ISSN 1870-4069

Analysis and Comparison of a Classification of Regular Metal Pieces by Convolutional Neural Networks with and without Principal Component Analysis

Mayra Mateo Jiménez[1], Carlos Eduardo Torres Reyes[2], Everardo Efrén Granda Gutiérrez[1], José Arturo Pérez Martínez[2]

[1] Universidad Autónoma del Estado de México, Centro Universitario UAEM Atlacomulco, Mexico

[2] Universidad Autónoma del Estado de México, Unidad Académica Profesional Tianguistenco, Mexico

mmateoj481@alumno.uaemex.mx, {cetorresr, eegrandag, japerezm@uaemex.mx}

Abstract. The classification of images using artificial intelligence techniques (such as neural networks) is fundamentally related to the characteristics of images, such as having a background of a single uniform color. If the illumination scale of the images is different, or the resolution of the image is not sufficient, this may cause false positives when classifying. This work analyzes the importance of classifying images of metal parts (with regular shapes, such as circle, square, rectangle, and flat washer, using a convolutional neural network with and without principal component analysis (PCA), to determine the percentage of classification of images in comparison with the two algorithms by evaluating sensitivity, specificity, accuracy, and precision. The methodology is divided into three steps: obtaining the dataset, training the convolutional neural network, and validating its results. Some representative results of a comparison with a convolutional neural network without PCA were obtained using the RMSProp optimizer with 85.3% precision. PCA achieved the accuracy 98.7%, which indicates that the implementation of PCA improved the classification, with 480 components used.

Keywords: Brightness, convolutional neural network, principal component analysis, classification.

1 Introduction

The regular figures classification is a process where it is necessary to implement pattern recognition algorithms. To carry it out, a wide variety of artificial intelligence techniques are available, and these can be catalogued according to the way in which the data are processed; for example, methods such as deep learning, random forest, and KNN (K-nearest neighbors) [1].

Image processing requires homogeneity, otherwise, loss of important characteristics of the patterns is generated. These problems are caused by certain characteristics of the images, such as their size, width, and length; image noise such as brightness, Gaussian noise, salt and pepper noise, or the resolution. It is difficult to consider all types of noise at the same time in the same job. If the image is not well-defined, the edges or lines of a specific figure cannot be detected. It is important to consider the number of images contained in a dataset. Using many images requires considerable memory storage that is generated due to the size of the dataset. [4].

A pattern detection application focuses on classifying handwritten digits, images, or hyperspectral objects. However, no studies have been done that are related to the classification of metal parts with respect to brightness variation, using algorithms to implement principal component analysis (PCA) and convolutional neural networks [3].

One machine learning application is pattern identification. In manufacturing field, it is necessary in some cases to identify faults in the metal parts produced, such as the manufacture of plugs for lights and the identification of rusted parts to automatically replace them. Using convolutional neural network and vector support machine (SVM) algorithms for the classification of satellite images, a precision value of 83.33% was obtained [6]. In the work of [15], using a convolutional neural network for pattern recognition in images, an accuracy of 92% was obtained.

The PCA algorithm and convolutional neural networks (CNN) belong to the set of deep learning methods because they collect data in a hierarchical way, creating abstract models [5]. However, for most algorithms, inputs for information compression can be data sets with distinct characteristics [2].

In addition to the vector support machine algorithm, PCA can be used for the analysis of data distribution within a Cartesian plane [7, 8].

The extraction of local features is done at high resolution, which minimizes the loss of information. In addition, an alternative implementation is to use a combination of low-resolution images to use more complex functions, if necessary [9, 10].

This type of architecture consists of a variable-length input, which refers to the data entry. In the case of a string or a sentence that has 12 words, the created instances must each be represented by a variable because of its data input of data, in which an internal operation related to the memory of the calculation must be applied. This process is known as a hidden layer [11].

