Universidad Nacional de Trujillo

Journal of Advanced Mining Modeling (JAMM)

Journal website: https://revistas.unitru.edu.pe/index.php/jamm/index
Vol. 2, Nº 1, pp. 42-60, January – June 2026

Journal of Advanced Mining Modeling (JAMM)

# Optimization of neural network for detection of unstable zones in rock mass through digital image processing

Anghelo Francisco Vasquez-Vargas[1*], Gersun Natanael Cruz-Pacheco[2], Alex Jeferson Guerra-Uriol[3], Deybis Yhojan Romero-Rodrigues[4]

[1.] Departamento de Ingeniería de Minas / Facultad de Ingeniería / Universidad Nacional de, Trujillo, Trujillo, Perú
[2.] Departamento de Ingeniería de Minas / Facultad de Ingeniería / Universidad Nacional de, Trujillo, Trujillo, Perú
[3.] Departamento de Ingeniería de Minas / Facultad de Ingeniería / Universidad Nacional de, Trujillo, Trujillo, Perú

* Corresponding author: afvasquezva@unitru.edu.pe (Anghelo Vasquez-Vargas)

Received: December 22, 2025 Revised: December 28, 2025 Accepted: January 3, 2026 Published online: January 7, 2026

**Abstract**

This study addresses the optimization and training of a convolutional neural network (CNN) aimed at the detection and segmentation of unstable zones in rock masses through digital image processing. To achieve this, YOLO v8 architecture was implemented using the Python programming language, leveraging its deep learning capabilities. The main objective was to develop a fast and effective tool to identify potentially dangerous areas in mining operations, especially in underground environments, to strengthen safety in the sector. The methodology included training the CNN with an image database collected in the surroundings of Huamachuco, complemented by a review of previous research that applied YOLO-based solutions to similar problems. The results show that the proposed procedure is suitable for achieving effective detection and segmentation of unstable zones in rock masses. In conclusion, the proposed solution has the potential to significantly improve mining safety by offering a reliable tool for identifying risk areas.

**Keywords:** Convolutional neural network (CNN); digital image processing; rock mass; unstable zones; optimization; deep learning; training; YOLO v8.

## 1. Introduction

In recent decades, artificial intelligence has demonstrated a remarkable impact on image processing across various fields. For example, in pavement analysis, fuzzy systems and neural networks have been used to classify mixtures and detect cracks [1,2]. In the medical field, deep learning has been employed for melanoma detection in dermoscopic images, yielding promising results in diagnostic accuracy [3]. Likewise, convolutional neural networks (CNNs) have proven effective in shape segmentation and object detection tasks [4,5].

In this context, Song, Gao, and Chen [6] proposed a hybrid network called MSFFN, based on YOLO-v3, which achieved superior performance compared to Faster-RCNN in multispectral image detection. Similarly, recent studies have trained artificial intelligence models to predict landslides in the Ancash region, achieving an Area Under the Curve (AUC) of 90.5%, outperforming other architectures [7].

These advances highlight the versatility of CNNs and their potential in identifying complex patterns. The growing digitalization and automation of processes in the mining industry have opened new opportunities to improve the precision and speed of risk detection without compromising result quality [8]. Detecting unstable zones in rock masses is a critical challenge for safety in underground operations. In this regard, digital image processing and deep learning have emerged as effective tools for identifying potentially hazardous areas.

The purpose of this study was to train and optimize a convolutional neural network based on the YOLO v8 architecture, implemented in Python, for the detection and segmentation of unstable zones in rock masses. The main objective was to provide a fast and reliable tool that contributes to improving safety and efficiency in the mining sector. To achieve this, specific objectives were proposed, including: (a) tuning the neural network

through transformations of the image dataset and two training stages; (b) optimizing the model using the Stochastic Gradient Descent algorithm; (c) creating comparative tables with relevant accuracy and loss metrics; (d) graphing training and validation values to identify possible adjustments or overfitting; and (e) performing estimations on a set of test images collected around Huamachuco.
In summary, this work aimed to demonstrate that the application of CNNs optimized through YOLO v8 constitutes an effective solution for detecting unstable zones in underground mining, offering a significant advancement in risk prevention and worker protection.

## 2. Materials and methods

### 2.1 Population

The population for this study is in the Sánchez Carrión province of the La Libertad region, consisting of photographs of rock masses found around the city of Huamachuco.


**Figure 1.** Province of Sanchez Carrion

### 2.2 Sample

Photographic captures were taken of rock masses in the surroundings of the city of Huamachuco – La Libertad (specifically along the Huamachuco–Sanagoran road and the El Potrerillo sector on the way to Yanasara), with a total of 1324 photographs collected.

### 2.3 Sample size

The images have an average resolution of 4000x5000 pixels and are mostly in JPEG format.

### 2.4 Type of research

The research is Applied – Experimental in image recognition and segmentation, as its goal is to solve real-world problems through the development of algorithms and systems that interpret and understand visual content in images, particularly in the mining field, such as recognizing the degree of fracturing that rock masses may exhibit.

### 2.5 Artificial Neural Network

A TLU (Threshold Logic Unit) is an artificial neuron that processes multiple inputs and produces an output. It works by computing a weighted sum of its inputs with associated weights and then activates if this weighted sum exceeds a threshold. The output may be 1 or -1, depending on the activation function used.
Perceptrons are layers composed of TLUs, with each one connected to all inputs. In a sequence of TLUs, the output layer is fully connected to the inputs. In a multilayer neural network (MLP), there is an input layer, hidden layers, and an output layer. The connections between the hidden layers and the output layer become more abstract as the network deepens.
In convolutional neural networks (CNNs) for object recognition, the focus is on image processing. CNNs use input pixels and take advantage of the hierarchical structure of real-world images. The first convolutional layer connects only to pixel regions in the image, extracting small features. These features are then merged with larger features in subsequent hidden layers. This hierarchical structure is key to the success of CNNs in image recognition.

