.

We have experienced the quick advancement of super-human Awe in perfect-information games like Chess and Go (AlphaGo Zero, [5]; LeelaChessZero [6]), but researchers have yet to reach the same progress in imperfect-information games (AlphaStar, [7]). An optimal theoretical solution to these games would be a Nash equilibrium i.e. a strategy no one can gain extra profit by deviating from it.

Fictitious play [8] is a popular method for achieving Nash Equilibria in normal-form (single-step) games. Fictitious Self-Play (FSP) [9] extends this method to extensive-form (multi-step) games. Neural fictitious Self-Play (NFSP, [10]) combines FSP with neural network function approximation. It is an effective algorithm and the first end-to-end reinforcement learning system that learns approximate Nash Equilibrium in imperfect information games without prior knowledge. It uses anticipatory dynamics; the agents choose their strategies from a mixture of average (supervised learning network) and greedy responses (Q-learning network).

With all of that said, it was proven that NFSP provides poor performance in games with large-scale search space and search depth [11], because it uses only crude data as input and its core aspect is represented by a Deep Q-Network which is offline; it doesn't make any real-time computations during the game. Solutions to these problems were proposed (MC-NFSP, [11]) that use Monte Carlo Tree Search instead. This, indeed, provides better and more stable performance but we are interested in a pure neural approach not using any brute force search methods. As we are going to apply this algorithm mainly to Poker, a game where intuition is key in winning, exhaustive search might not always be necessary. In this paper, we address this issue by adding real-time heuristics as features to the agents' field of view and by combining anticipatory dynamics with neural gradient play which yields, in theory, incremental better response search for our strategies. We test that in practice as well using as benchmark the performance against a certain common opponent.

Many AI bots have proven themselves to be above any human in no-limit Hold'em (Libratus [12], Pluribus [13]) but this does not mean that the game is completely solved. For that, we need a mathematical way of showing that the agent will definitely win money, given a certain interval of time or games, which was actually done with Cepheus [14] for the in-limit version. No-limit variant of Texas Hold'em is still considered unsolved in different formats to this day. In this paper, for the main agent we develop, we do provide a mathematical underpinning for the algorithm behind it, in the context of a 2-player zero-sum game; this is later empirically validated through the experiments in which we successfully approach Nash Equilibrium.

Furthermore, this paper also highlights a direct comparison to some of our previously developed agents. For this, we refer to our previous published paper on this matter: A View on Deep Reinforcement Learning in Imperfect Information Games [15].

We empirically evaluate the agents in heads up computer poker games and explain how an agent trained this way can work even in a multiple-player scheme with some performance loss. As input, we use raw data, as an image of cards from the current visible board combined with two hand-crafted scalar inputs: hard coded rankings of card combinations and Monte-Carlo heuristics for assessing an approximate strength of the opponent hand. The best agent built (with our modest hardware) learnt a strategy close to human expert play.

2. Background

There are two main theoretical parts this research project is based upon - fictitious self-play in extensive-form games and reinforcement learning [1] . In this chapter, we aim to provide some mathematical underlying that is going to be referenced in the main chapters.

2.1. **Reinforcement learning.**

Reinforcement learning (RL) [1] is widely considered as the third paradigm of learning where an environment is fundamentally defined and there are agent(s) that interact with it having a certain goal in mind. Hence, reinforcement learning can be viewed as a tool of solving optimization problems; these are usually modelled as a Markov Decisiion Process (MDP) [1]. Usually, in RL, optimization algortihms makes use of sequential experience. This is a form of history of states and actions that each agent possesses. Appropriately, it is modelled as transition tuples: (r): $(s_t, a_t, r_{t+1}, s_{t+1})$. The goal is to maximize the rewards. To represent that, an *action-value function* Q is used - defined as the expected gain of taking action a in state s and following the policy π: $Q(s,a) = \mathrm{E}^{\pi}\left[G_t | S_t = s, A_t = a\right]$. Here, $G_t = \sum_{i=t}^{T} R_{i+1}$is a random variable of the agent's cumulative future rewards starting from time t [1]. Ideally, we would want to follow the action that gives the highest estimated value Q,that's why *Q-learning* [21] was introduced as a way to learn this greedy policy and replaying past experience. In order to approximate the *action-value function* (or any function for that matter), a wide and deep enough neural network can be employed which seems to be the preferred way nowadays of using Q-learning for solving more complex games: *deep Q network (DQN)* [16].

2.2. **Neural Fictitious Self-Play.** Neural Fictitious Self-Play [10] is a model of learning approximate Nash Equilibrium in imperfect-information games using deep learning.

At each iteration, the agents choose their best response (greedy strategy) with a DQN and update their average strategy by supervised learning through a policy network. That is done by storing datasets of each agent's experience in self-play as transition tuples $(s_t, a_t, r_{t+1}, s_{t+1})$ in a memory M_{RL} (designed for RL) and by storing agent's own behavior (s_t, a_t) in a memory M_{SL} (designed for supervised learning). If we set the self-play sampling in a way that an agent's reinforcement learning memory approximates data of an MDP defined by the other players' average strategy profile, then we can be sure that we find an approximate best response from an approximate solution of the MDP by reinforcement learning.

As we can see, the respective data necessary to train the neural networks through backpropagation is collected within the simulated games during the training process which is offline so it naturally has problems in on-policy games where we need to sample opponents' changing strategy while we play. To see how we can improve on this and take more into consideration the opponents' ever-changing strategies, we need to look deeper at how NFSP uses anticipatory dynamics [17] to stabilize the convergence around Nash Equilibrium points.

Define $\Delta\left(n\right)$ as a standard simplex in R^n, $v_i \in \Delta\left(n\right)$ being the i-th vertex and let $\mathrm{H} : \mathrm{Int}\left(\Delta\left(n\right)\right) \to R$ the entropy function $\mathrm{H}\left(p\right) = -p^T \log\left(p\right)$. In a two-player game, each player chooses its strategy $p_i \in \Delta\left(m_i\right)$, $m_i \in N^*$ and accumulates its reward according to the value-function: $V_i\left(p_i, p_{-i}\right) = {p_i}^T M_i p_{-i} + \tau \cdot \mathrm{H}\left(p_i\right)$, where$-i$, $i \in \{1, 2, ..., n\}$ refers to the complementary set$\{1, 2, ..., i-1, i+1, ...n\}$ [17] and M_i is the game-dependent reward matrix. Consequently, we can define player i's best response as a function $\beta_i : \Delta\left(m_{-i}\right) \to \Delta\left(m_i\right)$,$\beta_i\left(p_{-i}\right) = \arg\max\ V\left(p_i, p_{-i}\right)$ and player i's average response until step k in the game as empirical frequencies $\pi_i\left(k\right) : N \to \Delta\left(m_i\right)$ of player P_i, [17].

In our previous work, we defined the differnt time abstractization of Fictitious Play (FP). Recall that in continuous time FP, we need to consider the derivative of the policy change over time:

$$\frac{d}{dt}\pi_i = \beta_i\left(\pi_{-i}\left(t\right)\right) - \pi_i\left(t\right), i = \overline{1, 2} \qquad (2)$$

Poker falls in this type of abstraction, in which each player has access to the derivative of his empirical frequency $\frac{d}{dt}\pi_i$. The strategy at moment t can be defined as:

$$p_i\left(t\right) = \beta_i\left(\pi_{-i}\left(t\right) + \eta\tfrac{d}{dt}\pi_{-i}\left(t\right)\right),\ \eta \text{ positive parameter } (3)$$

We interpret this formula as a player choosing his best response based on current opponent's average strategy profile combined with a possible change of it that may appear in the future [15].
The authors of the study that we have used to borrow these mathematical notations (*anticipatory dynamics of continuous-time dynamic fictitious play* [17]) prove that for a good choice of η, the stability in Nash equilibrium points can be improved. Of course, this choice of η is game-dependent. The challenge that comes with it though is the fact that the derivative cannot be directly measured and needs to be approximated or reconstructed by empirical frequencies measurements [15].
Recall the equation (3), subtracting π_i from both sides and using (1) yields:

$$\frac{d}{dt}\pi_i = \beta_i\left(\pi_{-i}\left(t\right) + \eta\frac{d}{dt}\pi_{-i}\left(t\right)\right) - \pi_i\left(t\right) (4)$$

In *NFSP* [10], the authors chose a discrete time approximation of the derivative: ${\beta^{t+1}}_i - {\pi_i}^t \approx \frac{d}{dt}{\pi_i}^t$ which, if substituted in (4) yields:

$$p_i\left(t\right) \approx \beta_i\left(\pi_{-i}\left(t\right) + \eta\left(\beta_i\left(\pi_{-i}\left(t+1\right)\right) - \pi_{-i}\left(t\right)\right)\right) \Leftrightarrow$$

$$p_i\left(t\right) \approx \beta_i\left(\left(1-\eta\right)\pi_{-i}\left(t\right) + \eta\beta_i\left(\pi_{-i}\left(t+1\right)\right)\right)$$

That's how the authors reach the combined policy method: $\sigma \equiv \left(1-\eta\right)\hat{\pi} + \eta\hat{\beta}$ which was empirically porved to be successful for games like in-limit Texas Hold'em Poker.
However, a discrete time approximation does have its limitations, that is why we suggest using an approach that borrows elements from *dynamic gradient play* [17] in order to approximate the derivative taking into consideration the opponents' average strategies as well.

3. Developing the agents

We are going to address the technical details and the main process of building the self-play agents mentioned in the introduction. It is important to recall our last published research article on this subject, *A View on Deep Reinforcement Learning in Imperfect Information Game* [15] because we will use some of the agents developed there for direct comparison with the new ones. Only a short introduction of each one will be provided as for more details we recommend reading the original paper.

3.1. **Agent 1 (previously developed)** [15].

This first agent is a reinforcement learning free one, we built it as our own mini remake version of Loki [18] featuring betting decisions with card heuristics and opponent-modelling.

We constructed this agent mainly as an expert system at its core with heuristics for betting decisions and opponent-modelling for exploitations [15]. For opponent modelling, this agent uses 2 classifiers: a naïve Bayes classifier (to replicate the Bayesian analysis presented in the Loki paper) and a deep neural network with a CNN architecture, the input being represented as an image of the current board state alongside some scalar associated features.

3.2. **Agent 2 (previously developed)** [15]**.** This deep reinforcement learning agent learnt to play Poker by training with **Agent 1** from scratch. Its strategy of play combines the greedy strategy β offered by the action-value function with the average strategy π obtained though supervised classification. The second agent managed to learn Poker training with the first agent trying to consistently beat him, treating the opponent as part of the environment.
Therefore, it uses 3 neural networks. First, a *DDQN* system [19] with a *value network* $Q\left(s, a \left|\theta^{Q}\right.\right)$ for predicting the Q values for each action based on data from M_{RL}. It trains through backpropagation using the *Bellman equation* with future Q values obtained through a *target network* $Q'\left(s, a|\theta^{Q'}\right)$. Secondly, we use a *policy network* $\Pi\left(s, a \left|\theta^{\Pi}\right.\right)$ to define our agent's average response based on data from M_{SL}. We choose our main policy σ from a mixture of strategies:$\beta = \varepsilon - greedy\left(Q\right)$ and $\pi = \Pi$: $\sigma \equiv \left(1-\eta\right)\hat{\pi} + \eta\hat{\beta}$, $\eta \in (0, 1]$. This actually represents the same approximation of *anticipatory dynamics* in *discrete time fictitious play* used in NFSP [10], but here we are using it to define our agent in a one-player game, we are not trying to approximate a Nash Equilibrium in this context. The other differences come from the model architectures, inputs and from how often we use each strategy of play to sample games. Moreover, unlike NFSP, we mainly considered a Poker game iteration to be just a hand of play here and reset the main policy accordingly.