Convolution is the basic component of convolutional neural networks. It consists of an operation between input matrices and filters (kernel), and it results in a feature map. The convolution operation of two MxN arrays, with an input image I (x, y) * k (x,y), is defined as follows [17, 18]:

$$C[x,y] = I[x,y] * K[x,y] = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} I[m,n]K[x-m,y-n] \; \forall \; x = 1,2,3 \ldots, Image\ width, y = 1,2,3 \ldots, Image\ height, \quad (1)$$

where:

I: input image matrix,

K: kernel,

$\frac{1}{MN}$: normalizing image size,

M, N: matrix of image filters,

x, y: image width and height.

The choice of evaluation method depends on the operation of the algorithm and the results to be obtained. The equations for the metrics are as follows [8]:

$$\text{Precision} = \frac{TP}{TN+FP}, \tag{2}$$

$$\text{Accuracy} = \frac{TP+TN}{TN+TP+FP+FN}, \tag{3}$$

$$\text{Sensitivity} = \frac{TP}{TP+FN}, \tag{4}$$

$$\text{Specificity} = \frac{TN}{TN+FP}, \tag{5}$$

where:

n corresponds to the total number of images.

TP: True Positives,
TN: True Negatives,
FP: False Positives,
FN False Negatives.

Therefore, the point of interest in this work is the percentage obtained from the performance evaluation metrics of the convolutional neural network algorithm with and without PCA.

2 Experimental Set-up

Convolutional neural networks, after being properly trained, can adequately classify classes due to feature extraction. Implementing PCA can improve the disadvantages in processing images. One of them is the extraction of characteristics of the images, avoiding redundancy. This in addition to compressing them, conserving the greatest amount of information. For this reason, we propose comparing the results obtained in the implementation of a convolutional neural network with and without the analysis of the main components in the classification of images of metal parts. Another complication of object detection is that they reflect brightness. We intend to evaluate this issue by pre-treating the image and standardizing the brightness values in the images. The images are captured with an industrial-type 1.3 megapixel camera.

The first stage of this work consists of the construction of the dataset, then the training process of the convolutional neural network, and finally the testing stage. From the batch of images processed in each of the stages, 60% are used for the training stage, 20% for the validation set, and 20% for the test set, for a total dataset of 800 images.

The computer resources utilized for the development of this project are 4GB of RAM, AMD Quad-Core Processor @ 1.00 GHz (up to 1.4 GHz), an industrial-type camera of 1.3 megapixels, camera features (European machine vision association (EMVA), 2006; Model: Basler acA1300-200um). The programming language used was Python 3.8. The stages of data processing are described below, as well as some of the development and performance evaluations and the confusion matrix.

2.1 Dataset

For the training and validation images there are four classes being considered: square, rectangle, circle, and flat washer. Each of the folders contains 120 images, 10 images were taken according to each of the classes (regular pieces) and considering the brightness levels (2000lx, 1500lx, 1000lx, 500lx, 100lx). For the training folder, 10 images of the same images per piece were also obtained. Regarding the distribution of images for each class within the folder, 60% of the total images per piece (equivalent to 30 images per class) were considered for this stage, for a total of 120 images per folder (Table 1).

To obtain the images of the validation folder, 10 images were taken for each of the pieces (Circle, Rectangle, Square, Flat washer). Also, the brightness levels of (2000lx, 1500lx, 1000lx, 500lx, 100lx) were considered; the division is shown in Table 1, according to the two illumination sources: a fluorescent lamp and a light bulb.

This folder contains four classes, which correspond to each of the figures. In total there are 40 images, that is, 10 images of each piece. Another 40 images are contained in the test folder, which is itself divided into two folders in different directories, to corroborate that the classification is correctly done. The number of images corresponds only to one type of brightness source to obtain the same number of samples with respect to the other, thus generating a total of 80 images.

Some images are seen in other rectangular formats. In consideration of this, it is convenient to choose a specific size, which was initially 150 x 150 pixels. When images are been capturing with a low intensity of light, such as at scales of 500 and 100 lux, result them too dim, which makes it difficult for the neural network to classify them. Because of this, it is necessary to adjust the brightness.