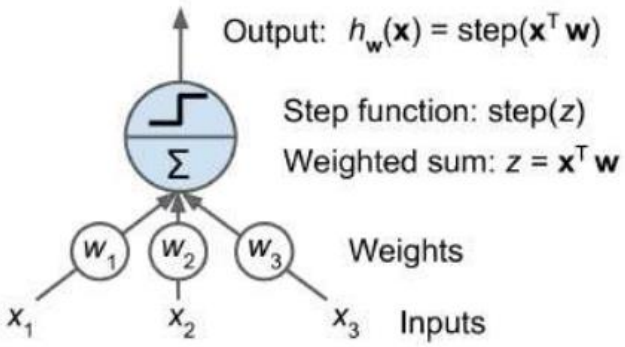



**Figure 2.** Structure of a TLU or threshold logic unit

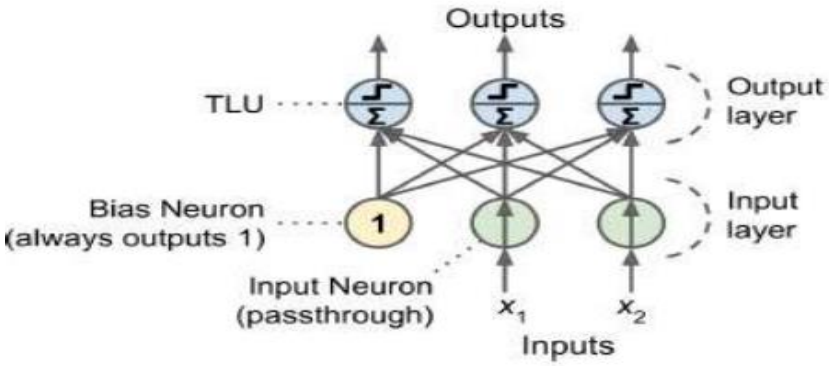



**Figure 3.** Composition of a perceptron

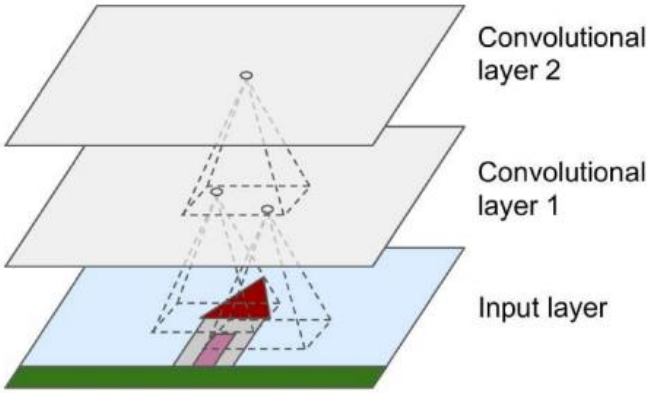



**Figure 4.** Diagram of how a convolutional neural network operates

Abstract model of an artificial neuron

| | |
|---|---|
| $y = \gamma\left(\sum_{i=1}^{m}(W_i * X_i) + W_0\right)$ | (1) |

Where:
$\gamma$: Activation function
$W_i$: Weights associated with each input
$X_i$: Input values or data received by the artificial neuron
$W_0$: Threshold weight or bias term

The function produces a binary output known as the all-or-nothing function, equivalent to the sign function. There is also a unit step function [9]:

| | |
|---|---|
| $\gamma = sgn(\mathcal{Z}) = \begin{cases} 1, & si\ \mathcal{Z} \geq 0 \\ -1, & si\ \mathcal{Z} < 0 \end{cases}$ | (2) |
| $\gamma = U(\mathcal{Z}) = \begin{cases} 1, & si\ \mathcal{Z} \geq 0 \\ 0, & si\ \mathcal{Z} < 0 \end{cases}$ | (3) |

Where:
$U$: Produces an output of 1 if the value is greater than or equal to zero, or 0 if it is less than zero.
$\mathcal{Z}$: The amount of "excitation" the neuron receives.

During the training of a neural network, connections are established based on input images, and weights are adjusted to achieve accurate results. Prior training is essential for effective analysis. The neural architecture is built, weights are assigned and adjusted through modeling to minimize error and meet objectives [10].

| | |
|---|---|
| $W_{ij}(t+1) = W_{ij}(t) + \Delta W_{ij}(t)$ | (4) |

In supervised learning, a supervisor checks the network's output and, if it is not as expected, adjusts the weights to correct it. This process is known as error-correction learning. In this approach, weight adjustments are made based on the difference between the actual and expected results [11].

| | |
|---|---|
| $Global\ mean\ error = \frac{1}{2} * \sum_{k=1}^{P} \sum_{j=1}^{M} \left(y_j^k - t_j^k\right)^2$ | (5) |

Calculation of the relative variation of the error:

| | |
|---|---|
| $\Delta W_{ij} = -\alpha * \frac{\partial * E * [W_{ij}]}{\partial * W_{ij}}$ | (6) |

Calculation of the accumulated relative variation of the error for all patterns:

| | |
|---|---|
| $\Delta W_{ij} = -\alpha * \frac{\partial * E * \left[W_{ij}\right]}{\partial * W_{ij}} = \alpha \sum_{k=1}^{P} \left(t_j^u - y_j^u\right) * x_j^u$ | (7) |

It is essential that a neural network be capable of generating new and unseen responses during training. This highlights the importance of an effective training process with low error, indicating that the network has learned the necessary patterns [10].
However, there is a risk of overfitting, where adjustments to reduce error result in increased error due to excessive memorization. To prevent this, cross-validation is used to identify the optimal point where the network has learned sufficiently [10].

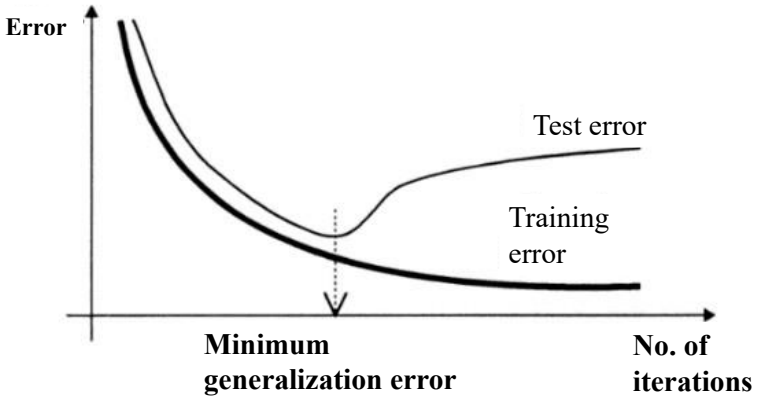



**Figure 5.** Analysis of the neural network "overfitting" phenomenon

### 2.6 Convolutional neural network

A Convolutional Neural Network (CNN) is an artificial network designed to detect and classify objects such as images by identifying spatial and temporal features. It contains convolutional layers that extract initial features like edges and corners, which are then used to recognize shapes in deeper layers. The final fully connected layer makes predictions based on these extracted features [12].

The Softmax activation function is used in the output layer for multi-class classification, scaling the inputs to a range between 0 and 1 and normalizing the output:

$$y_j = \frac{e^{Z}i}{\sum_{j \in group} e^{Z}i} \quad (8)$$

ReLU (Rectified Linear Unit) is commonly used and is defined as $f(x) = max(0, x)$ .

$$\big(ReLu, R(x)\big) = \max(0, x)) \quad (9)$$

The convolution operation applies a kernel to an image to produce a new output [25][13]:

$$S(i,j) = (K * I)(i,j) = \sum_{m}\sum_{n} = I(m,n)\, K(i-m, j-n) \quad (10)$$

In a Convolutional Neural Network (CNN), the convolutional layer is the core component where most of the computation takes place. It takes a color image as input with three dimensions height, width, and depth corresponding to RGB channels. A filter or kernel acts as a feature detector, sliding over the image to determine the presence of specific features [14].