3.3. **Agent 3 (our proposed approach in this paper).** Compared to the other two, the third agent, the main focus of this paper, shall decipher poker playing against itself using a new variant of fictitious self-play that employs deep learning.

To clarify, this agent will be based on self-play only, using deep neural nets, without any external help from other players for training and without brute force, real-time exhaustive search. This agent will learn by playing with itself, from scratch, both constantly trying to achieve better rewards. Below (figure 1), we can see the architecture of this self-play system and how the strategies are generated.

Like the Agent 2, we are devising the greedy and average strategies, this time through self-play, though, but we also have a reference to the opponent's

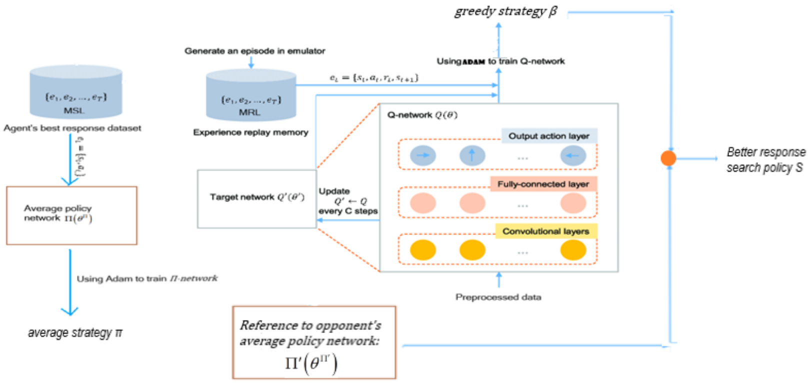

greedy strategy β
Generate an episode in emulator
Using ADAM to train Q-network
$\{e_1, e_2, \ldots, e_T\}$
MSL
Agent's best response dataset
$\{e_1, e_2, \ldots, e_T\}$
MRL
Experience replay memory
$e_t = \{s_t, a_t, r_t, s_{t+1}\}$
Q-network $Q(\theta)$
Output action layer
Fully-connected layer
Convolutional layers
Better response search policy S
$e_i = \{s, a\}$
Average policy network $\Pi(\theta^{\Pi})$
Target network $Q'(\theta')$
Update $Q' \leftarrow Q$ every C steps
Using Adam to train Π-network
Preprocessed data
average strategy π
Reference to opponent's average policy network: $\Pi'(\theta^{\Pi'})$

FIGURE 1. Agent 3, self-play system architecture

average strategy to construct a better response search. To understand how this is mathematically done, take the gradient of the value function:

$$\nabla V_i\left(p_i, p_{-i}\right) = M_i p_{-i}$$

We are interested in the differential equations system that defines the dynamic gradient play:

$$\tfrac{d}{dt}\pi_i = \mathrm{P}_\Delta\left[\pi_i\left(t\right) + M_i \pi_{-i}\left(t\right)\right] - \pi_i\left(t\right) \text{ with } i = \overline{1,2},$$

where $\mathrm{P}_\Delta : R^n \to \Delta\left(n\right)$ is the projection on the simplex $\Delta\left(n\right)$: $\mathrm{P}_\nabla\left[x\right] = \underset{s \in \Delta(n)}{\arg\min} \left|x - s\right|$.

Therefore, we can obtain a parametrized approximation of $\frac{d}{dt}\pi_i$ using two forms of behavioral evolution of strategy of play in FP (DT – discrete time FP, GP – gradient play). Using the definition, we get:

$$\frac{d}{dt}\pi_{-i} = \frac{\pi_{-i}\left(t+\eta\right) - \pi_{-i}\left(t\right)}{\eta} \approx \pi_{-i}\left(t+1\right) - \pi_{-i}\left(t\right) \overset{(4)}{\Rightarrow}$$

$$\frac{d}{dt}\pi_i + \pi_i\left(t\right) = \beta_i\left(\pi_{-i}\left(t+1\right)\right) \overset{(*)}{\approx} {\beta_i}^{t+1}; i = \overline{1,2}\,(5)$$

Let $S\left(t\right) \in \Delta\left(n\right)$ such that $S\left(t\right) = \mathrm{P}_\Delta\left[\pi_i\left(t\right) + M_i \pi_{-i}\left(t\right)\right]$ i.e.

$$\left|\pi_i\left(t\right) + M_i \pi_{-i}\left(t\right) - S\left(t\right)\right| < \varepsilon \text{ with } \varepsilon \text{ as small as possible.}$$

Then it follows that:

$$\frac{d}{dt}\pi_i + \pi_i\left(t\right) = S\left(t\right).$$

Combining this with (5) yields that for every $\rho \in [0,1]$ we have:

$$\frac{d}{dt}\pi_i \approx \rho\left({\beta_i}^{t+1} - \pi_i\left(t\right)\right) + \left(1-\rho\right)\left(S\left(t\right) - \pi_i\left(t\right)\right), i = \overline{1,2}.$$

Substituting now $\frac{d}{dt}\pi_i$ in (3), we get the final formula:

$$p_i\left(t\right) = \beta_i\left(\left(1-\eta\right)\pi_{-i}\left(t\right) + \eta\left(\rho \cdot {\beta_i}^{t+1} + \left(1-\rho\right) \cdot S\left(t\right)\right)\right)$$

which means our agent can choose their actions from a mixture of strategies:

$$\sigma \equiv \left(1-\eta\right)\hat{\pi} + \eta\left(\rho\hat{\beta} + \left(1-\rho\right)\hat{s}\right).$$

The motivation behind this choice is that the evolution of the GP strategy follows a better response search, adjusting the strategy of play in the direction of the gradient from the empirical frequencies of the opponent. Thus, using this form, especially in a game with imperfect information, where the best answer is harder to find, it is important that we don't stagnate and we always try to find a better solution than the current one (and if we have already found the best solution then the gradient should suggest so).

We want to favor finding the best response though, that is why are going to set theρ parameter to be:

$$\rho \approx 1-\eta+\varepsilon \text{ with } 0 < \varepsilon < 2/100.$$

Below, we present *Algorithm 1*, the main algorithm that agent 3 uses to get learn Poker from self-play.

Algorithm 1 — Agent 3, reinforcement learning (self-play) agent with fitted Q-learning

```
for 1:no_games do
    Initialize new game $G$ and execute agent via RUN_AGENT for each player in the game
end for
function RUN_AGENT(G)
    Initialize replay memories $M_{RL}$ (circular buffer) and $M_{SL}$ (own behaviour reservoir)
    Initialize average-policy network $\Pi(s,a|\theta^{\Pi})$ with random weights $\theta^{\Pi}$
    Iniitalize opponennt average-policy network $\Pi'(s,a|\theta^{\Pi'})$ with random weights $\theta^{\Pi'}$
    Initialize action-value network $Q(s,a|\theta^{Q})$ with random weights $\theta^{Q}$
    Initialize target network with weights $\theta^{Q'} \leftarrow \theta^{Q}$
    Initialize parameters $\eta, \rho$.
    for each episode do
```

$$S_{i,t} = \begin{cases} \begin{cases} \epsilon - greedy(Q) & \text{w/ prob } \rho \\ S = P_{\Delta}Pi + Q_{extended(m_i,m_i)} \cdot \Pi' & \text{w/ prob } 1-\rho \end{cases} & \text{w/ prob } \eta \\ \\ \Pi & \text{w/ prob } 1-\eta \end{cases}$$

```
    Observe initial information state $s_1$ and reward $r_1$
    for t=1,min_replay_memory_size do
        Sample action $a_t$ from policy $\sigma$
        Execute action $a_t$ in emulator and observe reward $r_{t+1}$ and next information state $s_{t+1}$
        Store transition $(s_t, a_t, r_{t+1}, s_{t+1})$ in reinforcement learning memory $M_{RL}$
        if agent follows best response policy $\sigma = \beta(= \epsilon - greedy(Q))$ then:
            Store behaviour tuple $(s_t, a_t)$ in supervised learning memory $M_{SL}$
        Update $\theta^{\Pi}$ with gradient descent on loss
            $L(\theta^{\Pi}) = E_{(s,a)\sim M_{SL}}[KLDivergence\Pi(s,a|\theta^{\{Pi\}})]$
        Update $\theta^{Q}$ with gradient descent on loss
            $L(\theta^{Q}) = E_{(s,a,r,s')\sim M_{RL}}[(r + max_{a'}Q(a',a'|\theta^{Q'}) - Q(s,a|\theta^{Q}))^2]$
        Periodically update target network parameters $\theta^{Q'} \leftarrow \theta^{Q}$
    end for
end function
```

We are using 3 deep neural networks: a DDQN [19] system to approximate the action-value function and a policy network to approximate the player's own average behaviour. The architecture for these neural nets for the two strategies (greedy and average) are the same. The input is represented by a *17x17x9* 3D array containing the images of the last two board states and the scalar features that we mentioned the *Developing The Agents* section – note that this is the same input as the one Agent 2 uses. As we said in [15], we add the last board state to the input because of the inspiration from *AlphaGo Zero* [5] interpreting it as an *attention mechanism.* The actual architecture of the networks is represented as a CNN with 4 layers of convolution. 2 MaxPooling and 1 fully connected as hidden layers. For the reinforcement learning part, we use *MSE* as loss (together with the Bellman equation to calculate the value of a state to get the predicted part). For the policy network we use *Kullback–Leibler Divergence* between two probability distributions as it is usually a good loss measurement, also used by the creators of AlphaGo Zero.

4. Experiments

The computer code is available at: link (backup directory for the whole project). Everyone can play against the agents at request at: poker.ptidor.com.

We are mainly testing the algorithm on heads-up no-limit variant of the game of Poker. The choice of heads-up is also determined by the limited resources of this research project. For evaluation, we are going to measure the performance of each agent against previously developed ones and some generic players that we previously defined in [15]. We also paired the final agent against a human player to get an intuition of its level of play in real world.

4.1. **General specifications.** The format we are using for the games is heads-up, no-limit with **100** chips as starting stack and **5** chips small blind. To evaluate the agents, we use two metrics: ***average stack*** over a fixed number of games and ***mbb/h*** (milli big blinds per hand) [15]. A mili big blind per hand is 1/1000 of a big blind, if a player wins a big blind it gets 1000 points, if a player wins a small blind it gets 500 points (and it loses the same amounts for the negative case). So, a player that always folds is expected to lose at a rate of 750 mbb/h – we obtain this figure by taking the mean over the big and small blinds. Therefore, the intuition is that the values for a mbb/h metric will usually stay in the interval [-750, 750]. This metric is a standard for Poker research nowadays and many other studies ([10], [13], [14], [11]) make use of it. It is regarded that a human professional player would aim for winnings of **50 mbb/h**, at a minimum.
For comparison reasons, we use a couple of generic Poker players:

(1) A player that only calls *(Call_player)*
(2) A player that chooses its actions randomly: 3 times out of 5 calls and in the remaining it can equally raise with a random amount or fold *(Random_player)*
(3) A player that chooses its actions based only on Monte-Carlo simulations and not look-up tables (*HeruristicMC_player*)

4.2. **No-limit Texas Hold'em Poker.** We want our self-play agent to be unbeatable in the long run, so now an episode will be represented by a game (which can have several hands) and not an only hand of play as we considered in Agent 2. Also, *Agent 3* will receive an immediate reward of **0** for each move and only at the end of a hand / end of a game, he will receive a non-zero reward depending on how many chips it won. Thus, Agent 3 will not be penalized immediately for a raise of 100 (all-in), for example, but if he loses that hand, then he will receive a reward of negative 100 at the end of it, which is very high. In this way, we tell the AI that it doesn't matter what moves he chooses as long as the reward at the end of the game is maximized.

