Inteligencia Artificial 28(76) (2025), 200-222
doi: 10.4114/intartif.vol28iss76pp200-222

INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Estimating the severity of coffee leaf rust using deep learning and image processing

Juan José Zuñiga Cajas[1], Oscar Daniel Peña Ramos[1], Emmanuel Lasso[1], Jacques Avelino[2], Juan Carlos Corrales[1], Cristhian Nicolás Figueroa Martínez[1]

[1] University of Cauca, Colombia
[2] CIRAD, France

[1] juanjosezuniga@unicauca.edu.co, poscar@unicauca.edu.co, eglasso@unicauca.edu.co, jcorral@unicauca.edu.co, cfigmart@unicauca.edu.co
[2] jacques.avelino@cirad.fr

Abstract The global coffee industry faces significant challenges from crop diseases, of which coffee leaf rust (CLR) caused by the fungus *Hemileia vastatrix*, stands out as one of the most damaging. Accurate assessment of disease severity is essential for applying effective control strategies. In response to this need, this study introduces a modern approach using deep learning and image processing techniques to identify and quantify CLR injury automatically. We developed thirteen models using convolutional neural networks, to classify lesions into different degrees of severity. It offers a promising alternative to conventional methods, especially under data-limited conditions, although some limitations remain in robustness across datasets. Manual rust detection requires close visual inspection of leaves, a laborious and error-prone process, especially in large cultivation areas. This challenge makes it harder to apply timely and effective disease management strategies.

Keywords: Rust, Coffee, Deep learning and disease detection.

1 Introduction

Coffee leaf rust (CLR) is one of the coffee industry's most damaging diseases. This disease is caused by the fungus *Hemileia vastatrix,* which is spread mainly by wind-borne spores, which can travel long distances, and by splashing rain, which facilitates infection in humid conditions. Hence, coffee growers require an accurate assessment of the severity of lesions caused by this disease to apply practical control strategies.

Detection of rust and other plant diseases has several limitations, mainly due to the complexities involved in accurate identification and the challenges posed by environmental factors. These limitations can hinder the effectiveness of traditional and modern detection methods, which impacts agricultural productivity. Coffee, essential to world trade and part of the cultural heritage of various communities, faces challenges such as CLR, a disease that causes significant economic losses and affects crop productivity. In Colombia, the 2008 epidemic resulted in a 30% reduction in production between 2008 and 2011 compared to 2007, while in Central America, the 2012-2013 epidemic reduced output by 16% as mentioned by Lasso et al. (2020) [1]. These crises directly affect the livelihoods of coffee farmers and agricultural workers, who must invest more in disease controls, worsening their economic situation [1].

Currently, rust detection methods are manual, based on the classification of healthy or infected leaves. This process is inefficient and error-prone due to subjectivity and variations in sampling methods, which limits comparability between studies and complicates the accurate measurement of disease incidence [1]. It highlights the need for more

ISSN: 1137-3601 (print), 1988-3064 (on-line)

precise and automated processes. Image processing and deep learning offer efficient solutions to classify and quantify leaf injury, improving interventions.
Automated detection of crop diseases, such as coffee rust, has seen significant advances through deep learning, but critical challenges remain. Approaches that leverage efficient CNN architectures such as EfficientNet-B0/B3 and vision transformers (ViT), which are widely used in remote sensing (RSI-SC) for their ability to handle local features and global dependencies as mentioned in Song et al. 2023 studies. [24, 25], which have shown promise in identifying lesions such as yellow spots and orange pustules by semantic segmentation and chromatic adjustments [1]. However, their accuracy is hampered by unbalanced datasets, variability in image capture due to shadows and illumination [2] and the need for large-scale data to train complex models such as ResNet or DenseNet [11]. In this sense, techniques such as hybrid CNN-ViT models, which combine local feature extraction and contextual modeling, or paradigms such as the Few-Shot Learning (FSL) mentioned in the study [18] could optimize performance with limited samples, overcoming the limitations of previous studies performed in controlled environments [3,4,5]. Furthermore, the integration of comprehensive tools for segmentation up to gravity quantification using attention mechanisms such as SE blocks and knowledge distillation as also mentioned by Song et al. (2024) is critical to develop robust systems for real-world environments, where visual complexity and noise demand adaptive solutions [14]. Thus, the fusion of advances in efficient architectures (CNN/ViT), FSL and integrated processing presents a viable avenue for improving coffee rust diagnostics, bridging the gap between research and practical applications.

Our study presents a modern approach that leverages deep learning and image processing techniques for automatic identification and quantification of coffee leaf rust (CLR) severity, addressing the challenges of manual assessment. We trained thirteen convolutional neural network (CNN) models to classify lesions by severity and developed a method that integrates ResNet, DenseNet, and (FSL) techniques. By implementing prototypical networks combined with FSL, our approach achieves a classification accuracy rate above 80% while reducing computational resources as it operates on a smaller dataset compared to conventional CNNs, improving generalization under data-limited conditions and optimizing model efficiency for precision agriculture, in particular enabling the deployment of lightweight models on mobile devices for field applications.

2 Methodology

This project adopted the CRISP-DM process model for structured and systematic development, proposing an advanced strategy for early detection of rust in coffee crops.