The convolution operation allows the network to focus on specific regions of the image, learning spatial hierarchies from local patterns such as contours, edges, color blobs, and lines [15].

A 2D filter represents a small region of the image and is typically a 3x3 matrix. The product between the input pixels and the filter is computed by applying the filter over sections of the image. This process is repeated until the filter has scanned the entire image, producing a feature map or convolved activation map as the final output [14].

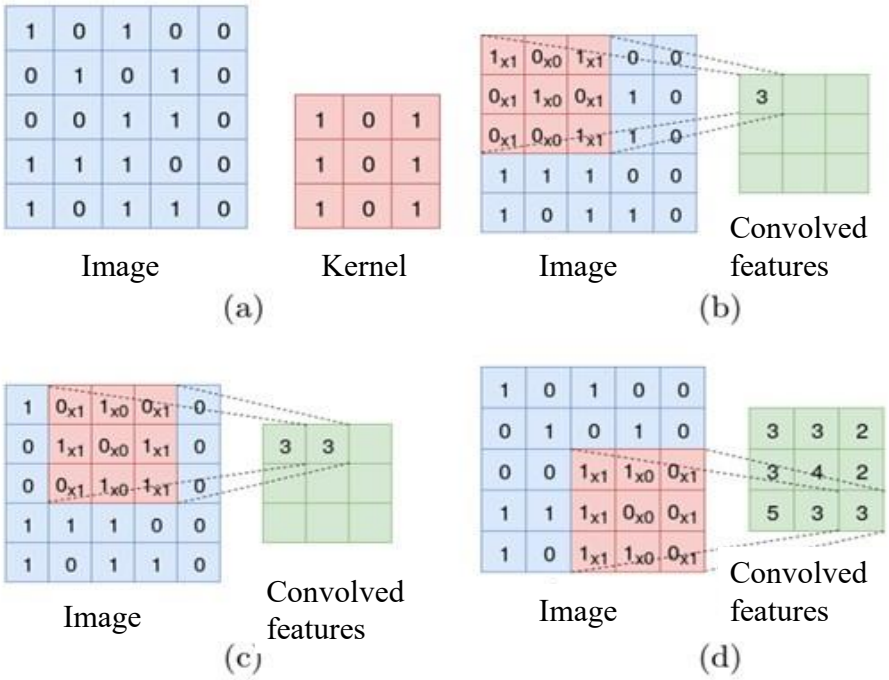



**Figure 6.** Convolutional layer

(a) Defines the fundamental goal to be achieved, (b) Describes the process, tools, data, and technical steps followed to obtain the results, (c) These are the direct findings from the experiment or study, and (d) It is the interpretation of the results.

The reduction of layers and the pooling operation are strategies used to reduce the amount of input data in a neural network by employing different methods to select or average values within a window [16][15].

The classification layer in a convolutional neural network (CNN) uses the features extracted by the previous layers and their filters to perform the classification task. Fully connected layers (FC) often use the softmax activation function to classify inputs, generating probabilities between 0 and 1. Meanwhile, convolutional and pooling layers commonly use ReLU functions.

In the visual representation of a CNN, the process starts with convolution, where filters multiply pixels and sum them to create a feature map. Then, the ReLU function is applied, modeling relationships between inputs and outputs by truncating negative values. Next, the pooling layer reduces the map size to speed up computation and prevent overfitting.

Finally, the fully connected layers act as classifiers in the convolutional network. They use techniques such as dropout, which deactivates neurons to prevent overfitting. In summary, a CNN extracts features, applies nonlinear functions, reduces redundant information, and performs the final classification [12].

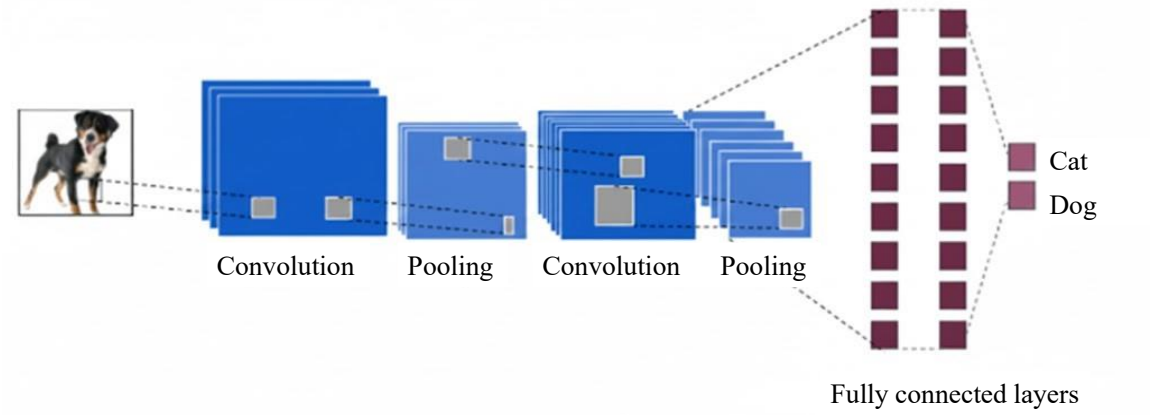



**Figure 7.** Convolutional neural network

## 2.7 YOLO-v8

YOLO (You Only Look Once) approaches object detection as a single regression problem, simultaneously predicting bounding boxes and class probabilities [17]. Unlike previous approaches, it does not reuse classifiers [17]. YOLOv8, released in 2023 by Ultralytics (the creators of YOLOv5), uses PyTorch and has no restrictions, reducing bounding box predictions and accelerating the Non-Maximum Suppression (NMS) technique. It offers five scaled versions, from nano to extra-large, and its architecture includes 1x1 reduction layers followed by 3x3 convolutions [17][18].