Brightness adjustment can be done with image pre-processing so that the grayscale images that exceed the threshold calculated according to the percentiles of the images are normalized. This algorithm can also determine which images do not need adjustment. If they do not need it, the algorithm is not applied to them to prevent from increasing in brightness (Figure 1). Some images within the upper levels, such as 2000 lx, 1500 lx, or 1000 lx, were discarded for the application of the brightness adjustment because their percentiles were considered high.

According to this number, a comparison is made with the percentiles of the other images, then the out-of-level percentiles are eliminated. The percentiles obtained are then normalized in a level from 0 to 255. New maximum and minimum percentile values are obtained, and as a result, a new image with the appropriate brightness is generated.

Images that have a suitable brightness scale are discarded from the balance preprocessing. The result of the application of brightness balancing is shown in figure 2. The result of the algorithm is shown in each of the brightness levels, which include high levels such as 2000 lx, 1500 lx, and 1000 lx. For the 500lx and 100lx images, the algorithm is applied according to the percentiles obtained, thus improving the image with respect to its brightness. The edges are also more defined after this configuration.

Table 1. Number of images per piece and brightness levels.

Brightness levels	Number of images per piece			
	Square	Circle	Rectangle	Flat washer
2000 lx	10	10	10	10
1500 lx	10	10	10	10
1000 lx	10	10	10	10
500 lx	10	10	10	10
100 lx	10	10	10	10
Total images	50	50	50	50

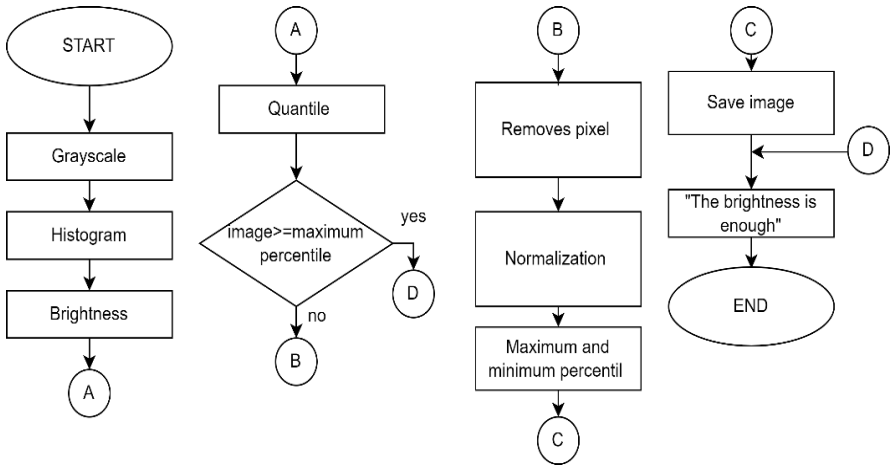


Fig. 1. Brightness balance.

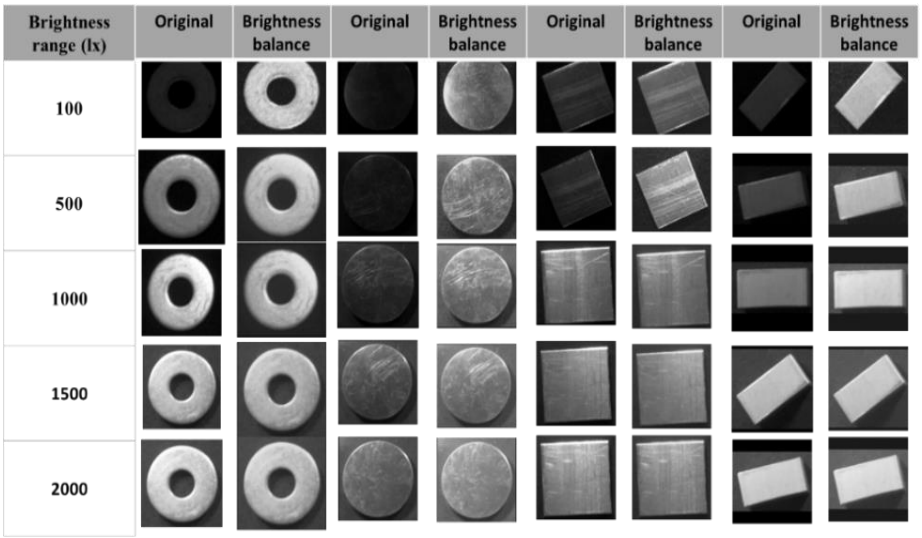


Fig. 2. Brightness balanced samples at each lux level.