2.1 Understanding the Data

This study proposes a strategy for the early detection of rust in coffee plantations using a dataset containing 13,050 photographs of rust-infected coffee leaves of the *Caturra* variety taken from the underside of the leaf, as rust begins to evolve on this leaf surface [3]. The dataset includes images capturing various stages of fungus evolution in coffee plantations. The photographs range from completely healthy leaves to leaves with varying degrees of injury, providing a broad and varied representation of disease progression. It should be noted that the photos were taken as part of the PROCAGICA project, the Central American Program for the Integrated Management of Coffee Rust, as mentioned in the study [6], in which a specific image acquisition method was followed. Merle et al. (2020) documented the evolution of incipient lesions on coffee leaves using photographs taken with a Huawei mobile device in HD resolution. The images were captured under natural light, in climatic conditions representative of Costa Rica (humid tropical climate country), covering municipalities such as *Aserrí*, *Turrialba* and *Heredia*. The study revealed a higher number of samples in class 1 (leaves with early symptoms), which is explained by the methodological approach: by following the progression of infections until leaf drop, many did not reach advanced stages during the observation period.

2.2 Data Preparation

A. Data classification

We carried out a process of organizing and classifying the images, dividing the dataset into five distinct classes based on the degree of injury, as proposed by *Barrios et a*l. [7], facilitating a more precise and structured evaluation of plant health. This classification considers the affected leaf area. Lesions, start with small pale spots that turn yellow over time and eventually become intense orange spots caused by rust, according to *Lasso et al*. [1].

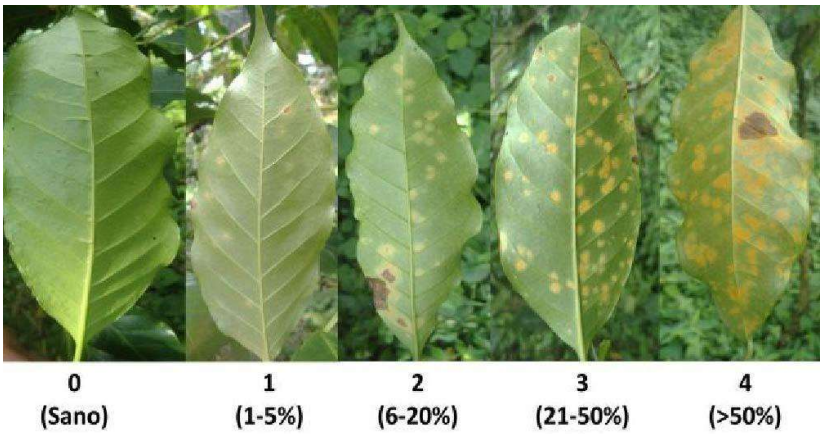

Figure 1. Phases of the evolution of the disease. Taken from: [1].

As shown in the figure, the five established classes are as follows:

- **Class 0:** Images of healthy coffee leaves without signs of rust.
- **Class 1:** Images where leaf injury by rust ranges from 1% to 5%.
- **Class 2:** Images with leaf injury by rust ranging from 6% to 20%.
- **Class 3:** Images where leaf injury by rust ranges from 21% to 50%.
- **Class 4:** Images with leaf injury by rust exceeding 50%.

B. Data Segmentation

After classification, we segmented the leaves to eliminate distracting factors for the training models. This segmentation was performed using the YOLOv8 and LabelMe tools since, on the one hand, YOLOv8 has a balance between speed and accuracy in object detection, allowing models to focus their analysis on relevant areas of interest and minimize background noise. In contrast, LabelMe is used for detailed and manual image labeling, facilitating the creation of customized datasets with accurate labeling of affected areas, thus improving training data quality and model robustness, as described in the study of Terven et al. (2023) [8].

Training the YOLO model involved a process that began with careful image selection. Next, we manually delineated each leaf on the selected images by drawing polygons around them using the LabelMe tool. These polygons were then converted to the appropriate file format for YOLO, and the data were reorganized to create the training set for the YOLO v8 model. This approach facilitated accurate image segmentation, removing unwanted elements and focusing detection on the leaves specifically. Seventy-three images were segmented from the leaf area recognition dataset to train the YOLOv8 model, resulting in a dataset in which the image was centered only on the leaf.

In addition, YOLOv8 was used in a local runtime environment to distinguish between healthy and damaged areas of coffee leaves, identifying and extracting areas of interest for more accurate analysis. Chromatic segmentation was also implemented to eliminate healthy areas and focus the study on the affected parts, thus improving the accuracy of injury quantification, which allowed the model to optimize its approach, focusing the analysis on the relevant areas and eliminating unwanted elements, simplifying the identification process and increasing training efficiency.