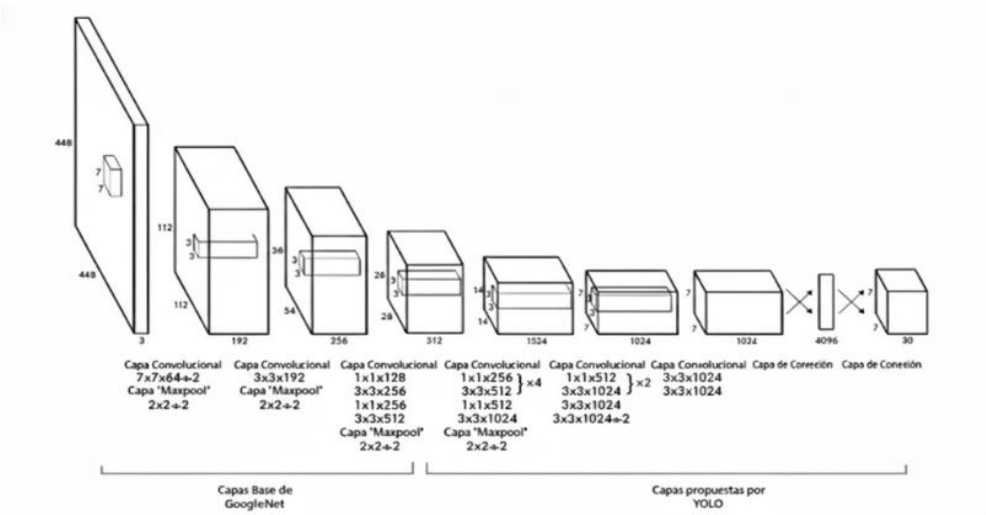



**Figure 8.** YOLO v8 architecture

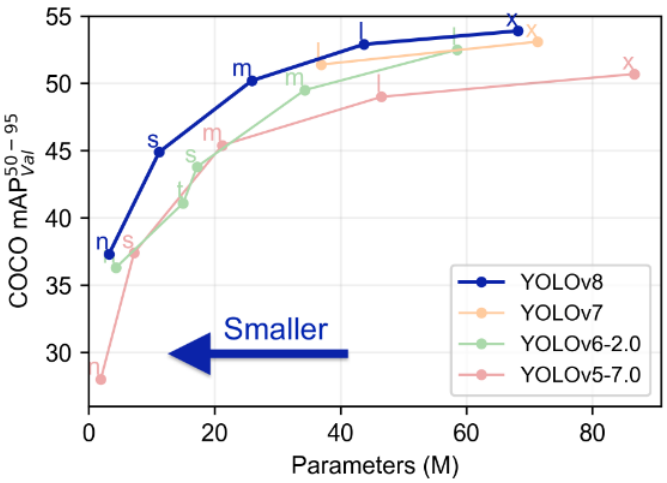



**Figure 9.** Number of parameters in YOLO v8 for Its different versions

## 2.8 Stochastic gradient descent (SGD) optimizer

Stochastic Gradient Descent (SGD) is an optimization algorithm used in convolutional neural networks to update the network's weights during training [19]. SGD is computationally efficient and can be improved with momentum to accelerate the model's convergence [20]. It is the basic optimization method, and the weight update step is defined as follows:

| | |
|---|---|
| $W = W - \alpha * \Delta W$ | (11) |

In this case, the learning rate is fixed, meaning it does not change during training.

## 3. Results

Once the neural network model is defined, some transformations will be applied to the collected image dataset to reduce training time and increase performance. New augmented versions of each image in the training dataset will be added, enabling the model to learn from more information.
Furthermore, the optimizer will be implemented and compiled with multiple epochs (in 2 stages) to properly regulate the neural network.
After researching various optimizers, it has been concluded that the most suitable one for the characteristics of our database due to its efficiency and strong results is SGD (Stochastic Gradient Descent). Although there are other options such as RMSprop, Adam, and more, SGD was selected for this study.

### 3.1 Database description

The image dataset consists of photographs showing fractures in rocks and rock masses from the surroundings of the city of Huamachuco – La Libertad. A total of 1,324 images were collected. To reduce processing time, only the most suitable and relevant images were selected, resulting in a final dataset of 226 images, which were divided into the following subsets:

**Table 1**.Initial image dataset

| Training | Validation | Testing |
|---|---|---|
| 159 | 46 | 23 |

### 3.2 Labeling, transformation, and creation of new image sets

Labeling was carried out by identifying visible fractures in the images and attempting to frame the entire corresponding area. This is a labor-intensive and crucial process. The quality of the labels determines how well the neural network learns to identify and successfully segment the fractures.

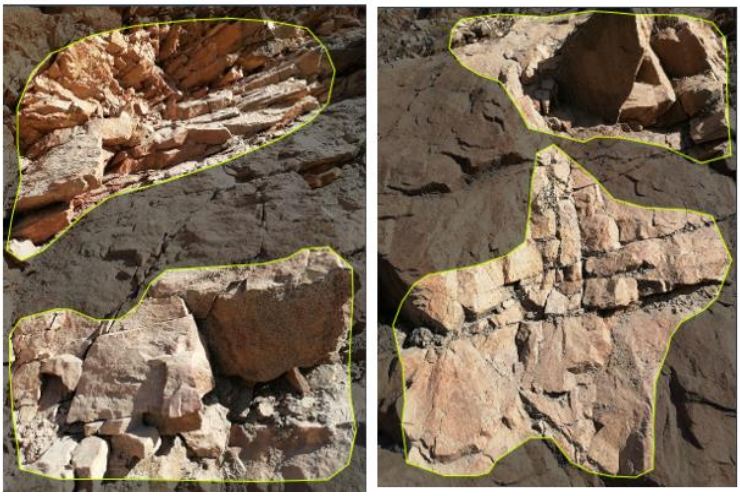


**Figure 10.** Example of image labeling

For the image transformations (before training and model compilation), which include cropping, resizing, rotation, and more, only two preprocessing techniques were applied due to their unique benefits for the dataset:

- *Resizing:* Reduce the image size to 650x650 pixels.
- *Auto-orientation:* Standardize the pixel ordering.

Additionally, the following augmentation parameters were used to generate new sets of images:

- *Flip:* Add horizontal or vertical flips to make the model insensitive to the subject's orientation.
- *90° Rotation:* Add 90-degree rotations to help the model be insensitive to camera orientation.
- *Rotation:* Add rotation variability to help the model be more robust to camera roll.
- *Shear:* Add perspective variability to help the model be more robust to subject pitch and yaw.
- *Blur:* Add random Gaussian blur to improve robustness to camera focus.
- *Noise:* Add noise to make the model more resilient to camera artifacts.

**Table 2.** Image transformation values

| Parameter | Transformation |
|---|---|
| Flip | Horizontal and vertical |
| 90° rotation | Clockwise, counterclockwise, and upside down |
| Rotation | Between -45° and +45° |
| Shear | ±25° Horizontal and ±25° Vertical |
| Blur | Up to one pixel |
| Noise | Up to 5% of the pixels |

For the bounding box, some additional parameters were also applied:

**Table 3.** Boulding box transformation values

| Parameter | Transformation |
|---|---|
| Flip | Horizontal and Vertical |
| 90° rotation | Clockwise, counterclockwise, and upside down |
| Rotation | Between -45° and +45° |
| Shear | ±25° Horizontal and ±25° Vertical |

After applying the various parameters, the dataset increased to 546 images, with the following distribution:

**Table 4.** Final image dataset

| Training | Validation | Test |
|---|---|---|
| 435 | 61 | 22 |

### 3.3 SGD optimizer metrics

For the optimizer parameters, standard values will be used, as they have yielded good results in similar research.