2.2 Implementing PCA

The algorithm is divided into three important sections: mean of the data, covariance matrix, and obtaining components. The mean of the data is obtained and subtracted from each of the characteristics (x_1 and x_2), by means of $\frac{x-\bar{x}}{\sigma(x)}$, where x_i is the first extracted feature subtracted from the mean ($\bar{x}$) of the sum of all features in the image. σ refers to the calculation of the covariance, with respect to the characteristics obtained from x. By obtaining a matrix that contains the input images, where from x. Is the set of input images [12, 13].

To acquire an eigenvalue (λ) the equation used is $B \cdot \vec{a} = \lambda \cdot \vec{a}$, where the original matrix (B) has dimension (a square matrix) and an Eigenvector ($\vec{a}$). With the eigenvalues obtained, the total variance of all components is calculated $\sum_{i=1}^{p} Var(x_i) = \sum_{i=1}^{p} \lambda_i$, where λ_i they represent the eigenvalues [14, 16].

The eigenvalues and eigenvectors generate the new coordinates corresponding to the components. To reduce the dimensions of the variance obtained, the number of components is calculated from the largest orthogonal variance. The variance consists of the combination of original variables such as the number of pixels, lines, or curves, according to the characteristics of the image.

To calculate a component, the value of the mean of the same variables is subtracted from the variables that make up the image. Thus, an average of zero is achieved in all components. Finally, the covariance of the eigenvectors is calculated. The behavior of the variance distribution can be seen graphically in Figure 3. The calculation of the covariance matrix was performed by using "cov_matrix = np.matmul (standardized_data.T, standardized_data)", which belongs to scikit-learn in the Python language.

The result is a matrix of (784x784*)* that corresponds to the multiplication of an image size of (28x28) pixels. The result of the calculation of vectors (eigen), by means of "new_coordinates = np.matmul (vectors, standardized_data.T)", is obtained as a result (2, 480), where *2* refers to the *x* and *y* coordinates, and *480* is the maximal number of components to use in the implementation of PCA in the convolutional neural network. The optimal number of components levels from 1 to 480. In Figure 4, an approximation with the use of 10 components demonstrate the behavior of the variance.

The result is the generation of a new image with reduced dimensions and non-redundant features, which is stored in a new vector for neural network training. In this case, the convolutional neural network consists of the following features: a convolutional layer with 32 2x2 feature maps, a 2x2 grouping or pooling layer, 64 convolutional layers of 2x2 feature maps, a 2x2 grouping or pooling layer, a 20% dropout, a flattening layer, a connection layer of 784 neurons with a rectifier activation layer, a connection layer of neurons with a *softmax* activation layer, and an-output layer.

The methodology is implemented in Python using Keras and TensorFlow. The metrics used to evaluate the convolutional neural network algorithm with and without PCA are precision (*1*), accuracy (*2*), sensitivity (*3*), and specificity (*4*). The results obtained are evaluated in a comparison of the accuracy metric, in consideration of the percentage obtained by [15], reaching a percentage greater than or equal to 93%.

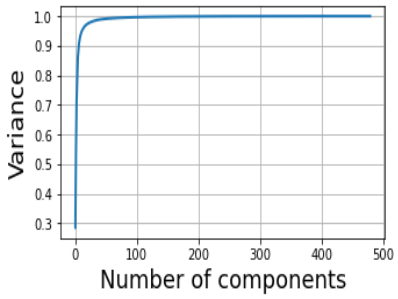


Fig. 3. Number of components in principal component analysis.