Following classification, we implemented a two-stage segmentation process to isolate relevant leaf features and optimize model training. Using YOLOv8 for its balanced speed-accuracy performance in object detection and LabelMe for precise manual annotation, we created a refined dataset focused specifically on leaf areas - removing background noise and distracting elements as demonstrated by Terven et al. (2023) [8]. The workflow began with careful image selection, followed by manual leaf delineation through polygonal annotations in LabelMe, which were then converted to YOLO-compatible formats. From the leaf recognition dataset, we processed 73 images to generate centered leaf-focused samples for YOLOv8 training. The model was subsequently deployed in a local runtime environment to perform dual functions: distinguishing between healthy and damaged leaf regions through chromatic segmentation (effectively filtering out healthy tissue), and precisely extracting areas of interest for targeted analysis. This combined approach of geometric and chromatic segmentation not only improved injury

quantification accuracy but also streamlined the identification process by eliminating irrelevant elements, thereby enhancing both training efficiency and analytical precision.

C. Data Augmentation and Balancing

Following the segmentation of the dataset images, an imbalance of data among the classes was evident, with *class 1* having a more significant number of images than other classes. Figure 2 shows the class imbalance, where classes 0, 2, 3 and 4 have fewer images than class 1.

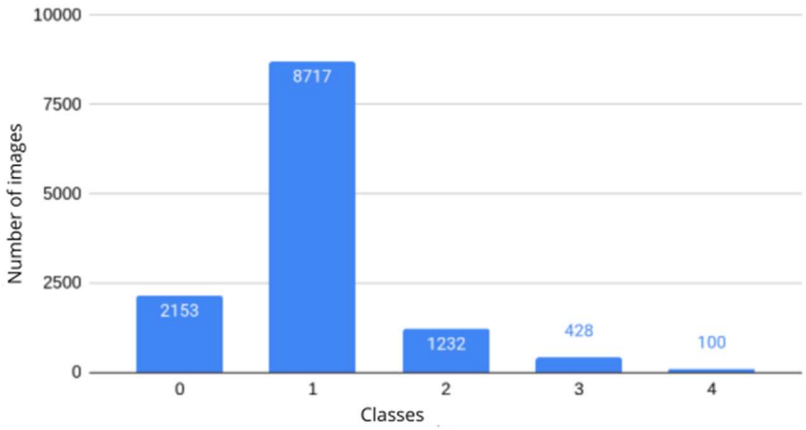

Figure 2. Class distribution. Own Source

To class balance, we adopted a data augmentation strategy, as demonstrated by *Escobedo et al*. [9]. Additionally, the study by *Nanni, L., et al.* [13] compared various data augmentation techniques, highlighting that geometric transformations such as rotation, translation, and scaling generate diverse perspectives. These techniques enable the model to adapt to different object orientations, sizes, and positions. Specifically, they simulate various angles of leaf observation, crucial for detecting and recognizing rust symptoms on leaves of differing sizes. Ultimately, this enhances the model's accuracy and reliability under varied conditions.

Therefore, based on the comparison of augmentation techniques presented by *Nanni, L., et al.* [13], we selected the geometric transformations strategy, which consists of obtaining new samples from an original sample performing rotations and axis inversions. This data augmentation enhances the dataset's generalization and diversity while preserving label consistency. It strengthens the model's robustness against variations in the orientation or scale of objects within the images. Geometric transformations, such as rotations and shifts, improve the model's robustness against object orientation and scale changes.

On the other hand, authors like *Shorten et al.* [10] mention that random cropping helps simulate various perspectives and improve the handling of partially visible objects. However, it may result in losing important information by cropping relevant image areas. Meanwhile, the study [11] states that brightness and contrast adjustment and noise injection simulate environmental variations, making the model more robust to lighting conditions or background noise changes. However, if applied excessively, these data augmentation techniques can degrade image quality and overly alter key features, negatively affecting the model's performance.

As a result of the data preparation phase, we obtained three datasets for training algorithms, each with different approaches to class balancing. The first approach ensured that classes with fewer data reached at least half the data of the most numerous classes, specifically class 1. The second approach equalizes the number of images among all classes. Finally, the third approach doubled the number of pictures for all classes based on the amount of data in the second approach. Thus, we obtained four datasets: the original unbalanced dataset and three augmented ones, divided as follows: 70% for training, 20% for validation, and 10% for testing, facilitating model evaluation.

2.3 Algorithm Selection

In this study, deep learning algorithms were selected, including variants of convolutional neural networks such as ResNet, MobileNet, DenseNet, Inception, and some versions of EfficientNet, for image classification under uniform hardware and training conditions. Studies [12,14] have shown good accuracy with these algorithms, highlighting the importance of large volumes of high-quality and relevant data, as emphasized in [15,16]. Additionally, an alternative approach was considered: Few-Shot Learning (FSL) using prototypical networks, which are particularly useful in smart agriculture due to the limited availability of data, as indicated in studies [17,18]. These techniques offer improved accuracy in object detection with smaller data volumes and resource optimization, outperforming conventional CNN methods, as demonstrated by studies [19,20] and the authors of [21,22]. We selected a list of the thirteen algorithms for training:

- EfficientNet
- DenseNet 121
- DenseNet 169
- DenseNet 201
- InceptionV3
- Xception
- MobileNet V1
- MobileNet V2
- ResNet50 V2
- ResNet101
- WideResNet
- ResNeXt
- FSL with prototypical networks.