**Table 5.** SGD optimizer values

| Parameter | Stage 1 | Stage 2 |
|---|---|---|
| Learning rate | 0.001 | 0.001 |
| Decay | 1e-5 | 1e-5 |
| Momentum | 0.9 | 0.9 |

### 3.4 Training and regularization details

The network training will consist of 2 stages of weak epoch training and one stage of strong epoch training. This is done to regulate the network and prevent overfitting and underfitting. The SGD optimizer (Stochastic Gradient Descent) will be used in all stages with the parameters previously mentioned. This algorithm is preferred because it avoids getting stuck in local minimum of the optimization function. The number of epochs is set at 55 and 80, respectively, for the corresponding training stages.

### 3.5 Evaluation metrics

Precision (Pres) is defined as the proportion of retrieved cases that are considered relevant. Meanwhile, Recall (Rec) refers to the fraction of relevant cases that have been identified relative to the total number of relevant cases [21].

$$Press = \frac{TP}{TP + FP} \tag{12}$$

$$Rec = \frac{TP}{TP + FN} \tag{13}$$

The Average Precision (AP) metric was introduced in the VOC2007 challenge [10] as a standard way to evaluate detector performance. AP corresponds to the area under the precision-recall (P-R) curve for a given class. Furthermore, the mean Average Precision (mAP) is calculated as the average of the AP scores across all classes.

To evaluate the metrics accurately, a reference metric is needed to identify correct predictions. In this case, the Intersection-over-Union (IoU) is used. This metric establishes the ratio between the areas of overlapping regions (true positives, TP) and non-overlapping regions (false positives, FP), using the following formula [21]:

$$IoU = \frac{Area(B_det \cap B_{gt})}{Area(B_det \cup B_{gt})} \quad (14)$$

In this context, where Bgt represents the ground truth region and Bdet is the detected region, a prediction is considered a true positive (TP) if the IoU > 0.5; otherwise, it is a false positive (FP). Based on this, you can then calculate the values for equations (12) and (13) accordingly [21].

### 3.6 Stage 1 results

It can be observed that in the first training stage, promising results have been obtained in terms of precision and mAP50, especially for both bounding box detection and segmentation. The precision values are quite high, indicating that the model is successfully identifying and segmenting with moderately acceptable performance.

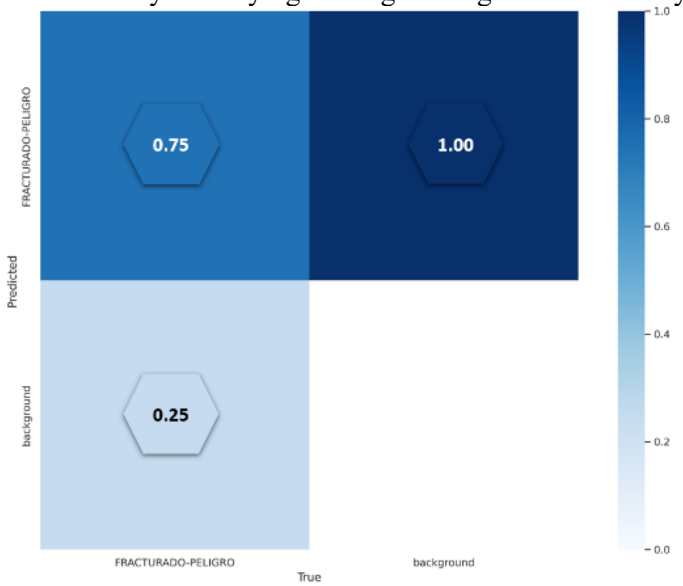



**Figure 11.** Confusion matrix for the training-validation set, for bounding box and segmentation – Stage 1

**Table 6.** Summary of relevant results from training – stage 1

| Variables | Box | Segmentation |
|---|---|---|
| Precision | 0.872 | 0.776 |
| recall | 0.639 | 0.663 |
| mAP50 | 0.782 | 0.695 |
| mAP95 | 0.497 | 0.336 |

In the table showing the top 5 results for both bounding box detection and segmentation, it is observed that training set precision is high in the early epochs and remains consistently good in the middle and later epochs, indicating a good fit of the model to the training data. However, in the validation set, precision slightly decreases, which may suggest initial signs of overfitting.

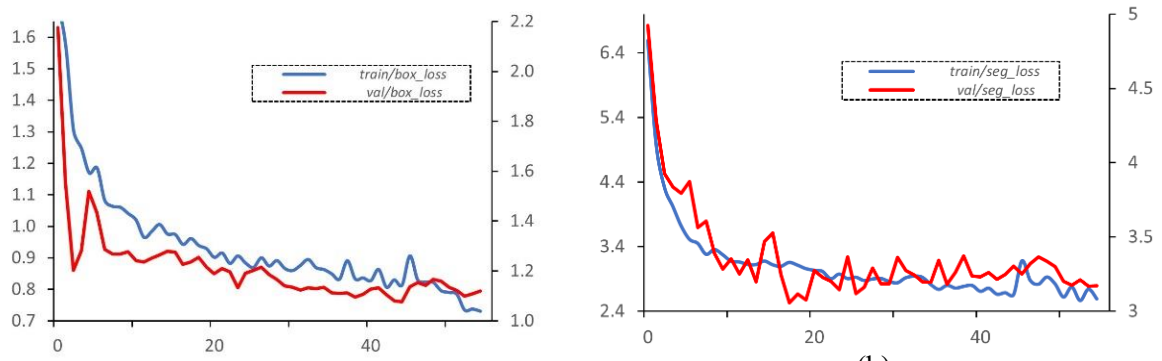
**Table 7**. Top 5 training results – bounding box (stage 1)

| Epoch | Loss-train | Precision-train | Loss-val |
|---|---|---|---|
| 7 | 1.0132 | 0.88663 | 1.2686 |
| 30 | 0.87902 | 0.87902 | 1.1341 |
| 19 | 0.87861 | 0.87382 | 1.2145 |
| 53 | 0.68747 | 0.87044 | 1.1086 |
| 31 | 0.82581 | 0.86708 | 1.1235 |

**Table 8.** Training results – segmentation (stage 1)

| Epoch | Loss-train | Precision-train | Loss-val |
|---|---|---|---|
| 7 | 3.2814 | 0.77319 | 3.6052 |
| 30 | 2.8387 | 0.71371 | 3.3618 |
| 19 | 3.0591 | 0.79444 | 3.0767 |
| 53 | 2.7419 | 0.79013 | 3.1671 |
| 31 | 2.9162 | 0.77074 | 3.273 |

Additionally, it can be seen in the graphs that as epochs progress, losses decrease in both cases (bounding box detection and segmentation), which is a positive sign that the model is learning and adjusting correctly to the training data. However, some spikes in loss during the final epochs may be an early indicator of overfitting.




(a) (b)

**Figure 12.** Results obtained for the training and validation sets – bounding box and segmentation – Stage 1(a) Results obtained for the bounding box training and validation set (b) Results obtained for the segmentation training and validation set – Stage 1

For the validation process results, bounding box estimation is quite accurate in almost all cases. However, segmentation results are weaker but still acceptable, considering the metrics and existing challenges, such as the dataset size and manual labeling process.