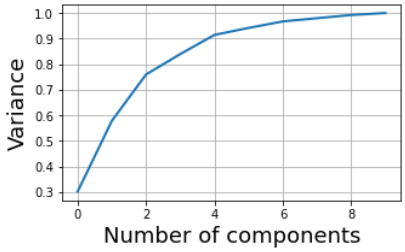


Fig. 4. First 10 components in principal component analysis.

3 Results and Discussion

The following two images are the graphs of the losses obtained for the model without PCA (Figure 6) and with PCA (Figure 7), respectively. In them can see the effect of loss during the training (red line) and validation (green line). In Figure 7, the green line is closer to the red line, that is, the loss is very similar in validation and training.

In the two graphs, it is indicated that the CNN can continue learning and to improve. It can also be observed that the training process stopped prematurely. This effect can be observed at the end of each of the loss curves.

For the execution of the training, the Adam, RMSProp, and Adadelta optimizers were used because they are the most-used optimizers for evaluating neural networks [13]. The results are shown in Table 2. The best percentages are accuracy: 98.8%, sensitivity: 98.8%, accuracy: 98.7%, which were achieved by the implementation of the convolutional neural network with an analysis of main components, using the RMSprop optimizer.

The confusion matrices were obtained using the scikit-learn metrics. The values were obtained automatically in the distribution of the matrix. Results of the classification of the 80 images by class are shown in Figure 8; each one is identified with the name of the algorithm to which it belongs. The labels correspond to 0: circle, 1: square, 2: rectangle, 3: flat washer. The correct classification of true positives is in the white color box in Figure 8, where the highest concentration of classifications

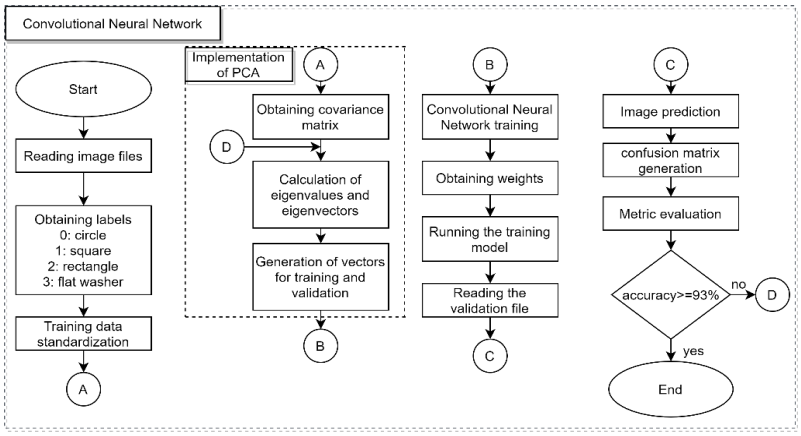


Fig. 5. Implementation of PCA in the neural network.

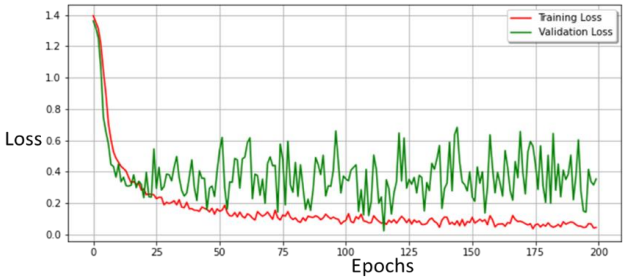


Fig. 6. Training CNN.