2.4 Algorithm Training

The algorithms were trained with uniform criteria. Each algorithm was trained with 12 epochs and a learning rate equal to 0.001, using the Adam optimizer to balance stability and speed of convergence, taking advantage of a limited number of epochs and reducing the need for manual adjustments to maximize performance in the available training time. Each algorithm was trained in four cycles, one for each data set. Key variables such as loss metrics, accuracy, and recall in the test phase, were analyzed during the training and validation phases.

A detailed definition of each of these variables, along with their corresponding algebraic relationship, is provided below to provide a clear and precise understanding of the criteria used to evaluate and optimize the performance of the algorithms throughout the training process.

Recall and precision metrics are used to evaluate the model's performance. Each metric offers a different perspective on the model's effectiveness, depending on true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) identified in the predictions [8]. Definitions of each term and the methodology for calculating these metrics are explained below:

Precision: measures the proportion of correct predictions (both positive and negative) among the total cases, and is equivalent to the sum of true positives and true negatives over the total cases.

Precision = (TP +TN) / (TP + TN + FP + FN) [8]

Recall: measures the proportion of actual positives correctly identified by the model, equivalent to the rate of true positives over the total actual positive cases.

Recall = TP / (TP + FN) [8]

F_1-Score: combines precision and recall to evaluate the accuracy of a model in classification tasks, considering both false positives and false negatives, defined as follows, where P is Precision and R is Recall:

F_1-Score= 2 * (P*R/P + R) [8]

Additionally, the confusion matrix obtained from the preliminary model evaluation is considered a reference point to determine the model's performance and operational effectiveness.

We propose the model training process in Figure 3; it begins with data preparation, where data are collected, cleaned, and pre-processed for model training. It is followed by model initialization, evaluating whether a pre-trained model will be used or a new architecture will be defined from scratch.

If a pre-trained model is used, the existing model is loaded, and the output layer is modified to fit the specific task. Otherwise, if no pre-trained model is used, the complete architecture of the algorithm is defined, specifying its layers, neurons, and activation functions.

Next, the training process is established, including configuring input data, partitioning into training and validation sets, and selecting evaluation metrics. Subsequently, hyperparameters such as learning rate, number of epochs, and batch size are set.

Once all these elements are defined, the model is trained, where it learns from the data by adjusting its parameters. Finally, the best version of the model is saved based on its validation performance, thus concluding the process.

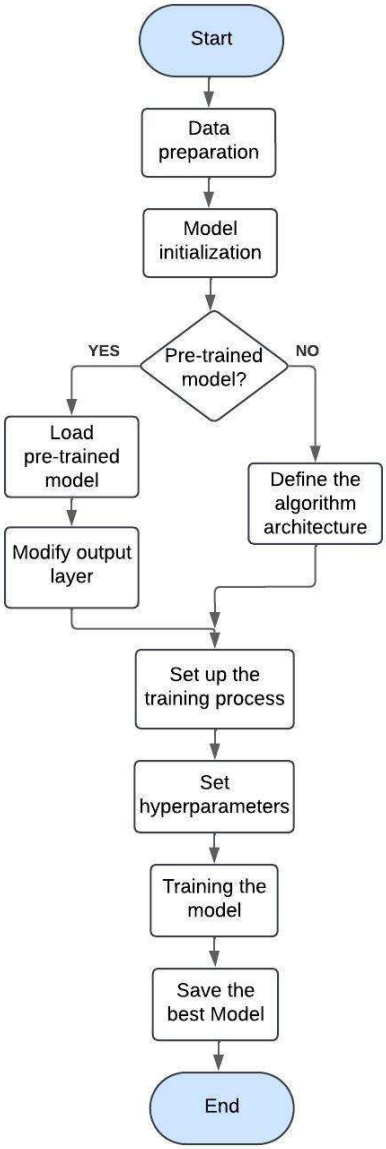

Figure 3. Training process of deep learning algorithms. Own source.

In implementing the models generated during the training of the algorithms, the model receives pre-processed images, generates a prediction value, and decodes it according to the class defined in the training set, allowing the severity estimate to be visualized. These estimates correspond to five classes, ranging from class 0, which represents healthy leaves, to class 5, which indicates leaves more than 50% affected, covering the range of CLR severity.

3 Analysis

After training the selected algorithms, it was observed that the FSL model obtained better results than conventional CNN models, necessitating a detailed analysis of the results based on the previously mentioned metrics. This initial evaluation provides a preliminary view of the algorithms' behavior and overall performance, an essential step before their application in real scenarios.

3.1 Algorithms' training

For model validation, it is considered to implement it in two different environments: first, in a laboratory setting, where various factors can be controlled, allowing precise analysis under ideal conditions. Subsequently, the model was tested in a natural, less controlled environment, reflecting coffee producers' real challenges when inspecting leaves affected by the fungus and evaluating the injury level. This strategy ensures an understanding of the model's effectiveness in both ideal conditions and the complexity of the real world, providing valuable insights for its optimization and fine-tuning before final implementation.

The following graphs show the performance of each of the thirteen selected algorithms as a function of the three metrics: loss, precision, and recall. These algorithms were trained using the four available data sets.

These images are sectioned considering the data augmentation distribution of the original dataset mentioned at the end of section 2.2 in the following order:

- **A:** Original Dataset.
- **B:** Dataset with data augmentation, maintaining a ratio.
- **C:** Balanced dataset with data augmentation.
- **D:** Balanced dataset with data augmentation with twice as many images.