We let the algorithm train for roughly 3 days straight (80 hours to be exact). For compute, we used an *NVIDIA Tesla T4 Workstation* with *32*GB of RAM and a *NVIDIA GTX 1050ti* with *16*GB of RAM. However, at inference, the artificial players can be run on day-to-day hardware.

The algorithm descendance to Nash-Equilibrium can be observed in figure 2. Parameters η and ε were set to 0.1, 0.9, respectively, ρ was set to 0.92, max length for M_{RL} to 200k and for M_{SL} at 1m. We make one stochastic gradient update of mini-batch size of 256 per network for every 64 steps and the target network parameters were reset every 1000 updates.

The choice of the hyper-parameters (apart from ρ) was inspired by the NSFP paper [10]. Little effort was put into experimenting with hyper-parameter search because of time constrains and the fact that similar hyper-parameters already existed within the NSFP context. However, note that even in this paper (NSFP), the choice of hyper-parameters wasn't clearly reasoned. The architecture of the neural networks was not explored in this paper but it was inspired by standard image classification neural networks.

In order for the copies of the same agent to be in Nash-Equilibrium, we have to observe a convergence towards 0 of the difference in modulus in winnings (mbb/h - aggregated over a batch of recent games) of the two players. This is actually what we plot in figure 2 and as we can see, that measurement value narrows down and starts to approach 0 after the 250's batch. Note that we calculate the mean of the absolute difference in winnings over the most recent 500 games for the y-axis in the figure. That's why we have 300 iterations for 150k games.

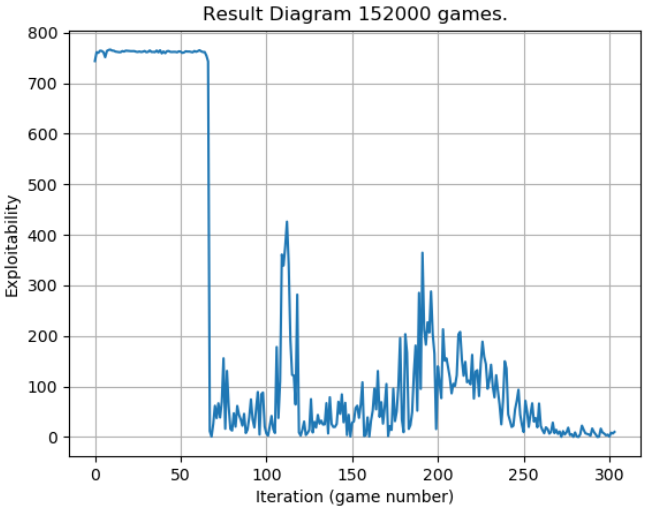


Figure 2. Training evolution (mbb/h) of Agent 3 (ρ= 0.92), with hand-crafted metrics as input, in 3 days straight.

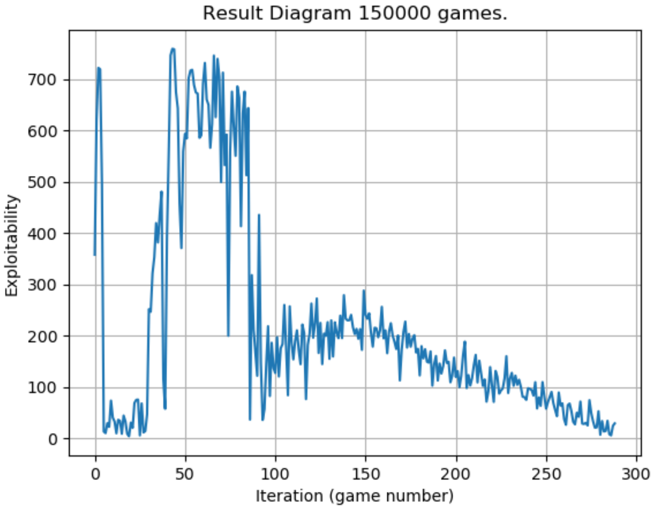


Figure 3. Training evolution (mbb/h) of Agent 3 (ρ= 0.92), without hand-crafted metrics as input, in 3 days straight.

There are obvious spikes that disturb the balance as we can see around the 100s and 200s iteration, this is because both copies are continuously learning by playing one another and it is possible that one learnt a clever strategy faster and it is able to exploit that for a brief moment. Of course, one can argue that huge spikes like these can appear again if we let it train for more iterations. This is possible; however, it is unluckily - note that for the last approximatively 50 iterations (or 25,000 games) the mean absolute difference in mbb/h stayed steady in the range 0 to 20, which wasn't the case until then. Moreover, this range is good enough to call this an approximation of the Nash-Equilibrium because if we recall the critical value 50 mbb/h that a professional player usually aims to achieve in a match, everything below that would still be considered indecisive.

We also trained the algorithm with raw data, without hand-crafted input metrics, just like in NFSP [10], to see if the algorithm still converges without any prior knowledge of the domain (figure 3). And if so, how does it compare to the version above in which we are actually using solid prior knowledge of the game?

After the same amount of training time, it seems the algorithm still converges to approximate Nash-Equilibrium, but slower than our main proposed version. We base this claim on the range of the y-axis values for the last 50 iterations. It also concludes a little smaller number of games in 72 hours. This experiment does seem to suggest that hand-crafted metrics do really help a self-play algorithm train better.

4.2.1. *Experimenting with an expert Poker player.* For this experiment, I've invited a semi-professional human Poker player, *Serban.* He is very experienced with the game, playing constantly on real high money stakes but lacks the tournament play.

He played **56** hands against our agent, from *figure 2*, (during a 10-game match) and the results were crushing. our agent recorded winnings of **241.07 mbb/h** with the final score **7-3**.

The human player said he was very impressed with the style of play of our agent but he recognized some mistakes during the match regarding the preflop stage of the game, which can be very costly during a professional match. Mainly, the agent does not recognize very weak cards in the preflop, such as 7-3, at which point he should not call for a raise.

A temporary solution could be a Monte-Carlo search, which immediately draws attention to very weak combinations of cards at any stage of the game. Indeed, this version is still not perfect, or close to perfect, but training on more iterations should strengthen our AI bot considerably. It is an important victory, though, all things considered.

4.2.2. *Comparison with NFSP and other artificial Poker players.*

Results using greedy + average strategy against Agent 2				
Player	No. hours trained	No. Games Played	Final Average Stack	Winnings (mbb/h)
Agent3_GP	6	250	117.83	263.37
Agent3_GP	11	250	117.48	318.51
Agent3_GP	17	250	117.1	338.82
Agent3_GP	45	250	110	340.18
Agent3_DP	11	250	120.71	318.93
Agent3_DP	17	250	115.43	309.82
Agent3_DP	45	250	99.2	192.55
Agent3_GP	6	1000	118.45	248.35
Agent3_GP	17	1000	116.54	356.17
Agent3_GP	45	1000	111.9	361.08
Agent3_DP	11	1000	116.42	299.03
Agent3_DP	45	1000	102.3	234.22

Table 1. Results of different versions of Agent 3 vs Agent 2.

Since we want to test the effect of that better response search through gradient play proposed in the theoretical part, we will analyze the behavior / performance of an Agent 3 trained against a copy of itself taking into account the policy $\hat{s}$, $(\rho < 1)$ and the behavior performance of an Agent 3 trained against a copy of itself without regard to the policy $\hat{s}$, $(\rho = 1)$, as of *Algorithm 1*. We will therefore call these two agents: **Agent3_GP**, **Agent3_DP**, from gradient play, discrete play respectively (which refers to the method used to approximate the CDP derivative). Note that **Agent3_DP** is a theoretically a replica of NFSP.

We tested (table 1) multiple versions of these agents against Agent 2 (the one that beat an amateur human player). In this match-up, it is easier to see the difference between the two versions of Agent 3. In 250 matches played against Agent 2, both variants won, but the one that uses better response search exceeds the threshold of 320 mbb / h, and the situation improves when we increase the number of iterations. For some reason, the performance of Agent3_DP decreases at 45 hours compared to less trained versions. This trend remains consistent for the experiment with 1000 games as well, in which Agent3_GP reaches over 360 mbb/h in winnings but Agent3_DP can't cross 300mbb/h.

We need to mention that Agent3_GP took 6h to train for 50k games, whilst Agent3_DP took 11h, that's why we have no measurement for Agent3_DP for less than 11 hours. Note that Agent3_GP consistently beats Agent 2, in both

experiments (250 and 1000 games, respectively), this makes Agent 3 take the status of the best agent developed so far, after only **6** hours of **self-play**!
We have repeated the experiments many times to assure the consistency of the results, there is a statistical error of around +/- 4.5 in terms of average stack and around +/- 40 for mbb/h for the 250 games case. These figures get roughly halved for the more stable experiments with 1000 games played.

It is important to clarify that this does not necessarily mean that Agent3_DP is definitely worse; however, we have established in the introduction section that such a benchmark will be used to draw interpretations. Agent 2 is the previous best agent we have developed that can rival amateur human play [15], so it is a decent artificial opponent for these 2 agents. It is good practice to evaluate poker bots against each other as we can make use of a bigger amount of sample games (compared to matches against humans) and we can also compute statistical significance.
Out of curiosity, we paired up Agent3_GP after 30 hours of training against Agent 1. The results are not surprising at all, getting a win rate of **88.46%** and an average stack of **175** after **130** games against the expert system with neural opponent modeling.

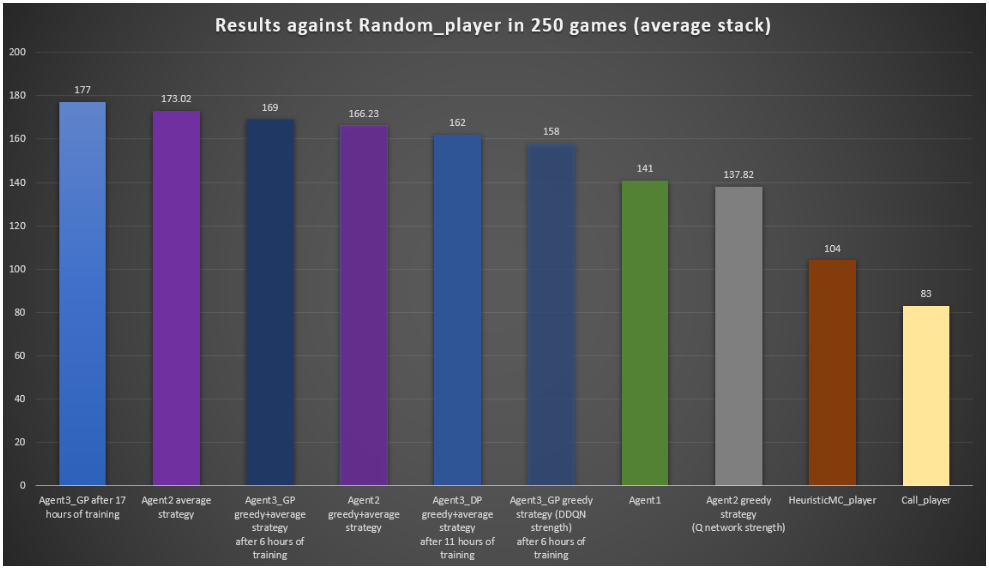


FIGURE 4. Results of some previous players against Random_player compared to Agent 3; statistical error +/-4.5

Next up, we compared the performance of Agent3_GP and Agent3_DP against the Random_player. Both versions are the ones trained on 50k games - this is an important threshold as both agents seem the overpower all the other ones after crossing this limit.
In figure 4, we can observe the performance of most of the agents we have tested during our study (against the Random_player). It is clear that both Agent3_GP and Agent3_DP crush in the benchmarks, however, Agent3_GP reaches almost 180 average stack in 250 games, which hasn't been done by any of our agents until now. This is another bonus point for the better response search technique that Agent3_GP uses.
What is very impressive here (figure 4) is the fact that we used the version of Agent 2 that trained with Random_player, having as sole objective to defeat it. Although Agent 3 had n**o interaction** with Random_player, learning the game of poker only through self-play, he achieves a performance almost identical to that of Agent 2, even surpassing the performance of all the other deep reinforcement learning agents, after just **17 hours** of training!