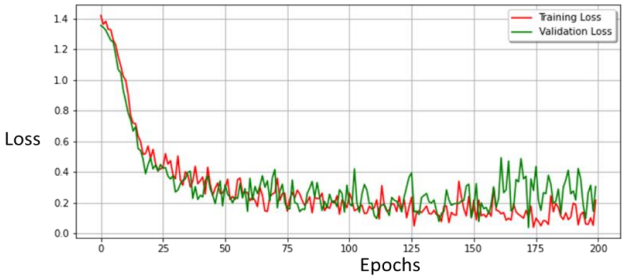


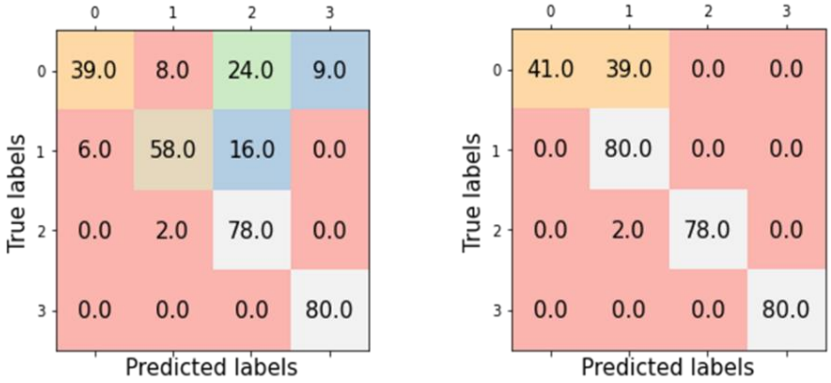
Fig. 7. Training CNN+PCA.

belongs to the confusion matrix of the PCA algorithm together with the convolutional neural network.

The interpretation of the results in the matrix (Figure 8, subparagraph a) indicates that most of the images (true positives) were classified correctly on the main diagonal. However, there are also false positives, which represent images not classified correctly.

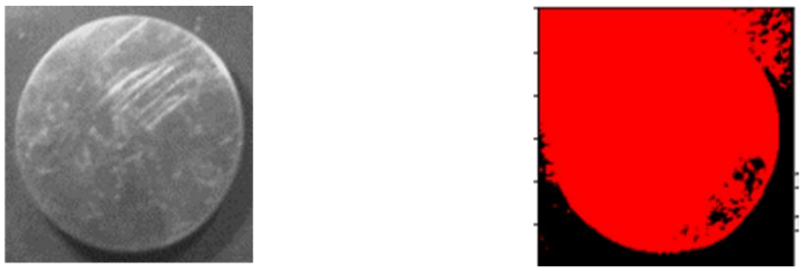
Table 2. Comparison of PCA and non-PCA usage, using the Adam optimizer.

ALGORITHMS	METRICS			
	Optimizer	**Sensitivity**	**Specificity**	**Accuracy**
CNN+PCA	Adam	0.969	0.969	0.968
	Adadelta	0.322	0.204	0.321
	RMSprop	0.988	0.987	0.987
CNN	Adam	0.797	0.786	0.796
	Adadelta	0.250	0.1	0.250
	RMSprop	0.853	0.842	0.853



a) Confusion matrix CNN b) Confusion matrix CNN+PCA

Fig. 8. Confusion matrix results.



a) Image non-PCA. b) Image with PCA

Fig. 9. Results of the application of the PCA algorithm.

For example, in the case of circles, 39 images were considered squares; this is due to the characteristics that the images present, such as the object inclination, its level of brightness or because the image is dark, or because the projections generated by PCA

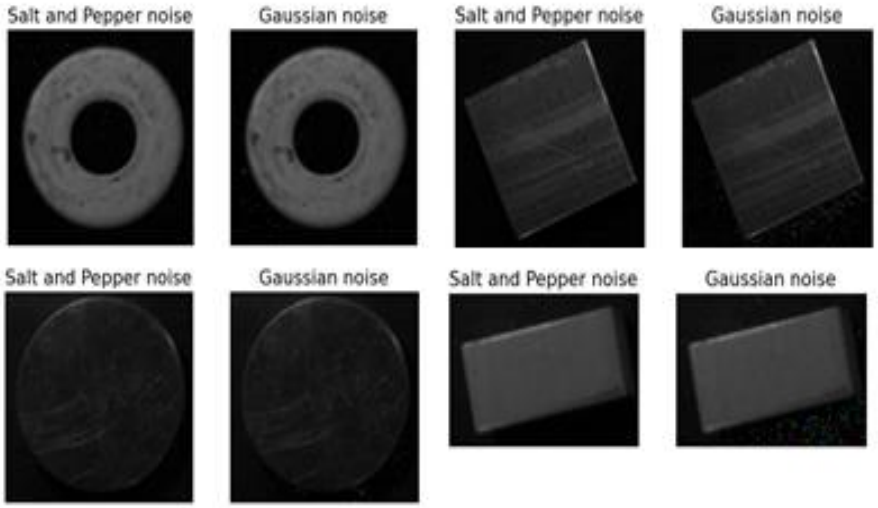


Fig. 10. Types of noise in images.