A. **Original Dataset.**

This dataset did not undergo data augmentation; the classes present the original data quantity.

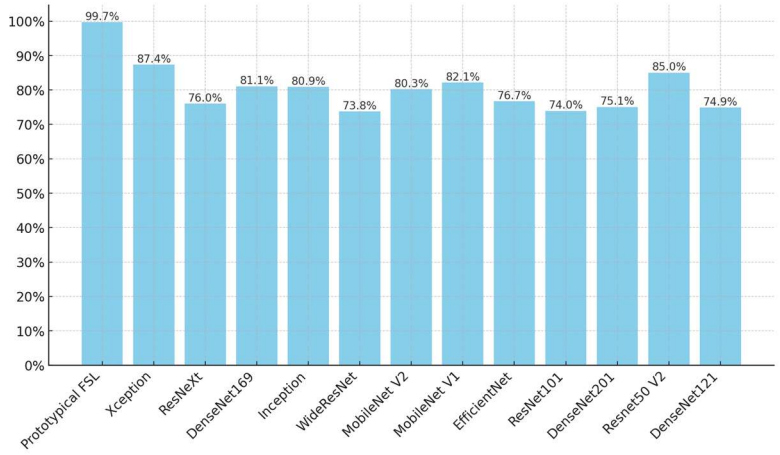

Figure 4. Training accuracy, Dataset A. Own source.

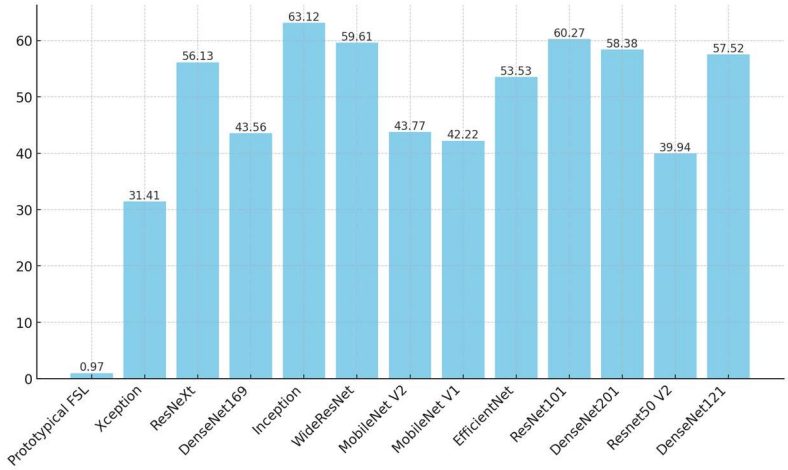

Figure 5. Loss, Dataset A. Own source.

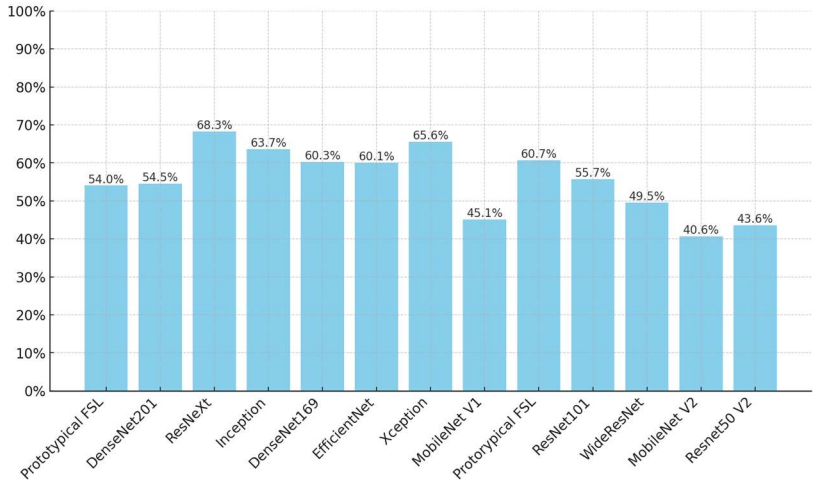

Figure 6. Recall, Dataset A. Own source.

On the original dataset (Figures 4–6), the FSL model outperformed all other models in accuracy and loss. While several CNNs achieved recall above 60%, FSL still led overall, confirming its effectiveness under minimal data augmentation.

The combined analysis of accuracy and loss metrics, which reflect training quality, and recall, which measures the rate of true positives, highlights the overall superior performance of the FSL model compared to the other models assessed.

B. **Data augmentation ratio.**

To class balance, the dataset is augmented proportionally to increase the minority classes to half the size of the majority class, ensuring training avoids overfitting.