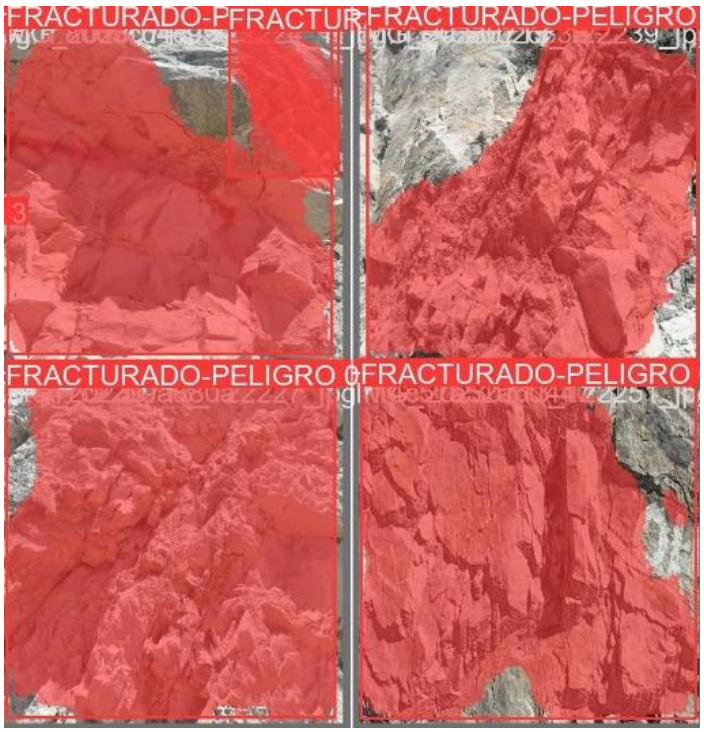



**Figure 13.** Validation process results – stage 1

For the test process results, with 22 images that the network had not previously interacted with, it is observed that bounding box estimations remain fairly accurate. Although, due to the aforementioned limitations and deficiencies, segmentation results are average yet in some cases, notably good.


**Figure 14.** Image test process results – stage 1

In the summary charts, which show the most relevant metric results, the performance and good fit of the model to the training data are more clearly demonstrated.

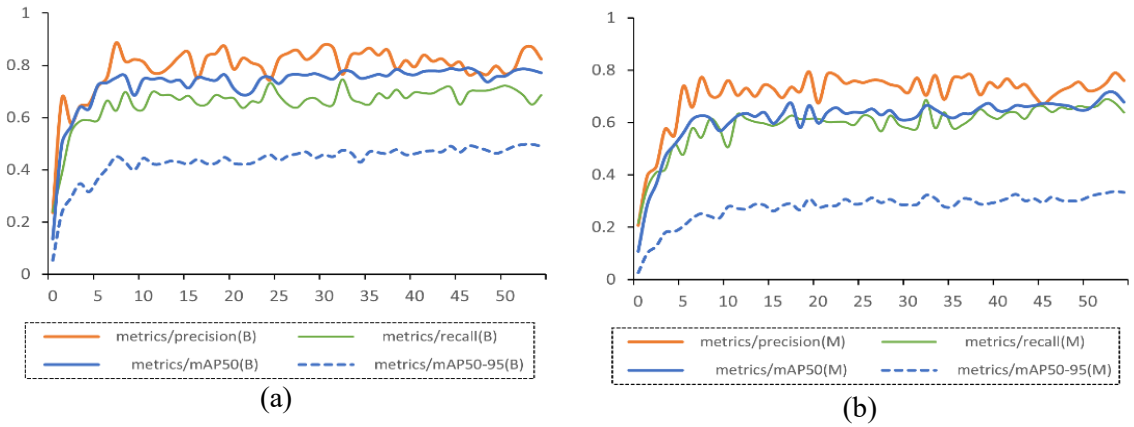



**Figura 15.** Results of the Most Relevant Training Metrics for Bounding Box and Segmentation – Stage 1. (a) Results of the most relevant training metrics for bounding box. (b) Results of the most relevant training metrics for segmentation – Stage 1

### 3.7 Stage 2 results

The results in the second training stage are very promising, showing a moderately significant increase in both precision and mAP50 for bounding box detection and segmentation. Precision has improved in both tasks, recall has significantly increased, and mAP50 has also shown improvement.

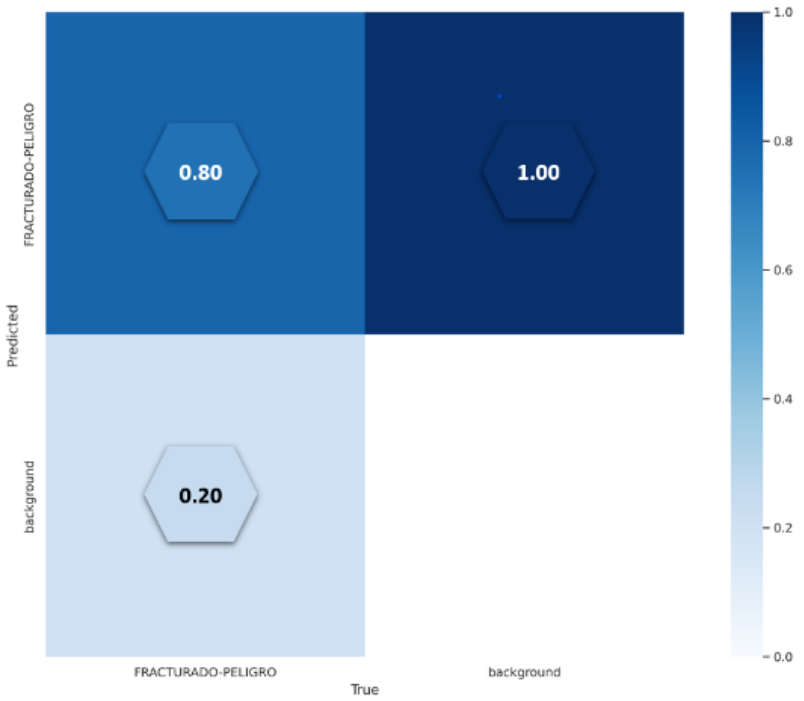



**Figure 16.** Confusion matrix for the training-validation set, for bounding box and segmentation – Stage 2

**Table 9.** Summary of key training results – stage 2

| **Variable** | **Bounding box** | **Segmentation** |
|---|---|---|
| Precision | 0.885 | 0.743 |
| Recall | 0.743 | 0.699 |
| mAP50 | 0.826 | 0.71 |
| mAP95 | 0.521 | 0.36 |

In the table of the top 5 results for bounding box detection and segmentation, it is observed that precision is high during early epochs and remains consistently good throughout mid and later epochs. This indicates a solid fit of the model to the training data. However, on the validation set, precision slightly decreases, which could be a sign of the onset of overfitting.