were not correct and therefore the images were not correctly classified. In the same way, for the images that correspond to the class of the rectangle, two of the images were considered to be squares, when they in fact were not. However, the results are far more accurate than the classification performed without the PCA algorithm, that is, it correctly classifies the largest number of images in each class.

In Figure 8, paragraph b, the only class that prevails as correctly classified is that of the flat washer because the resizing preserves the geometric shape of the image. However, the images are deformed through the convolutions of the CNN algorithm, causing confusion, such that a rectangle, when it is compressed, is classified as a square.

An image without PCA processing is shown in paragraph a) Figure 9. With PCA is shown in paragraph b) Figure 9, some regions of the image are dark, causing the edges to be lost or a shadow to be taken as part of the image to be evaluated. This causes amusing confusion in the convolutional neural network at the time of classification. The loss of characteristics is necessary in the analysis of principal components at the time of calculation, as it reduces the noise that the original image set may contain.

It is important to mention the application of the erode and dilate filter (Figure 10) to the images that contained Gaussian noise, salt and pepper noise, and extra brightness, in order to reduce such noise. The brightness was regularized through the implementation of balancing.

4 Conclusions

The purpose of this study is to demonstrate, through the evaluation of metrics, that the implementation of component analysis in a convolutional neural network improves image classification. Therefore, as assessed by the precision metrics and the RMSProp

optimizer, it shows 98.7%, compared with 85.3% for the convolutional neural network without PCA. Lastly feature extraction reduces feature redundancy in images, resulting in higher classification accuracy.

The position of the metal piece respect to the light source reflects the light in a distributed manner so that, when processed by PCA, sections of the background of the piece of interest end up being considered. This problem causes the component projection to be wrong for regions that should be discarded, as in the case of the circle. Adam (accuracy: 96.8%) and RMSProp (accuracy: 98.7%) resolve decreasing learning rates, compared to training without PCA. In addition, it improves the results obtained compared to the Adadelta optimizer (precision: 32.1% with PCA). Adadelta affects the relationship between the step size and the current gradient. However, the precision remains below 32.1% for each training related to the Adadelta optimizer. Therefore, the classification of true positives is incorrect.

The noise in the images affects the loss of information during the extraction of characteristics by the convolutional neural network with and without PCA. Some types of noise presented in 1.3 MP images are useful: luminance noise, Gaussian noise (which is generated by the camera by itself) or salt and pepper noise.

These types of noise provide complexity in the extraction of main components because the noise is detected as another component of the part, which is unnecessary. Likewise, the PCA algorithm achieves an improvement in the classification. Thus, it is important to consider the number of components that are within the determined level of 1 to 480 components. If fewer are used, the projection is not adequate; therefore, it will generate false positives and false negatives. In the same way, components after the 480th one cannot be picked up, as there are no more combinations to plot on the Cartesian plane. Due to the characteristics of the distribution of the number of images with respect to illumination, no more images were considered.

Therefore, as future work, it is proposed to add more images, with lighting scales of 2000 lx, 1500 lx, 1000 lx, with a lighting balance pretreatment, highlighting the contours of the metal parts, with an opaque black background, to avoid unnecessary light reflection, in order to implement them for training and validation of the neural network according to the proposed balance of 60% training and 40% validation. We also intend to carry out more tests using PCA according to the number of components necessary to project the figure of a circle (which is the piece with greater difficulty to project).