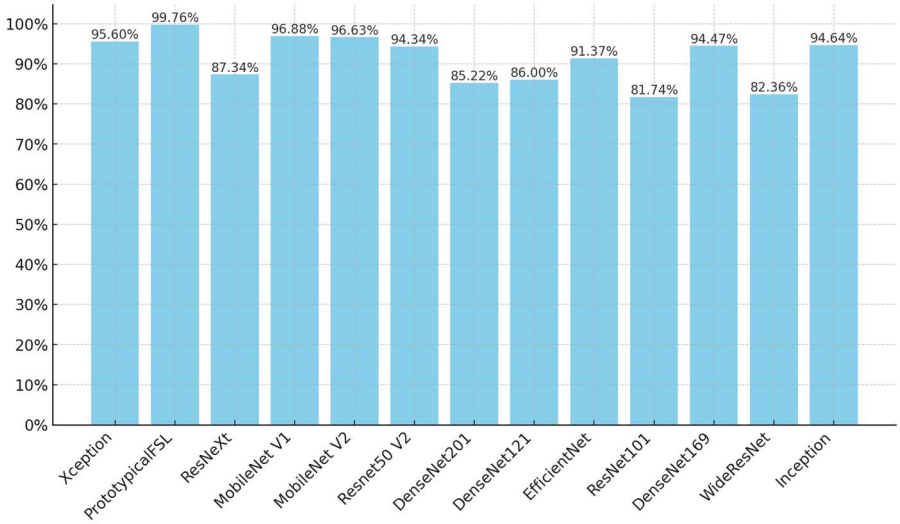

Figure 7. Training accuracy, Dataset B. Own source.

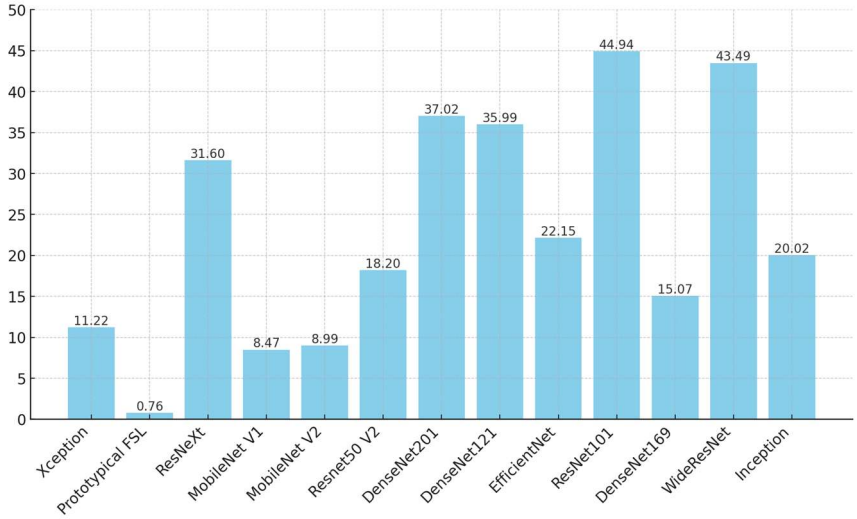

Figure 8. Loss, Dataset B. Own source.

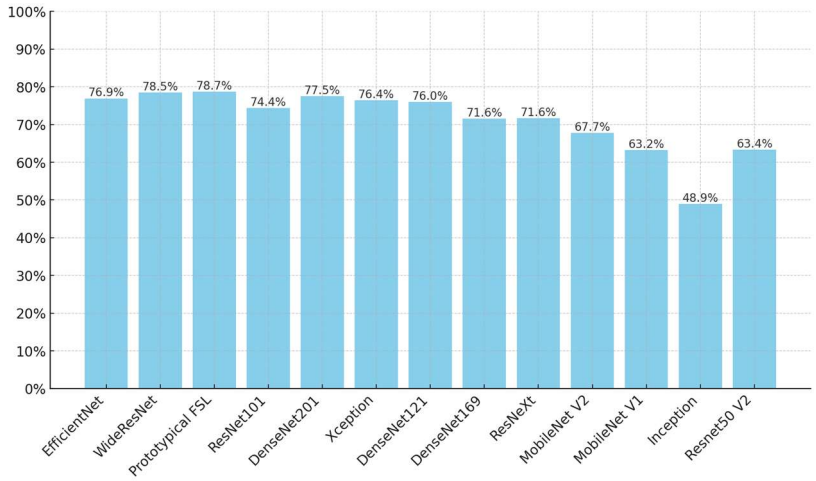

Figure 9. Recall Dataset B. Own source.

With moderate data augmentation (Dataset B), most models achieved over 80% accuracy, and recall generally improved (Figures 7–9). FSL continued to lead, followed closely by DenseNet and EfficientNet, while WideResNet and ResNet101 consistently lagged behind.

In this process, when an increase in data is applied while maintaining a ratio with the majority class, an overall improvement in the algorithm training is observed. It also contributes to the decrease in losses, which allows a better representation of positive cases. Models such as Inception, MobileNetV1, ResNet50V2, and prototypical networks with FSL stand out for their better performance in the training process and better identification of positive cases.

C. **Balanced dataset with data augmentation.**

Data augmentation is performed for classes with fewer samples than class 1, aiming to reach an equal number of samples among the classes, adjusted to have half the data of class 1.