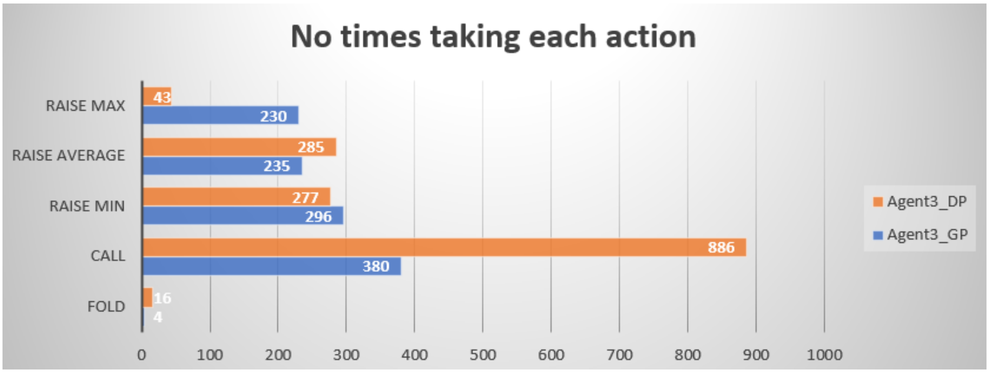


FIGURE 5. Agent 3 play style in 250 games vs Random_player

This match-up was also an opportunity to study the differences between Agent3_GP's style of play and Agent3_DP's. Agent3_DP plays much safer and is much more reserved about a raise, mainly choosing to wait through calls, very rarely choosing to go all-in (figure 5). Instead, Agent3_GP is much more aggressive, bouncing back between calls (predominant action) and raises.
The proposed approach can be adapted to play a multi-player Poker game. Although it may lose performance compared to the heads-up variant, we can make a small change in the inputs that are fed to the predict function to get the next action. The only input components that we use, relevant to a multi-player game, are the average estimated opponent strength, which can be recalculated with respect to the number of players through Monte-Carlo

simulations and the opponent's stack which can be replaced with the average stack of all the opponents.

5. Discussion and further research

Although the results looked pretty successful, it is very hard to correctly assess the level of play of the best agents. Until we test them against a professional player or top computer programs like *Hyperborean*, We can't know for sure that they are indeed at top human level. Furthermore, due to time and hardware constrains, we couldn't experiment on more iterations, we can maybe descend even more closer to a Nash-Equilibrium in optimal conditions. Improvements can also be made regarding the format of the game. All the agents were trained in heads-up, no-limit, 100-100 starting stack with 5 small blind formats, but for more general play, it is recommended to consider the small blind as percentage of the starting stack.

6. Conclusions

We have showed the power and utility of deep reinforcement learning in imperfect information games and we have developed an alternate new approach to learning approximate Nash equilibria from self-play that does not use any brute force search and only relies on the *intuition* provided by deep neural networks. When applied to no-limit Hold'em Poker, training through self-play drastically increased the performance compared to fictitious play training with a normal-form singe-step approach to the game. The experiments have shown the self-play agent to converge reliably to approximate Nash equilibria with crude data and limited hand-crafted metrics as input and the final artificial player can rival expert human play.

References

[1] Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT press, 2018.

[2] Lambert Iii, Theodore J., Marina A. Epelman, and Robert L. Smith. "A fictitious play approach to large-scale optimization." Operations Research 53.3 (2005): 477-489.

[3] Nevmyvaka, Yuriy, Yi Feng, and Michael Kearns. "Reinforcement learning for optimized trade execution." Proceedings of the 23rd international conference on Machine learning. 2006

[4] . Urieli, D. and Stone, P. (2014), "Tactex'13: a champion adaptive power trading agent." In Proceedings of the 13th International Conference on Autonomous Agents and Multi-Agent Systems, pages 1447–1448.

[5] Silver, David, et al. "Mastering the game of go without human knowledge." nature 550.7676 (2017): 354-359.

[6] Gary Linscott, "Leela Chess Zero", 2018.

[7] Arulkumaran, Kai, Antoine Cully, and Julian Togelius. ”Alphastar: An evolutionary computation perspective.” Proceedings of the Genetic and Evolutionary Computation Conference Companion. 2019.

[8] Brown, George W. ”Iterative solution of games by fictitious play.” Activity analysis of production and allocation 13.1 (1951): 374-376.

[9] Heinrich, Johannes, Marc Lanctot, and David Silver. ”Fictitious self-play in extensive-form games.” International Conference on Machine Learning. 2015.

[10] Heinrich, Johannes, and David Silver. ”Deep reinforcement learning from self-play in imperfect-information games.” arXiv preprint arXiv:1603.01121 (2016).

[11] Zhang, Li, et al. ”Monte Carlo Neural Fictitious Self-Play: Approach to Approximate Nash equilibrium of Imper-fect-Information Games.” arXiv preprint arXiv:1903.09569 (2019).

[12] Noam Brown, Tuomas Sandholm, “Safe and Nested Subgame Solving for Imperfect-Information Games”, 2017. 31st Con-ference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

[13] Noam Brown1, Tuomas Sandholm, “Superhuman AI for multiplayer poker”, 2019. Brown et al., Science 365, 885–890

[14] Bowling, Michael, et al. ”Heads-up limit hold’em poker is solved.” Science 347.6218 (2015): 145-149.

[15] Pricope, T.V.. A View on Deep Reinforcement Learning in Imperfect Information Games. Studia Universitatis Babeș-Bolyai Informatica, [S.l.], v. 65, n. 2, p. 31-49, dec. 2020. ISSN 2065-9601.

[16] Mnih, Volodymyr, et al. ”Playing atari with deep reinforcement learning.” arXiv preprint arXiv:1312.5602 (2013).

[17] Jeff S. Shamma and Gurdal Arslan, “Dynamic Fictitious Play, Dynamic Gradient Play, and Distributed Convergence to Nash Equilibria.”, 2005.

[18] Denis Richard Papp, “Dealing with Imperfect Information in Poker.”, 1998.

[19] Van Hasselt, Hado, Arthur Guez, and David Silver. ”Deep reinforcement learning with double q-learning.” Proceedings of the AAAI conference on artificial intelligence. Vol. 30. No. 1. 2016.

[20] Yakovenko, Nikolai, et al. ”Poker-CNN: A pattern learning strategy for making draws and bets in poker games using con-volutional networks.” Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 30. No. 1. 2016.

[21] Watkins, C. J. and Dayan, P. “Q-learning”, 1992. Machine learning, 8(3-4):279–292.

The University of Edinburgh, School of Informatics, 10 Crichton St, Newington, Edinburgh EH8 9AB, United Kingdom

Email address: T.V.Pricope@sms.ed.ac.uk

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXVI**, Number 2, 2021
DOI: 10.24193/subbi.2021.2.05

AUTOMATIC FACE SHAPE CLASSIFICATION VIA FACIAL LANDMARK MEASUREMENTS

ALEXANDRU-ION MARINESCU

Abstract. This paper tackles the sensitive subject of face shape identification via near neutral-pose 2D images of human subjects. The possibility of extending to 3D facial models is also proposed, and would alleviate the need for the neutral stance. Accurate face shape classification serves as a vital building block of any hairstyle and eye-wear recommender system. Our approach is based on extracting relevant facial landmark measurements and passing them through a naive Bayes classifier unit in order to yield the final decision. The literature on this subject is particularly scarce owing to the very subjective nature of human face shape classification. We wish to contribute a robust and automatic system that performs this task and highlight future development directions on this matter.

1. Introduction

Of the major areas of application of the topic of face shape classification, we will mention the most prominent two: hairstyle or eye-wear *recommender systems* and forensic analysis of human subjects, by complementing *3D facial reconstruction.*

Recommender systems are first and foremost an important marketing tool and a major revenue source for the fashion and entertainment business sectors. They seek, aided through computing processing power, to mimic the way the potential customer thinks, by keeping track of the products she/he finds interesting. To put it simply, they create a psychological profile of the customer, attuned for the target product category. There exist a plethora of recommender system types: some are trained for music or movie recommendation, based on music genre (i.e. classical, pop, jazz, rock) or movie category

Received by the editors: 15 September 2021.

2010 *Mathematics Subject Classification.* 68T45.

1998 *CR Categories and Descriptors.* I.2.10 Computing Methodologies [**ARTIFICIAL INTELLIGENCE**]: Vision and Scene Understanding – *Modeling and recovery of physical attributes* .

Key words and phrases. artificial intelligence, computer vision, naive Bayes, face shape, facial landmarks.

(i.e. horror, drama, comedy, action) preferences, whilst others such as the ones employed by major online stores, attempt to track what the end client would be interested in buying next. Another possible application of a face shape classifier would be in the field of forensic analysis. Here, for example, a suspect's face shape could serve as a hash check for fast querying against a police database of known criminals.

Nevertheless, our particular focus in this paper will be on a hairstyle and eye-wear recommender system. More specifically, we will discuss the implementation of face shape classifiers, which serve as the basic building block of such an application. Face shape recognition has become very useful in many computer vision applications. So, an algorithm to classify the face shape correctly is needed. There can be issues if the images are not of good quality and have pose variability. We aim to distinguish seven types of face shapes: oval, round, rectangle, square, heart, diamond and triangular (see Figure 1). The face shape is to be analyzed from the frontal/neutral pose. Cancelling the yaw, pitch and roll of the subject's face has been discussed previously in [3, 7]. In the following sections we will describe how this can be accurately achieved using a combination of facial landmark measurements in the standard 68-landmark model (Figure 2) and train a naive Bayes classifier in order to yield the final decision regarding the user's face shape.

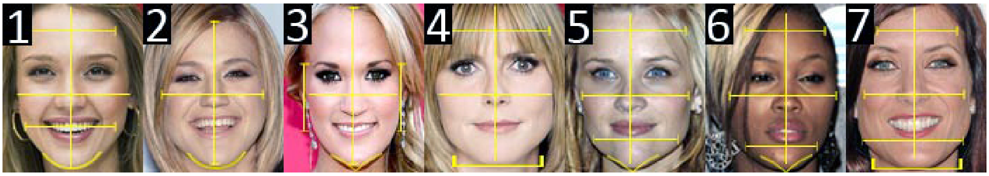


Figure 1. The 7 generally acknowledged face shapes, in reading order from left to right: oval, round, rectangle, square, heart, diamond and triangle (*thehairstyler.com*).