References

1. Arista Jalife, G.C.: Clasificación de imágenes urbanas aéreas: Comparación entre descriptores de bajo nivel y aprendizaje profundo. Inf. Tecnol, 28(3), pp. 209–224 (2020) doi: 10.4067/S0718-07642017000300021.
2. Abiodun, O.I., Aman, J., Omolara, A.E.: State-of-the-Art in Artificial Neural Network Applications: A Survey. Heliyon, 4(11) (2018) doi: 10.1016/ j.heliyon.2018.e00938.
3. Amir, A.N., Mingui, H.: Convolutional Neural Network with PCA and Batch Normalization for Hyperspectral Image Classification. School of Electronics and

Information Northwestern Polytechnical University, pp. 959–962 (2019) doi: 10.1109/IGARSS.2019.8899329.
4. Berzal, F.: Redes neuronales y Deep Learning (2018)
5. Camacho, F.: Redes neuronales convolucionales aplicadas a la traducción del lenguaje verbal español al lenguaje de señas Boliviano. Revista Ciencia, Tecnología e Innovación, 12(13), pp. 755–762 (2016)
6. Cruz Roa, A.S.: Classification and Automatic Mapping of Land Covers in Satellite Images Using Convolutional Neural Networks. Universidad de los Llanos, Villavicencio, Meta. Colombia Suple (2017)
7. Ruiz, D., Bacca, B.: Hyperspectral Images Classification Based on Inception Network and Kernel PCA. In: IEEE Latin America Transactions, 17(12) (2020)
8. Del Carpio Gallegos, J.C.: Mejora de la calidad aplicando la metodologia de superficie respuesta y redes neuronales. Industrial Data, 9(1), pp. 59–63 (2006)
9. Jaya Prakash-Sahoo, S.A., Kumar-Patra, S.: SVM, Hand Gesture Recognition using PCA based Deep CNN Reduced Features and components. In: IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS), 4 (2019) doi: 10.1109/iSES47678.2019.00056.
10. Lerma, B.C.: Análisis de componentes principales de imágenes multiespectrales en el ámbito del arte rupestre. In: Conference: 1st Congress in Geomatics Engineering, pp. 41–47 (2017) doi: 10.4995/CIGeo2017.2017.6597.
11. Ortega, M., González, C.R.: Validación de métodos alternativos para análisis microbiológico de alimentos y aguas. Revista Cubana de Higiene y Epidemiología, 51(1), pp. 111–121 (2013)
12. Mardiansyah, A., Contessa-Djamal, E., Nugraha, F.: Multivariate EEG Signal Using PCA and CNN in Post-Stroke Classification. FORTEI-International Conference on Electrical Engineering (FORTEI-ICEE), 6 (2020)
13. Rivas-Asanza, W.: Redes neuronales artificiales aplicadas al reconocimiento de patrones. Machala: Universidad Técnica de Machala, 29 (2018)
14. Rodríguez, Y.B.: Integración de la red neuronal convolucional con el algoritmo de función. Corporación Mexicana de Investigación en Materiales. División de estudios de posgrado, 87 (2018)
15. Santamaría-Colula, L.Á.: Reconocimiento de patrones en imágenes por medio de redes neuronales convolucionales. Tesis para obtener el título de Maestro en ciencias de la computación. Benemérita Universidad Autónoma de Puebla, pp. 50–76 (2019)
16. Silva-Santos, M., Bernarda-Ludermir, T.: Using Factorial Design to Optimise Neural Networks. Universidade Federal de Pernambuco, 2 (2020) doi: 10.1109/IJCNN.1999.831064.
17. Wenqiang, Y., Peng, C.: Recognition Algorithm of Emitter Signals Based on PCA+CNN. In: IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference, 5, pp. 2410–2414 (2018)
18. Winarno, E., Husni Al Amin, I., Februariyanti, H.: Attendance System Based on Face Recognition System Using CNN-PCA Method and Real-time Camera. In: International Seminar on Research of Information Technology and Intelligent Systems, pp. 301–304 (2019) doi: 10.1109/ISRITI48646.2019.9034596.