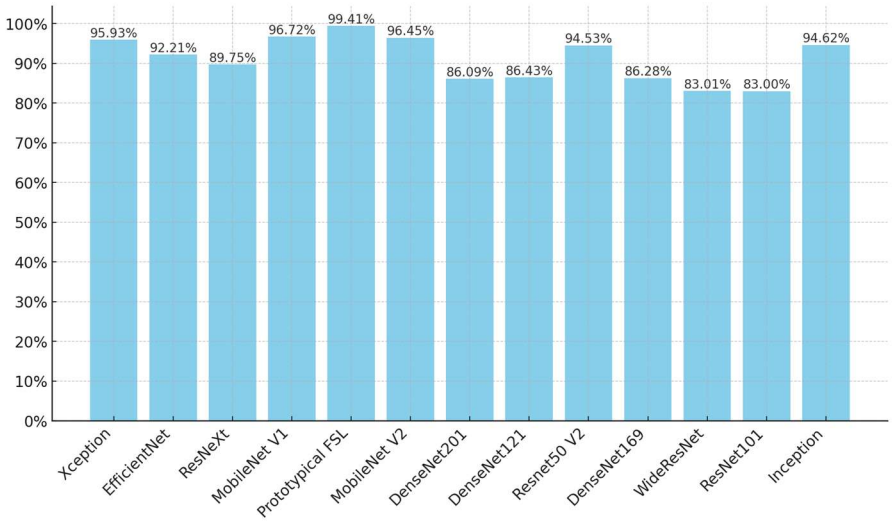

Figure 10. Training accuracy, Dataset C. Own source.

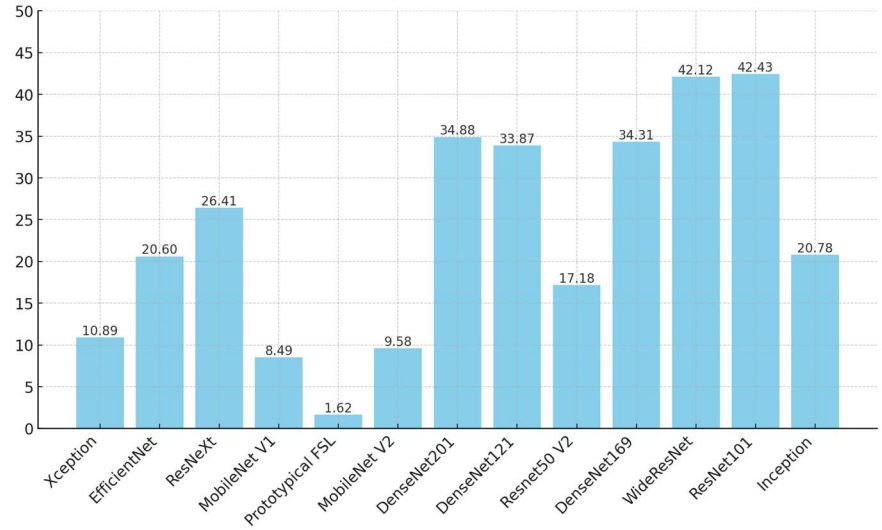

Figure 11. Loss, Dataset C. Own source.

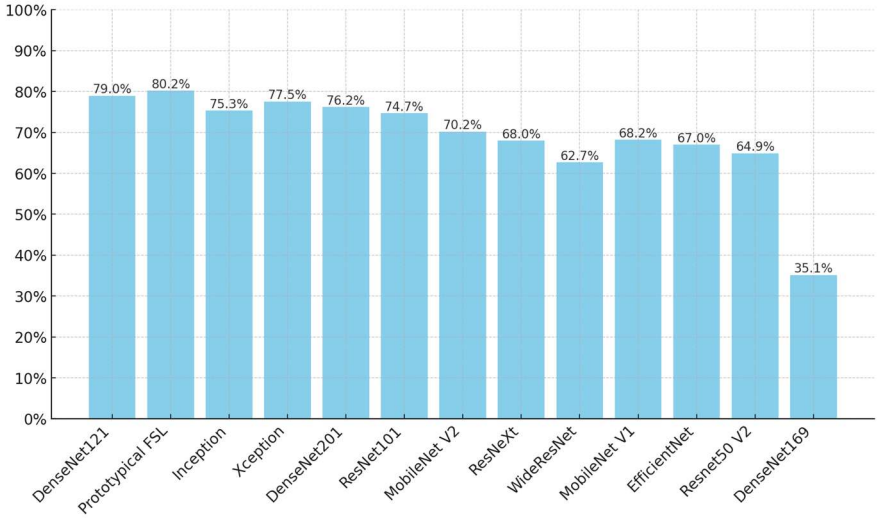

Figure 12. Recall, Dataset C. Own source.

Balancing the dataset further (Dataset C) resulted in higher performance across nearly all models. FSL again obtained the best recall (80%), with DenseNet121 and Xception also showing strong performance (Figures 10–12).

The models performed better overall by applying data augmentation while maintaining a balanced data set. Accuracy increased significantly and losses decreased, highlighting the good performance of ResNet50V2 and the FSL model with prototypical networks. This adjustment allowed a better representation of positive cases, indicating a better ability to correctly identify positive instances, especially in models such as DenseNet and MobileNet.

D. **Balanced dataset with data augmentation, doubling the image numbers.**

Data augmentation is performed for all classes to obtain a balanced dataset.