2. State of the art

The subject of face shape classification is a difficult one mainly due to the fact that determining the face shape of a human is very subjective and open to interpretation. In general, a person does not belong strictly to one of the seven classes of shapes, but instead, possesses a combination of at least two principal shapes. At most, what we can say is that a person has "*predominantly*" the facial traits of a certain category. As a consequence, a standardized face shape classification is yet to be developed. Some sources suggest fewer face

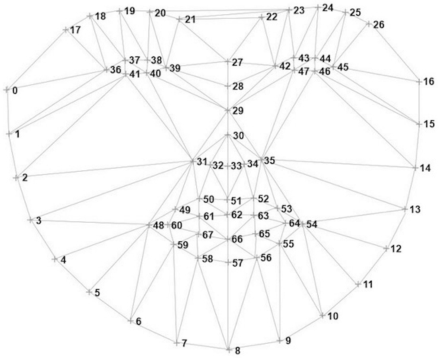


FIGURE 2. The 68 landmark-based face model, which serves as input for our face shape classifier, as defined by the DLIB [5] computer vision toolbox.

shape categories, considering that statistically poorly represented classes can be merged with more dominant ones.

The authors of [1] present a novel idea for face shape classification based on three techniques: facial region similarity, correlation and fractal dimensions. Their experiments demonstrate that the proposed approach based on the first technique, namely facial region matching gives effective results for face shape classification. It relies on determining the intersection over union (IOU) between the contour of a human subject's face and an idealistic version of each of the face shape classes. In [8], the authors propose a full pipeline which takes data in the form of a neutral pose image of a female subject, passes it through a classifier to obtain a good estimation of the face shape and finally yields the most appropriate hairstyle recommendation. The core of their pipeline is the VGGNet [12] deep learning classifier architecture, which was successfully combined with feature concatenation and was subjected afterwards to fine-tuning.

The authors of [14] have designed a face shape classifier based on convolutional neural networks (CNN), which they claim is a first in literature. All approaches until their time of writing relied on linear discriminant analysis (LDA), support vector machines (SVM) with different kernel functions, or multi-layer perceptrons (MLP). Their research was driven by the fact that they could refresh these existing techniques using deep learning. Concretely, they employed transfer learning, and retrained the final layer of an Inception v3 architecture [13], thus being able to achieve an accuracy of $\approx 84\%$. Another major contribution from the authors is the creation of their own manually labeled data set, which was made publicly available. Their data consists of 500 images of celebrities for which the face shape is known. There are 100 images per face shape class (heart, oblong, oval, round and square). There is however a trade-off here: there are multiple images of the same celebrity throughout the data set, so instead of having 100 images of distinct individuals per shape class, the data actually contains about 8-10 celebrities per class. Additionally, the authors have tried to bring the subjects in the images to a neutral pose, but one can only cancel the image roll (in the case of 2D images), still leaving the pitch and yaw unresolved.

All the approaches discussed so far are based on 2D images. However, substantially more information regarding the human face shape can be extracted provided we have a full vertex-based model of the face (see Figure 3). Such an approach is discussed in [4], where instead of computing landmark Euclidean 2D distances via a landmark detector, they compute the local deformation of the face in a given basis. They conclude that their proposed method achieves better results than existing methods on extracting the traits of the human face.

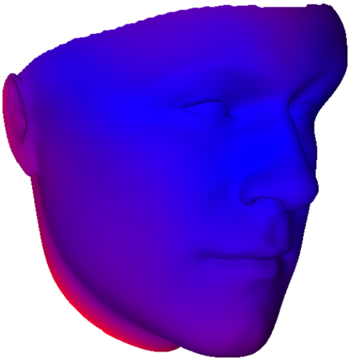


Figure 3. Example of 3D heat map visualization of local face vertex deformations versus a standardized, average human face model. A red-shift indicates pronounced deformation, whereas a blue-shift indicates a close match.

3. Proposed approach

The face shape is an important factor in selecting the shape of the eyeglasses; although it is quite difficult to objectively determine the face shape, in the *visagisme* community the following face shapes are generally accepted: rectangle, round, square, heart, diamond, triangle and oval. The rules for determining one's face shape are numerous and leave a lot of space for interpretation, as they involve measuring some features of the face and determining if one measurement is "larger" than another. But all the existing methods rely on measuring the widest part of the face, the height of the face and determining the jaw shape. To automatically measure the face shape we first need to accurately segment the face area and then mimic the measurements required to compute the face shape.

The main difficulty in this task is related to the forehead area, as there are multiple occlusions (hair, bangs, accessories etc.) present in this area and it is quite difficult to determine the boundary between the skin and the hair area. This boundary is required to measure the height of the face (one of the most discriminative measurements when deciding upon the face shape), as well as for the forehead width measurement.

For the shape segmentation we used the same U-Net architecture [9] employed to segment the hair area, as described above, with an off-the-shelf facial landmark detector. To estimate the area of the lower face region, we combined the output of the DLIB [5] facial landmark detector with the segmentation mask. For the forehead estimation, we selected 5 boundary points on the hair segmentation mask, and estimated a symmetrical contour, as depicted in Figure 4.

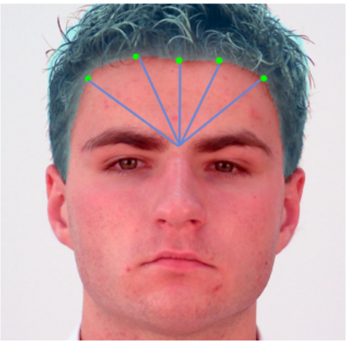


Figure 4. The 5 boundary points on the hair segmentation mask, spaced equally at 30 degree angles: 30, 60, 90, 120, 150 degrees, respectively.

The procedure of face segmentation has been already approached and documented, and is available to the general public in the package provided by [5]. On top of this, in order to improve the quality of the classification, we bring our original contribution which derives from a deep learning approach for hair segmentation [2], out of which the forehead line can be extracted, and the facial contour now becomes complete. Finally we extrapolate the full contiguous face contour by merging the face and hair segments, by means of a construct known as *line iterator* (please refer to Figure 5).

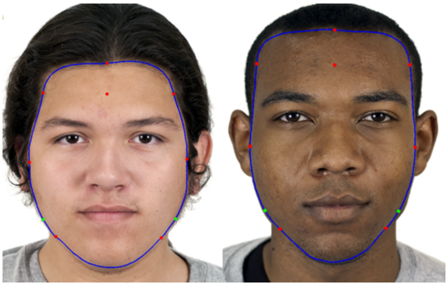


Figure 5. The contiguous face contour obtained from the merged face and hair masks.

We determined a set of common, relevant landmarks and measurements that should be used in the classification process. These landmarks are pinpointed in Figure 6: (1) a point in middle of the forehead area, (2) and (10) two extreme points situated to the left and right of the middle forehead point, (3) and (9) two points that determine the largest width of the face, (4) and (8) two points around the jaw, (5) and (7) two points that determine the chin width and (6) the lowest middle point of the chin.

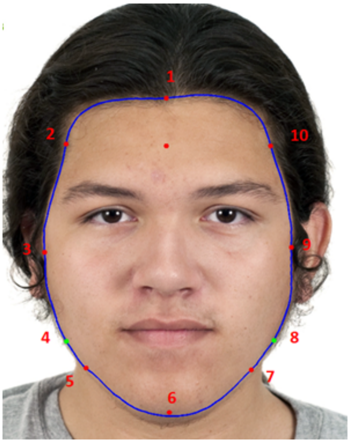


Figure 6. The landmarks used in face shape classification.

The metrics of interest from the contour and internal facial landmarks are listed below:

- face rectangularity; this relies on the minimum bounding rectangle (MBR); the MBR is a standard relationship used to measure the rectangularity of a shape, and it is defined as the ratio of the area of a region to its minimum bounding rectangle [10]; the face MBR is obtained by computing the MBR for the entire face contour

- middle face rectangularity; the MBR of the contour determined by the points (3), (4), (8) and (9)
- forehead rectangularity; the MBR of the contour determined by the points (3), (9) and (1)
- the chin angle, measured between the points (5), (6) and (7)
- ratio between the lower face width over the middle face width ($RBot$)
- ratio between the upper face width over the middle face width ($RTop$)
- the difference between $RTop$ and $RBot$
- the ratio between the width and the height of the face (fAR)

4. Experimental results

One of the most popular classifiers and also one of the fastest to prototype and train is the naive Bayes classifier [11]. It owes its simplicity to the assumption that every pair of features to be classified is independent of each other. Experimentally, we train a naive Bayes classifier by starting from the Chicago [6] face database (annotated with the face shape tag), on top of which we add 290 images (a morph between existing contour and the corresponding contour template for each face shape type). The details regarding the employed data subset are as follows: **604** total train data set samples, **115** total test data set samples, with a train/test scheme of **85/15**, for which the naive Bayes classifier yields an accuracy of **85%**. As a post-processing step, after we obtain the decision from the naive Bayes classifier, we apply the following post rules of classification, obtained through *empirical experimentation* (here $class_1$ is the class predicted with the highest probability, and $class_2$ is the class predicted with the second-highest probability, respectively). This has been done in an attempt to rectify the misclassification of outliers. Ideally, provided a consistent and balanced data set, these rules should be reconsidered.

```
if class_1 is ”Square” and class_2 is ”Rectangle” and width / height > 0.75
    then ”Rectangle”
if class_1 not ”Round” and class_2 is ”Square” and width / height > 0.75
    then ”Square”
if class_1 is ”Oval” and class_2 is ”Round” and width / height > 0.75
    then ”Round”
if class_1 is ”Oval” and class_2 is ”Rectangle” and forehead MBR > 0.85
    then ”Rectangle”
if class_2 is ”Triangle” and RBottom − RTop > 0.10
    then ”Triangle”
```

The naive Bayes classifier was one of the candidates for our training, the other being support vector machines (SVMs). In Tables 1 and 2 we give a comparison between the naive Bayes classifier and the SVM classifier, on our data set of choice.

Naive Bayes				
Shape	Precision	Recall	F1-score	Support
diamond	1.00	0.60	0.75	5
heart	0.92	0.75	0.83	16
oval	0.86	0.90	0.88	21
rectangle	0.87	0.93	0.90	29
round	0.78	0.88	0.82	16
square	0.83	0.88	0.86	17
triangle	0.80	0.73	0.76	11
accuracy			**0.85**	115

Table 1. Results for the naive Bayes classifier, following training and testing. The accuracy *without* the post-processing step is **0.83**.

SVM				
Shape	Precision	Recall	F1-score	Support
diamond	0.60	0.60	0.60	5
heart	0.85	0.69	0.76	16
oval	0.67	0.76	0.71	21
rectangle	0.81	0.86	0.83	29
round	0.75	0.75	0.75	16
square	0.93	0.82	0.87	17
triangle	0.55	0.55	0.55	11
accuracy			**0.76**	115

Table 2. Results for the support vector machine classifier, following training and testing. The accuracy *without* the post-processing step is **0.73**.

Although the SVM hyper-parameters (*kernel type*, with choices between "*linear*", "*poly*" - polynomial, "*rbf*" - radial basis function or "*sigmoid*"; regularization parameter - "*C*" and kernel coefficient - "*gamma*") were thoroughly explored using a grid search of available values, still the naive Bayes classifier proves significantly more accurate and was the preferred choice during the face shape application deployment.

As far as future development is concerned, we target the creation of a unified dataset and benchmark for face shape classification, since this is the most important milestone in achieving accurate face shape classification. Currently, in our setup, we hand-picked and manually annotated images which we considered to be representative for their corresponding face shape class. This was done to minimize the bias between two closely related face shapes (such as "diamond" and "heart") and to enforce the robustness of the naive Bayes classifier. In the future we wish to augment our data set with the one supplied by [14].