**Table 10.** Top 5 highlighted training values for bounding box – stage 2

| Epoch | Train loss | Train precision | Validation loss |
|---|---|---|---|
| 42 | 0.74145 | 0.92994 | 1.122 |
| 59 | 0.71418 | 0.90587 | 1.1341 |
| 54 | 0.76185 | 0.90476 | 1.1531 |
| 45 | 0.74168 | 0.90378 | 1.1521 |
| 72 | 0.72744 | 0.89877 | 1.1236 |

**Table 11.** Top 5 highlighted training values for segmentation – stage 2

| Epoch | Train loss | Train precision | Validation loss |
|---|---|---|---|
| 54 | 2.5476 | 0.83792 | 3.4276 |
| 51 | 2.5253 | 0.83451 | 3.4341 |
| 59 | 2.4381 | 0.83152 | 3.4662 |
| 50 | 2.6359 | 0.82744 | 3.4834 |
| 78 | 2.415 | 0.82464 | 3.3269 |

Moreover, the graphs show that as the epochs progress, training loss decreases in both cases (bounding box and segmentation), but in the final epochs, validation loss spikes, diverging from training loss especially in segmentation, suggesting the model begins overfitting in that range.

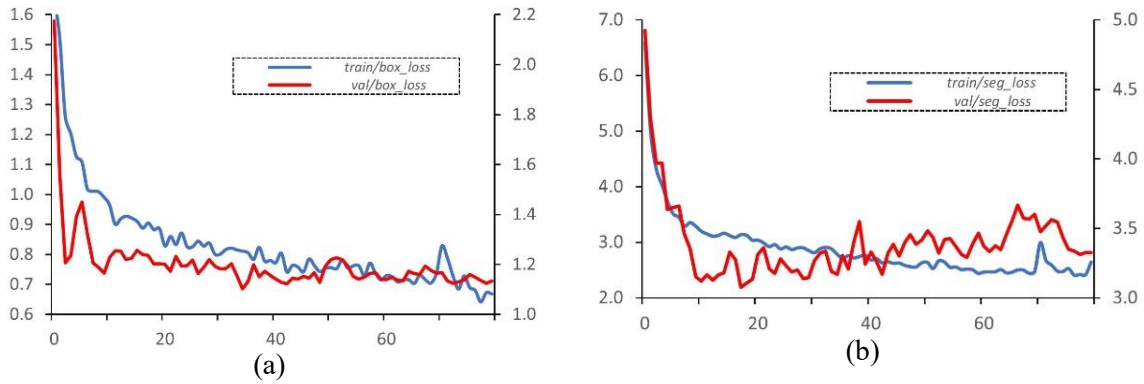



**Figure 17.** Results for training and validation sets – bounding box and segmentation – Stage 2. (a) Training and validation results for bounding box. (b) Training and validation results for segmentation

For the validation process results, the bounding box estimation is quite accurate in nearly all cases, similar to Stage 1. However, in terms of segmentation, the performance is weaker, and signs of overfitting can be observed. Even so, the estimations remain acceptable.


**Figure 18.** Validation image results – stage 2

In the summary graphs, where the most relevant metric results are displayed, the improvement achieved in Stage 2 becomes clearer. However, signs of overfitting affecting the segmentation parameters are also noticeable.

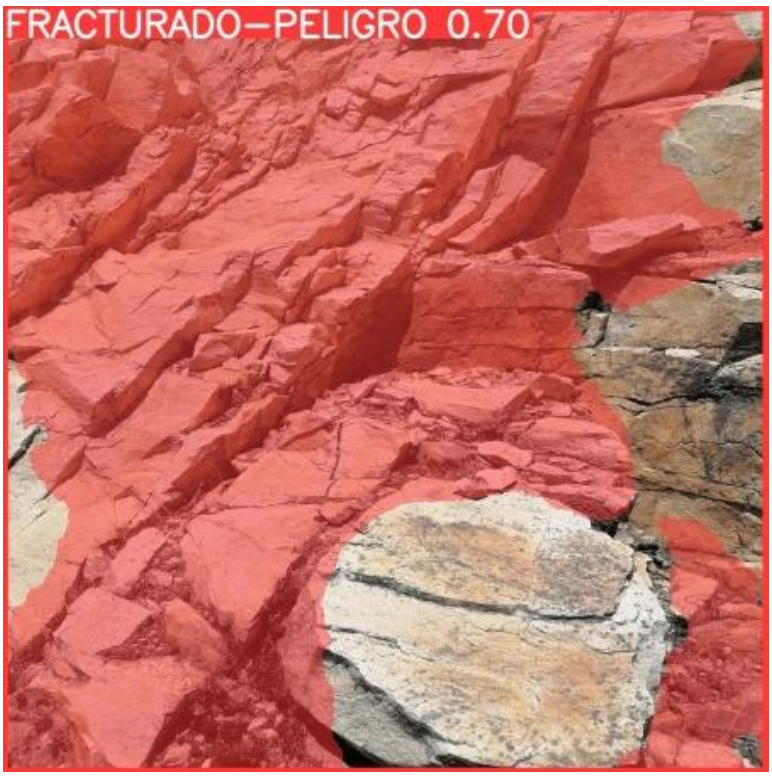



**Figure 19**. Test image results – stage 2

In the summary charts, which show the most relevant training metrics, the results and the model's good fit to the training data are more clearly evident.

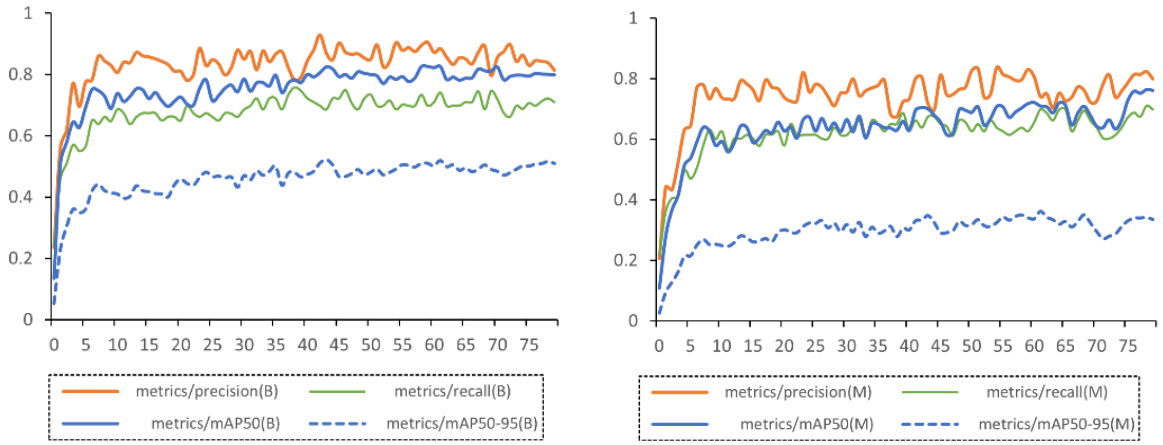



**Figure 20.** Most Relevant Training Metrics for Bounding Box and Segmentation – Stage 2. (a) Most relevant training metrics for bounding box. (b) Most relevant training metrics for segmentation in Stage 2