5. Conclusions

The currently available face shape estimation module makes several assumptions: first of all, it assumes that the person depicted in the image has a near frontal pose. Secondly, as it relies on images, implying 2D projections of the human face, it is quite difficult to extract information about the depth related measurements, such as the length of the jawline. To address this issue, we plan to develop a library to compute a 3D model of the subject's face. We will insist two approaches: one relying on multi-view geometry, while the other using LIDAR data. Once the model of the face is precisely extracted, we can measure all the required distances and angles directly on the 3D model, and therefore develop a classical rule-based algorithm.

Although extracting a 3D face model to estimate the face of the subject can lead to the development of a simple rule based face shape determination algorithm, the problem is that the rules used in face shape determination are highly subjective. Therefore, we envision developing a graph based convolutional neural network model to analyse the relationships between all the relevant facial landmarks and to automatically recognize the face shape.

References

[1] Bansode, N., and Sinha, P. Face shape classification based on region similarity, correlation and fractal dimensions. *IJCSI International Journal of Computer Science Issues 13*, 1 (2016).

[2] Ileni, T. A., Borza, D. L., and Darabant, A. S. Fast in-the-wild hair segmentation and color classification. In *Visigrapp* (2019).

[3] Ion Marinescu, A., Alexandru Ileni, T., and Sergiu Dărăbant, A. A versatile 3d face reconstruction from multiple images for face shape classification. In *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)* (Sep. 2019), pp. 1–6.

[4] Jiang, Z., Wu, Q., Chen, K., and Zhang, J. Disentangled representation learning for 3d face shape. *CoRR abs/1902.09887* (2019).

[5] Kazemi, V., and Sullivan, J. One millisecond face alignment with an ensemble of regression trees. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2014).

[6] Ma, D., Correll, J., and Wittenbrink, B. The chicago face database: A free stimulus set of faces and norming data. *Behavior research methods 47* (01 2015).

[7] Marinescu, A. I., Dărăbant, A. S., and Ileni, T. A. A fast and robust, forehead-augmented 3d face reconstruction from multiple images using geometrical methods. In

2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM) (Sep. 2020), pp. 1–6.

[8] Pasupa, K., Sunhem, W., and Chu Kiong, L. A hybrid approach to building face shape classifier for hairstyle recommender system. *Expert Systems with Applications 120* (11 2018).

[9] Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. *CoRR abs/1505.04597* (2015).

[10] Rosin, P. Measuring shape: Ellipticity, rectangularity, and triangularity. *Machine Vision and Applications 14* (08 2001).

[11] Russell, S., and Norvig, P. *Artificial Intelligence: A Modern Approach*, 3 ed. Prentice Hall, 2010.

[12] Simonyan, K., and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (2015), Y. Bengio and Y. LeCun, Eds.

[13] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. *CoRR abs/1512.00567* (2015).

[14] Tio, A. E. D. Face shape classification using inception v3. In *Electrical and Electronics Engineering Institute, University of the Philippines Diliman, Quezon City, Philippines* (2017).

Dept. of Computer Science, Babeș-Bolyai University, Kogălniceanu 1, 400084 Cluj-Napoca, Romania

Email address: alexandru.marinescu@ubbcluj.ro

STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXVI**, Number 2, 2021
DOI: 10.24193/subbi.2021.2.06

STRATEGIES FOR STUDENT EVALUATION IN THE ONLINE CONTEXT

SANDA-MARIA AVRAM

Abstract. In this paper we conducted an investigation on the performance of the students during the second semester of the academic year 2020-2021. We looked at the performance results obtained by students on the laboratory work, practical and final exams while we were forced by the Covid pandemic to move entirely into an online education system. Our focus was to determine the impact of a consistent behaviour (or lack of it) on the final student performance. We determined that, even in an online setting, a good involvement (in terms of attendance and good performance) guarantees good final results. The investigations were performed using the Formal Concept Analysis, which is a very powerful instrument already used by us in previous research in order to detect student behaviour in using an e-learning portal. Another set of results showed that the change of the final mark computation formula to be based in a higher proportion on the lab work was closer to the actual overall performance of students.

1. Introduction

Nowadays the Covid pandemic forced most of us to migrate to the online educational system. Although in the last decade the online educational systems has shown a rapid development [15], being somehow forced to adopt an only-online education brought to light many challenges and many places where this system needs improvement. It also put everything into perspective and maybe made us to better appreciate the aspects that make traditional education valuable. Online educational systems are the set of techniques, methods and environments that provide access (through Internet) to educational support for students [15]. There are synchronous components (e.g., students may attend live lectures; real-time interaction between educators and students may

Received by the editors: 9 November 2021.

2010 *Mathematics Subject Classification.* 68T30, 68P20.

1998 *CR Categories and Descriptors.* H.3.5 [**Information Storage and Retrieval**]: Online Information Services – *Web based services*; I.2.6 [**Artificial Intelligence**]: Learning – *Knowledge acquisition*; K.3.1 [**Computers and Education**]: Computer Uses in Education – *Distance learning.*

Key words and phrases. student evaluation, formal concept analysis, online learning .

exist with the possibility of instant feedback), and there are also asynchronous components (e.g., students may access teaching material in an any-time, any-place, any-pace manner).

From the teaching perspective, the online instruments used to be considered an extension and support of traditional learning. Authors J. Liebowitz and M. Frank define in [12] the concept of blended learning as "blend" between traditional and online learning. The proportion in which each of the two learning systems is used creates different types of blended learning.

Y. Park compares in [13] a *discussion-based blended learning* model with a *lecture-based blended learning* model. In the first type of learning model, the students are expected to be actively involved in online forums, while in the second type of learning model the main online activities of students are: submitting tasks or downloading materials. The investigation from the data collected in this study show that there is a linear prediction between online activities and student performance, i.e. the total score that they obtain in the case of *discussion-based blended learning course.* However, no linear prediction exists in the case of *lecture-based blended learning course.* Therefore, it is concluded that the type of online activity is important in determining whether the online involvement of students could predict corresponding outcomes.

In our current research, we use standard deviation to see the distribution of marks and Formal Concept Analysis as a technique to discover patterns in the effect that the integral online education system (that was somewhat forced by the Covid pandemic) had over the teaching process and whether the changes we made in our approach had a positive or a negative effect.

The investigation described in this paper was done through an FCA-based analysis performed on the students' activity results during one semester (i.e., 14 weeks) by considering the marks obtained during and at the end of the semester. The considered activities took place during the second semester of the academic year 2020-2021. The rest of the paper is structured as follows: section 2 presents a very short overview of the "forced" online learning during the Covid pandemic. Section 3 presents the theoretical background for the method/instrument/technique we use in our investigation. The motivation of this work as well as some prerequisites and other useful details are described in section 4. Section 5 presents the actual tests and results. The last section (i.e., 6) contains the conclusions and future work.

2. Online learning in the COVID context

There is a consistent body of research based on this topic, especially in these last couple of years due to the Covid pandemic. Among the papers to mention is [2], where the author presents a fivefold perspective (i.e., "strengths, weaknesses, opportunities, and challenges") on what the Covid pandemic crisis brought to the educational system. In terms of *strengths* there is the time and space flexibility, the customisation of the learning process based on the student's needs, the possibility of creating an environment that is collaborative and interactive. In terms of *weaknesses* the author mentions the loss of direct human communication, the technical problems or difficulties (e.g., frustration and confusion generated by the online environment, imature instruments), loss of student's accountability due to the time and space flexibility. The main *opportunity* is the "online learning boom" that forces both the teachers and the students to find and try new technical solutions. This comes with the possibility of acquiring new abilities such as critical thinking, adaptability, resilience. The opportunities are also great for the IT community, which could and do help with developing instruments/programs that are tailored to the online education. Maybe the most numerous and important points are touched upon in terms of *challenges.* Due to this "forced" change from the traditional to the online education, most of the challenges stem from trying to integrate, engage and motivate all the participants in the learning process (e.g., teachers and students). The online environment is not mature yet, therefore there are not as many rules and regulations nor the actual infrastructure as in the case of traditional system. Another important challenge is the additional costs involved with the equipments, trainings and creating online educational content. These costs have also an equity aspect in the sense that not all teachers and/or students have the possibility to acquire the required equipment and/or the Internet connection.

The study presented in [1] investigates the perspective of Pakistani higher education students on the online learning in the pandemic context. The results obtained show that traditional learning is preferred in this case.

Authors of [8] put everything in perspective by discussing the difference between "Emergency Remote Teaching and Online Learning". They begin their investigation stating that online learning seems to be perceived as having a lower quality than traditional (i.e., face-to-face) education, despite the fact that the research shows that that is not always the case. The view that this study lights upon is that the traditional educational system, because it exists for so long, has developed an "ecosystem" (a consistent infrastructure) in which lectures are only one component, thus, similarly, in time, an effective online educational system will require to build its own "ecosystem" in order to support efficient education.

3. Formal Concept Analysis

Formal Concept Analysis (FCA) stemmed from applied mathematics but it is nowadays positioned in the Knowledge Discovery and Representation field. In the formal (dyadic) context, two sets are considered, the first being a set of objects, and the second being the set of attributes. Between the elements of the two sets there is defined a relation that states "object *o* has attribute *a*" [7].

Maximal groups of objects that have the same attributes are determined. Such groups are called *concepts*. All determined concepts form a complete lattice by introducing an order relation between the concepts. The order relation states that between two concepts there is a relation in which one of them is considered *subconcept* and the other is considered *superconcept* if and only if all elements of the set of objects of the *subconcept* are included in the set of objects of the *superconcept* .

In the triadic format, alongside the two sets considered by the dyadic FCA, a third set is introduced that is called *conditions*. Therefore, the relation between the three sets now states "object *o* has attribute *a* under condition *c*" [11].

In the dyadic format, the relation can be represented as an incidence table on which each object takes up a row and each attribute takes up a column. In the triadic form, however, we have a tridimensional representation of the relation between elements of the three sets, so that for each condition an incidence table as the one described for the dyadic form can be used to describe the relation.

A triadic formal concept is also called a *triconcept* and consists of maximal groups of objects that have a specific set of attributes under a specific set of conditions [11].

The set of objects form a formal concept is called *extent*, while the corresponding set of attributes is called *intent* , both in the dyadic and triadic context. The corresponding set of conditions is called *modus* in the triadic context. [11].

There are several tools that have been developed for FCA. The one we used in our investigation is the FCA Tools Bundle [10, 9] as it offers a user-friendly visualisation for contexts and in addition to this, it enables navigation in the triconcepts. The concepts of a dyadic context can be visualized as a concept lattice. Triconcepts cannot be viewed in a concept lattice like in the case of a dyadic context. However, exploring triconcepts can be done by deriving dyadic contexts from them by projecting along one of the dimensions of a triadic concep (i.e., the extent, the intent or the modus).

Our own previous contributions use the FCA instruments in order to investigate into the behaviour of students [6, 4, 5] while using an educational portal called PULSE [3].

4. Defining the problem

The students' activity investigated in this paper was done online. We used Microsoft Teams for video conferencing with students. The laboratory work was evaluated after an interview conducted in a one-to-one (i.e., student-to-teacher) manner. The practical exam was done using Moodle [14] quizzes oriented more on the practical concepts. The final exam consisted also in Moodle quizzes randomly selected from a large set of questions based (in a balanced manner) both on theory and practical aspects. The students involved in the investigation belonged to two sections (with slightly different study objectives) which we are going to refer further on as S1 and S2. During the final exam S1 students were supervised using video on Microsoft Teams and with shared screen on https://meet.jit.si/, while S2 students were supervised only using video on Microsoft Teams. Due to concerns that online examination setting is more prone to cheating possibilities we changed the formula of computing the final mark by decreasing the weight of the exams and increasing the weight of the laboratory work.

As a teacher you would like to convey as much information to students as possible and perfect your methods/skills each year. And as much one would like that all students would acquire the maximum level of information, one would also like to have a fair and an honest evaluation system. One would prefer the marks to reflect as closely as possible the level of information that students have acquired.

We are trying, therefore, to take a closer look at the entire activity of students and more specifically their activity during the semester (i.e. their laboratory work where they have to implement the new concepts in order to solve some given problems) and the practical and final exams.

We analyse the students' marks in order to determine patterns in their behaviour which can have an impact on their overall performance. We considered in our investigation a mandatory subject. The students' evaluation included their marks obtained during the entire semester (14 weeks - in which they were supposed to complete 8 assignments), a practical examination and a final exam.

Table 1 presents the number of the students involved in the investigation. The students study this mandatory subject in their first year of study for students in S1 and in their second year of study for students in S2. During the last week of the semester there is a practical exam, and at the end of the semester there is a final examination.

TABLE 1. Details about the number of students

	Sections		**TOTAL**
	S1	**S2**	
Total number of students	82	107	**189**
First-time enrolled students	79	88	**167**
Re-enrolled students	3	19	**22**
Students who passed	64	80	**144**
Students who failed	18	27	**45**
Promovability rate	78%	75%	**76%**

The students which do not pass the subject during this phase have another chance within a re-examination. All the students which do not pass this subject after re-examination have to re-enroll within one of their next years of study. Therefore, the "Re-enrolled students" from Table 1 are (in this case) students in their third (final) year of study.

In the Romanian education system the marks are given on a scale from 1 to 10 (10 being the maximum), a mark equal or above 5 denotes passing the exam/subject. We are going to coarse the range of the marks following the qualification system applied in our primary educational system, but also in other countries. The four qualifications we use are depicted in Table 2.

TABLE 2. Qualification classes used for student results

Qualification name	**Denoted by**	**Represents marks**
insufficient	i	from 1 to 4
sufficient	s	5 and 6
good	g	7 and 8
very good	vg	9 and 10

We consider that a student has a consistent activity when the marks obtained vary only slightly. In order to determine the consistency of students activity we used the standard deviation applied on the marks they obtained as detailed in the following bullet list. We are analysing these values considering the two thresholds (i.e., $th1$ and $th2$) that demarcate the 3 classes of acceptable (*acc*), *big*, and respectively *too big* values, as depicted in the Table 3:

- *labActivDEV* is the standard deviation for the 8 lab marks (any unhanded laboratory work was marked with 0). The first threshold for this value is 1,5 and the second threshold is 3.

- *semActivDEV* is the standard deviation for the evaluation of the activity done during the semester (lab average and practical exam). The first threshold for this value is 1 and the second threshold is 2. We considered that having here only two values they are proper to be closer and therefore we have lower values for th1 and th2.
- *examsDEV* is the standard deviation for exam results (practical and final exam). The first threshold for this value is 1 and the second threshold is 2.
- *averagesDEV* is the standard deviation for lab average, practical and final exam. The first threshold for this value is 1 and the second threshold is 2.
- *activDEV* is the standard deviation for the entire activity (lab marks, practical and final exam). This means that we have here 10 distinct marks in the standard deviation computation. The first threshold for this value is 1,5 and the second threshold is 3.

Table 3. The use of thresholds to demarcate the classes of standard deviation values

Classes of values	**The use of thresholds to demarcate the classes**
acceptable (acc)	$0 \leq val < \mathbf{th1}$
big	$\mathbf{th1} \leq val < \mathbf{th2}$
too big	$\mathbf{th2} \leq val$

Another aspect that we considered is the attendance. Our faculty enforces a rule that states that students have to have a 90% laboratory attendance in order to be allowed to enter the examination and/or the re-examination. Having not met this rule, any student is considered to have failed the subject. Thus, in our investigation, students that have the *attendance* attribute have met this requirement.

5. Results and Discussions

5.1. **Formal Concept Analysis Results.** First we have built a dyadic formal context, where we considered the students as objects and their lab results and their attendance as attributes. A simplified example of such a context is depicted in the Table 4.

For the simplified example we have 3 objects (the students) and 5 attributes (lab marks qualifications and attendance) and a total of 6 concepts:

```
[
    [["Stud1","Stud2","Stud 3"],[]],
```

Table 4. Example of simplified dyadic context

	lab i	lab s	lab g	lab vg	attendance
Stud1		x			
Stud2			x		x
Stud3				x	x

```
     [["Stud1"],["lab s"]],
     [["Stud2","Stud 3"],["attendance"]],
     [["Stud 3"],["lab vg","attendance"]],
     [["Stud2"],["lab g","attendance"]],
     [[],["lab i","lab s","lab g","lab vg","attendance"]]
  ]
```

A concept is actually all the rectangles in Table 4 which are completed with ‘x’ obtained by moving columns and/or rows. Each such concept is then represented in a concept lattice. The resulting form (i.e. lattice) for this simplified example is depicted in Figure 1. Here, each node represents a concept. That is why, having 6 concepts, the lattice representation has 6 nodes.

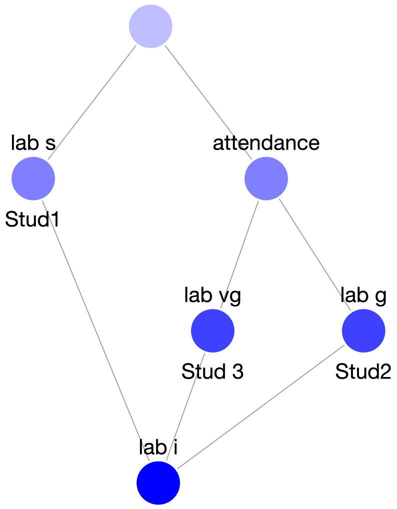


Figure 1. FCA-based lattice representation for the simplified example

The lattice in Figure 1 is read as follows: starting from a node, the objects (which in our case are students) for that node are collected going downwards and the attributes for that node are collected going upwards. The attribute-labels are placed above a node and all nodes below it (directly or indirectly connected with that node but only by following descending arcs/links) have that attribute and object-labels are placed below a node and all nodes above it (directly or indirectly connected with that node but only by following ascending arcs/links) contain that object. Thus, for the node that has the label "attendance" we can see that we have 2 students, i.e. "Stud 3" and "Stud 2", as we go downwards to collect the objects, which in our case are students.

For the node that has the label "Stud2" in order to determine its attributes we go upwards observing that such attributes are "lab g" and "attendance". The label "lab i" placed on the lowest node indicates that no student has that attribute.

The complete data set considered had all of the students (189) and the 5 attributes mentioned above. The resulting lattice is depicted in Figure 2.

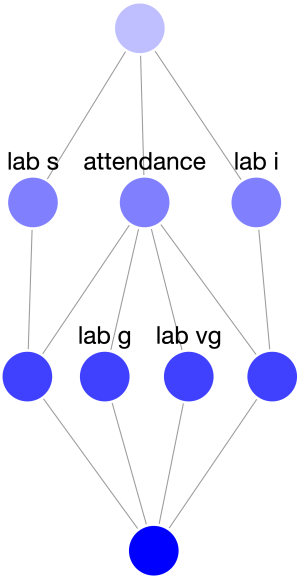


FIGURE 2. FCA-based lattice representation for the first results

We were not able to list all the students as labels below the nodes because the lattice would not be intelligible with a large number of labels under some nodes, but the colour intensity of the nodes denote the number of students (the top node containing all students). As it can be deduced from Figure 2, all the students with good (*lab g*) and very good (*lab vg*) lab marks have attended at least 90% of labs (i.e., have "attendance" attribute), while part of students which lave sufficient (*lab s*) and insufficient (*lab i*) lab performance did not have a satisfactory attendance rate.

That is because, for instance, the node connecting below the nodes with labels "*lab s*" and "*attendance*" represent/contain all the students with sufficient lab performance and an acceptable attendance. There are also students that have a sufficient lab performance but not an acceptable attendance. They are depicted in the node with the label "*lab s*" alongside those with acceptable attendance. The node with the label "*attendance*" contains all the students with an acceptable attendance regardless of their lab performance. We can conclude then that this rule enforced by our faculty has merit and it is justified by the results we have obtained.

5.2. **Triadic Formal Concept Analysis Results.** Next we modelled our data in the form of triadic contexts (G, M, B, I) where the object set G consists of students (as in the dyadic setting), the attribute set M contains activity qualifiers obtained (i.e., i, s, g and vg) while the condition set B contains the activities for which the students obtained the qualifiers, meaning the lab average, the mark for the practical exam, the mark for the final exam and the final mark obtained by the formula:

$$60\% \times lab\_average + 20\% \times practical\_ex + 20\% \times final\_ex$$

A small selection of this triadic context is depicted in Tables 5(A) and 5(B). We have here a $2\times4\times2$ triadic context, the "slices" being labeled by condition names.

There are exactly 7 triconcepts of this context, i.e., maximal tridimensional cuboids full of incidences:

```
[
    [["Stud 2"],["vg"],["practical exam"]],
    [["Stud 2"],["g"],["lab average"]],
    [["Stud 1"],["g"],["practical exam"]],
    [["Stud 1"],["s"],["lab average"]],
    [["Stud 1","Stud 2"],["i","s","g","vg"],[]],
    [["Stud 1","Stud 2"],[],["lab average","practical exam"]],
    [[],["i","s","g","vg"],["lab average","practical exam"]]
]
```

Table 5. Example of triadic context

(a) a. *lab average* condition *slice*

lab average	i	s	g	vg
Stud1		x		
Stud2			x	

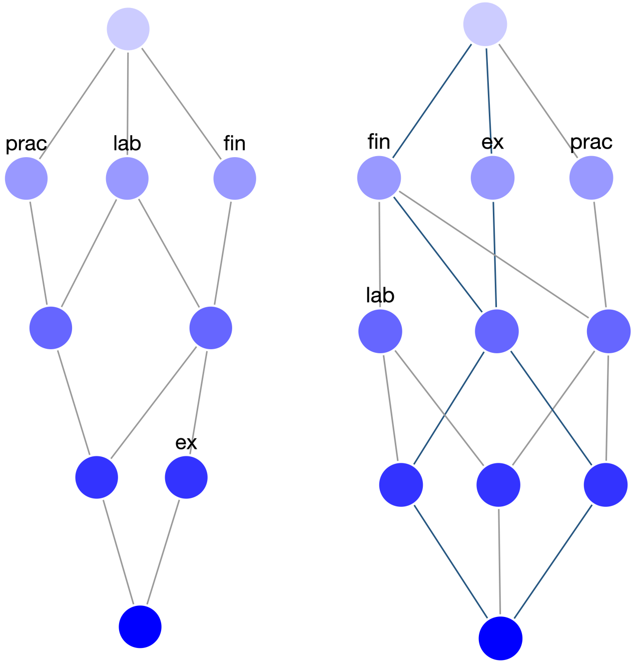
(b) b. *practical exam* condition *slice*

practical exam	i	s	g	vg
Stud1			x	
Stud2				x

The first four of these triconcepts are proper, meaning that they have all the sets (i.e., objects, attributes and conditions) non-empty.