## 4. Conclusions

The regulation process of the convolutional neural network, based on data augmentation techniques such as 90° rotation and shearing, allowed for an increase in the training set from 218 to 518 images. By defining two training stages—one "weak epoch" of 55 cycles and one “strong epoch” of 80 and applying the Stochastic Gradient Descent (SGD) optimization algorithm with a learning rate of 0.001 and a momentum of 0.9, superior convergence and optimized performance were achieved. These methodological adjustments were crucial for enabling the network to reach greater generalization capacity in identifying complex patterns in the rock mass. Quantitative results confirm the superiority of the second training stage, especially in bounding box detection, where precision increased from 0.872 to 0.885 and recall significantly improved from 0.639 to 0.743, raising

the mAP50 to 0.826. In the area of segmentation, although precision slightly decreased (from 0.776 to 0.743), the mAP50 rose from 0.695 to 0.71. However, the analysis of the loss curves revealed the emergence of instability spikes during validation, indicating a more pronounced overfitting phenomenon in segmentation during Stage 2, slightly affecting the clarity of final estimations on the test images.
As a main limitation, it was found that the observed overfitting restricts the network's ability to maintain consistent accuracy in environments with extreme variations in lighting or textures not present in the training dataset. Therefore, for future work, it is recommended to implement stricter regularization techniques such as Dropout or Early Stopping, and to expand the dataset by incorporating a wider variety of geological samples. Additionally, integrating this model into real-time monitoring systems for geomechanical risk prevention is proposed, allowing for immediate response to signs of instability in mining operations.

## 5. Conflict of interest

The authors declare no conflict of interest.

## 6. References

[1] O. J. R. Olarte, «Técnicas de inteligencia artificial utilizadas en el procesamiento de imágenes y su aplicación en el análisis de pavimentos,» *Revista EIA,* pp. 189-207, 2019.
DOI: https://doi.org/10.24050/reia.v16i31.1215

[2] Salima Nebti, «Inteligencia artificial de enjambre para reconocimiento facial,» *ScienceDirectet,* 2017.
DOI: https://doi.org/10.1016/j.swevo.2016.07.001

[3] Á. D. Mario, Optimización de una red convolucional para la detección de melanomas en imágenes demoscópicas, 2020.
DOI: https://doi.org/10.24054/rcta.v1i39.1378

[4] O. Palacios Prades, «Redes neuronales artificiales para el procesamiento de imágenes, una revisión de la última década,» *Escola d' Enginyeria (UAB),* 2017.
DOI: https://doi.org/10.26507/ponencia.1565

[5] J. A. Ramírez Q. y M. I. Chacón M., «Redes neuronales artificiales para el procesamiento de imágenes, una revisión de la última década,» *RIEEC,* 2016.
DOI: https://doi.org/10.4272/978-84-9745-246-5.ch6

[6] X. Song, S. Gao y C. Chen, «A multispectral feature fusion network for robust pedestrian detection,» *Alexandria Engineering Journal,* p. 73–85, 2021.
DOI: https://doi.org/10.1016/j.aej.2020.05.035

[7] J. Chahua J., «Predicción de deslizamientos desencadenados por lluvias con métodos de inteligencia artificial,» *lareferencia,* 2022.
DOI: https://doi.org/10.11144/javeriana.10554.61566

[8] E. Ferrante, «Inteligencia artificial para el análisis de imágenes médicas,» 2021.
DOI: https://doi.org/10.1016/j.swevo.2016.07.001

[9] E. Silva, Entrenamiento de la Red Neuronal Convolucional YOLO para objetos propio, 2020.
DOI: https://doi.org/10.14201/gredos.158770

[10] J. Redmon, S. Divvala, R. Girshick y A. Farhadi, You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2016.
DOI: https://doi.org/10.1016/j.iot.2023.100881

[11] C. Bonilla Carrión, Redes Neuronales, 2020.
DOI: https://doi.org/10.1016/j.bbe.2016.06.008

[12] J. Martinez, «Redes neuronales convolucionales en profundidad,» 2018.
DOI: https://doi.org/10.3390/app14146294

[13] Departamento de Matemática Aplicada, «Redes Neuronales Convolucionales,» 2021.
DOI: https://ieeexplore.ieee.org/document/7780460

[14] J. Redmon, S. Divvala, R. Girshick y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Dtection,» 2016.
DOI: https://doi.org/10.47460/uct.v27i118.686

[15] Y. Bengio, «Practical Recommendations for Gradient-Based Training of Deep Architectures,» 2012.
DOI: https://doi.org/10.35537/10915/90903

[16] R. Salas, «Redes Neuronales Artificiales,» 2004.
DOI: https://doi.org/10.35537/10915/90903

[17] R. Robalino, A. Getino y R. J., Matemática oculta bajo el proceso de aprendizaje en redes neuronales convolucionales..
DOI: https://doi.org/10.48550/arXiv.1506.02640

[18] R. Borja Robalino, A. Monleón Getino y J. Rodellar, Matemática oculta bajo el proceso de aprendizaje en redes neuronales convolucionales, 2022.
DOI: https://doi.org/10.1007/978-3-642-35289-8_26

[19] B. Montenegro y M. Flores Calero, «Pedestrian detection at daytime and nighttime conditions based on YOLO-v5,» *INGENIUS,* 2022.
DOI: https://doi.org/10.4272/978-84-9745-246-5.ch1

[20] R. Flores y J. Fernandez, «Las Redes Neuronales Artificiales Fundamentos teóricos y aplicaciones prácticas.».
DOI: https://doi.org/10.37811/cl_rcm.v6i5.3158

[21] L. Everingham, C. K. I. Van Gool, J. W. Williams y A. Zisserman., «The Pascal Visual Object Classes (VOC) Challenge," International Journal of Computer Vision,» 2010.
DOI: https://doi.org/10.37811/cl_rcm.v6i5.3158

[22] Nebti, Salima; Boukerram, A., «Swarm intelligence inspired classifiers for facial recognition,»
DOI: https://doi.org/10.1109/icsp62122.2024.10743933

[23] F. Yua, G. Xiang, F. Zhao, X. Wang, H. Liu, P. Lin y. Chen, «Improved YOLO-v5 model for boosting face mask recognition accuracy on heterogeneous IoT computing platforms,» 2023.
DOI: https://dialnet.unirioj|a.es/servlet/libro?codigo=395241

[24] P. Ebenezer, «Automated object and image level classification of TB images using support vector neural network classifier,» 2016.
DOI: https://doi.org/10.1007/s11263-009-0275-4