For our tricontexts we considered all students that had at least one activity, eliminating the students which did not submit any laboratory work. Therefore, for section $S1$ we have 67 students (objects) and for $S2$ we have 84 such students.

Having three sets (i.e. objects, attributes and conditions), the representation of a triadic context is tridimensional, and therefore it can be represented as a trilattice, which can be hard to navigate. Therefore, the FCA Tools Bundle allows analysing triadic contexts by projecting along one of the dimensions of a triadic concept (i.e., the extent, the intent or the modus) in order to obtain a dyadic context which can then be represented as a bidimensional lattice which is easier to navigate. This is done starting from one triadic concept and "locking" on one of the dimensions by setting that dimension (i.e., the extent, the intent or the modus) with the set of values within that triconcept. By right-clicking on the nodes (representing concepts) of a lattice obtained in this manner, one can see the triadic concept associated with it. From this point on, one can lock on a different dimension in the triadic concept and generate another bidimensional lattice. One can analyse thus the triadic context by repeating this process. In Figure 3 we locked to see only the very good performances in one or more activities. As it can be seen in Figure 3(A), all of the $S2$ students that had a very good performance on final exam (ex) did also very good during laboratory activity (lab) and also had very good final marks (fin). Moreover, a very good performance within the practical examination ($prac$) reflected in very good final results (fin) only for students that have done very good also in laboratory activity (this is depicted within the lattice by the common node next to the node having the label "ex"). For the students in $S1$ (results depicted in Figure 3(B)) all the students with very good lab performances (lab) had very good final marks (fin). The results for students in $S1$ are different from the results for students in $S2$ considering the perspective that students with very good lab performances (lab) had very good final marks (fin), but not necessarily a very good performance on final exam (ex). That is due to the fact that (as mentioned in Section 4) students in $S1$ were more closely monitored during the exam, and as it usually happens some students do not perfom well under stress.



(A) Results for students in S2

(B) Results for students in S1

FIGURE 3. Results for the students that had a very good performance

From these investigations results a strong correlation between the work involvement of students and their final results.

Another triadic context considered was the one where we have students as objects, classes of values depicted in Table 3 as attributes and the standard deviations described in Section 4 of the paper (i.e., $labActivDEV$, $semActivDEV$, $examsDEV$, $averagesDEV$ and $activDEV$) as conditions. The results showed that there is a strong correlation between $labActivDEV$ and $activDEV$.

Figure 4 shows for instance that $S2$ students with similar class of *labActivDEV* and *examsDEV* had only either *big* or *acc* (acceptable) standard deviations. The label "*too_big*" placed on the lowest node denotes that no student had both *labActivDEV* and *examsDEV* too big. All the students that do not have the two deviations either "*big*" or "*acc*" are contained in the topmost node.

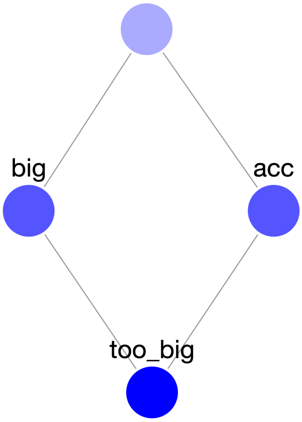


Figure 4. Students in S2 with similar class of labActivDEV and examsDEV

The same scenario as the one depicted in Figure 4 takes place when we group any of the five standard deviations considered and/or if we consider them all.

Other such similarities can be deduced using FCA. But the conclusions drawn from the result showed that general consistency in the students' work generated predictable outcomes.

Finally, we wanted to see how close is our final average (computed as $60\% \times lab\ average + 20\% \times practicalexam + 20\% \times final\ exam$) to the actual average of all evaluations (i.e., 8 lab marks, 1 practical exam, 1 final exam) by computing the standard deviation between the two. Almost all the values obtained were less than 0.5 for this standard deviation, the actual percentages are detailed in Figure 5. We were also interested in how good the decision was to change the previous final average formula (computed as $20\% \times lab\ average + 40\% \times practical\ exam + 40\% \times final\ exam$), denoted as "*prev*" in Figure 5. On the Oy axes we have the percentage of students and on the Ox

axes we have the value of the standard deviation. As it can be seen, the current computation formula for the final mark for both sections reflects better the students' performance than the previous one.

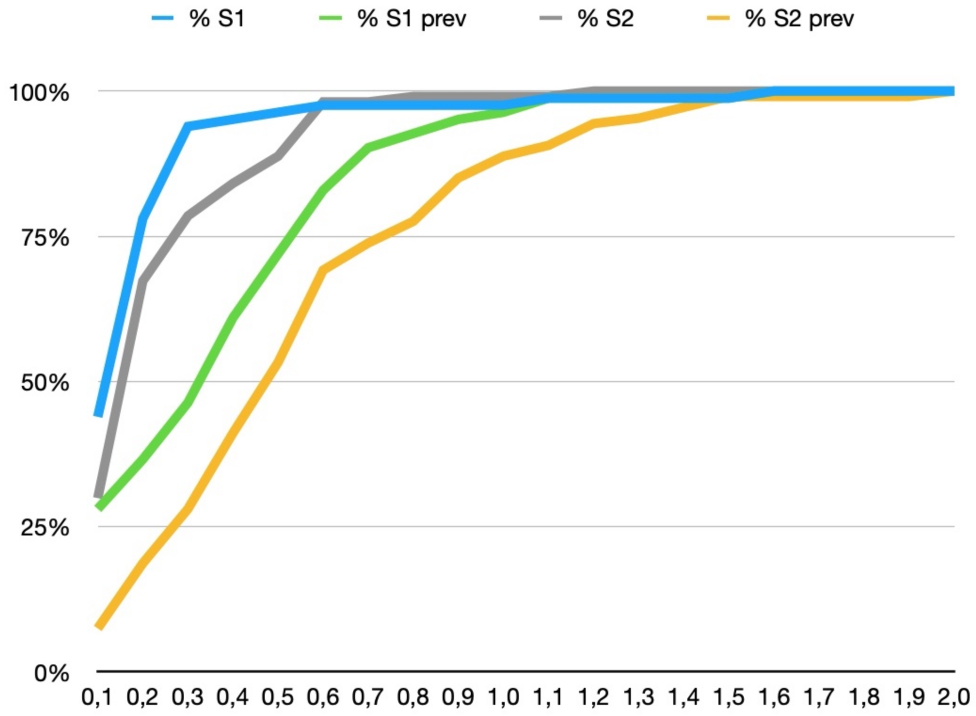


FIGURE 5. Comparison between the previous formula of computing the final mark with the current formula by measuring (through standard deviation) how close are they to the student evaluations

6. Conclusion and Future Work

During this Covid pandemic we were forced to have our activity online. This new arrangement brought new perspectives and new challenges, both of which have as well advantages as disadvantages.

As advantages we can mention:

- faster and easier communication

- more electronic teaching materials which can be consulted at any time - any place - any pace

Disadvantages and challenges that we observed were:

- students are not as focused as in a classroom
- there is non-verbal communication that does not apply in a virtual meeting (for instance an experienced teacher can detect at a glance if the concept presented was understood or more time/effort/examples are required).
- evaluation tests can not be monitored as closely as in a classroom, as there are students (especially in our domain of activity - which is computer science) who are really creative.

In this paper we wanted to address the main question: Did the consistency or inconsistency during the semester (i.e., in the lab work) affected the students performance on the exams (practical exam and/or on the final exam)? In order to answer this question, our investigation was fourfold.

First, we investigated the students' involvement from the perspective of their attendance by using dyadic FCA. The results showed that all students with good and very good lab performance meet the attendance requirements.

Second, we used triadic FCA to see the correlation of students' activities in the case of very good performances. Our results showed that all S1 students with very good lab performance obtained very good final marks, while all S2 students with very good performance in the final exam did also very good in lab and had very good final marks.

Third, we used triadic FCA on standard deviations of students' activities. From this investigation we observed that a standard deviation that we considered to be "too big" cannot be observed consistently though all activities (i.e., lab, practical exam and final exam).

Fourth, we observed that the current formula used to calculate the final mark (changed during the Covid imposed online period) gives a better appreciation of the students' performance than the previous one (used in the traditional face-to-face setting).

The work presented here was mainly focused on good and very good performances. Further investigations can be focused on sufficient and insufficient performance to try to determine other causes for such results.

Moreover, in order to address the disadvantages mentioned above, we would like to conduct a comparison with pre-Covid results and further on maybe to post-Covid ones.

References

[1] M. Adnan and K. Anwar, *Online Learning amid the COVID-19 Pandemic: Students' Perspectives*, Online Submission, vol. 2, no.1, 2020, pp. 45-51.

[2] S. Dhawan, *Online learning: A panacea in the time of COVID-19 crisis*, Journal of Educational Technology Systems, vol, 49, no. 1, September 2020, pp. 5-22.

[3] S. (Avram) Dragoș, *PULSE - a PHP Utility used in Laboratories for Student Evaluation*, in International Conference on Informatics Education Europe II (IEEII), November 2007, pp. 306–314.

[4] S. (Avram) Dragoș, D. Haliță, and C. Săcărea, *Attractors in Web Based Educational Systems a Conceptual Knowledge Processing Grounded Approach*, in Knowledge Science, Engineering and Management, Springer, October 2015, pp. 190–195.

[5] S. (Avram) Dragoș, D. Haliță, and C. Săcărea, *Distilling Conceptual Structures from Weblog Data Using Polyadic FCA*, in 22nd International Conference on Conceptual Structures, Springer, Cham, July 2016, pp. 151-159

[6] S. (Avram) Dragoș, D. Haliță, C. Săcărea, and D. Troancă, *Applying Triadic FCA in Studying Web Usage Behaviors*, in Knowledge Science, Engineering and Management, Springer, October 2014, pp. 73–80.

[7] B. Ganter, and R. Wille, *Formal concept analysis: mathematical foundations*, Springer Science & Business Media, December 2012.

[8] C. Hodges, S. Moore, B. Lockee, T. Trust and A. Bond, *The difference between emergency remote teaching and online learning*, Educause review, vol. 27, no. 1, 2020, pp.1-9.

[9] L.L. Kis, C. Săcărea and D. Șotropa, *Visualizing Conceptual Structures Using FCA Tools Bundle*, in Proceedings of the 23rd International Conference on Conceptual Structures, June 2018, pp. 193 – 196.

[10] L.L. Kis, C. Sacarea, and D. Troanca, *FCA Tools Bundle-A Tool that Enables Dyadic and Triadic Conceptual Navigation*, in 5th Workshop "What can FCA do for Artificial Intelligence" at ECAI, August 2016, pp. 42-50.

[11] F. Lehmann, and R. Wille, *A triadic approach to formal concept analysis*, in International Conference on Conceptual Structures, Springer, Berlin, Heidelberg, August 1995, pp. 32-43.

[12] J. Liebowitz and M. Frank, *Knowledge management and e-learning*, CRC press, April 2010.

[13] Y. Park, *Analysis of online behavior and prediction of learning performance in blended learning environments*, Educational Technology International, vol. 15, no. 2, November 2014, pp. 71-88.

[14] W. Rice, and H. William, *Moodle*, Birmingham: Packt publishing, 2006.

[15] V. Singh, A. Thurman, *How many ways can we define online learning? A systematic literature review of definitions of online learning (1988-2018).*, American Journal of Distance Education, vol. 33, no. 4, 2019, pp. 289-306.

Faculty of Mathematics and Computer Science, "Babeș-Bolyai" University , Cluj-Napoca, Romania

Email address: sanda.avram@ubbcluj.ro