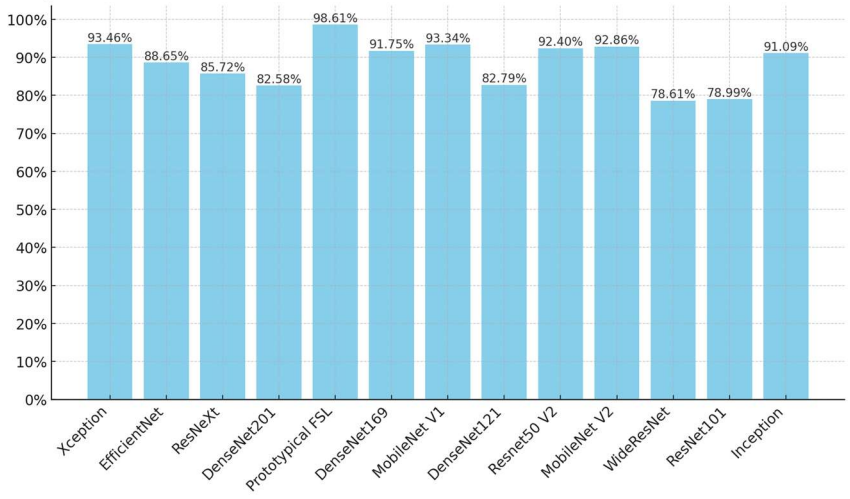

Figure 13. Training accuracy, Dataset D. Own source.

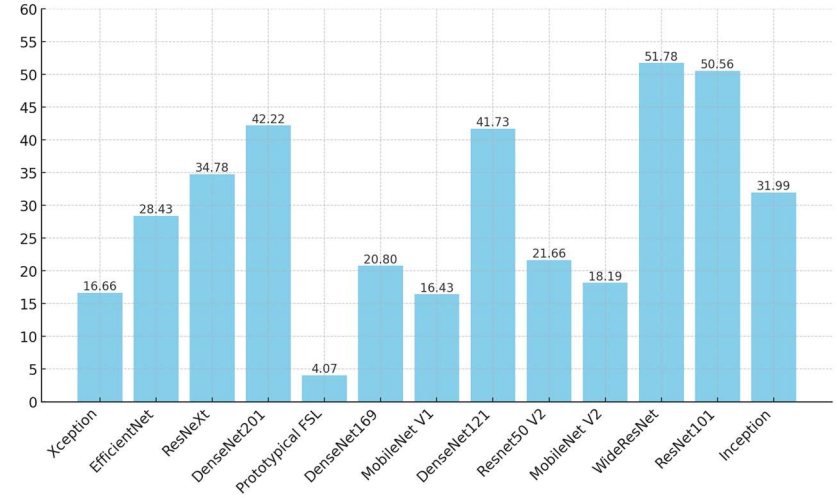

Figure 14. Loss, Dataset D. Own source.

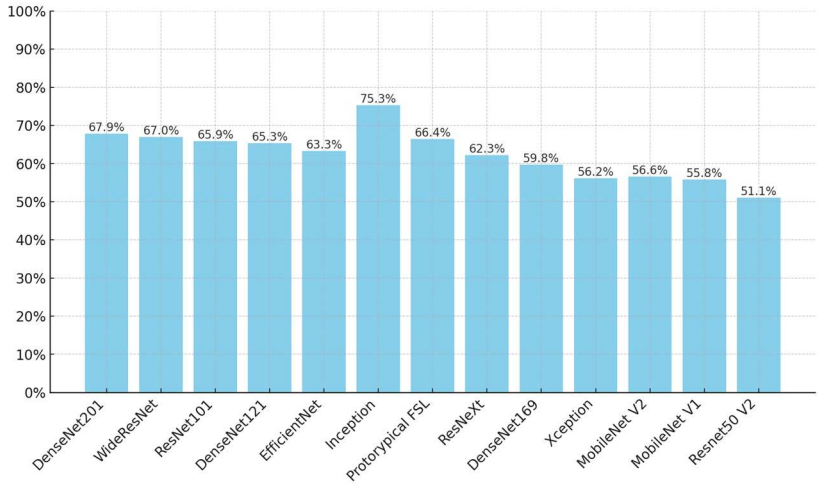

Figure 15. Recall Dataset D. Own source.

Doubling the dataset size (Dataset D) yielded diminishing returns for most models. While InceptionV3 had the highest recall, overall performance dropped slightly, and FSL showed signs of overfitting (Figures 13–15).

Using a balanced dataset with twice as many images resulted in a marked improvement in training. Data doubling allowed models such as InceptionV3, MobileNetV2 and prototypical networks with FSL to achieve higher precision and lower losses. In addition, improved recall was observed, meaning that these models could more effectively identify positive cases, thus improving their performance compared to previous sections.

3.2 Comparative Model Performance

The results of this research highlight the quality of model training when analyzing precision and loss values. High values for precision and low values for loss are indicators of adequate training, with FSL performance standing out in particular. In the recall plots, which evaluate the rate of true positives and the ability of the models to correctly detect these instances, models such as ResNeXt, Xception, Inception, DenseNet169, EfficientNet, and prototypical networks with FSL obtained outstanding scores.

The application of data augmentation techniques, maintaining a ratio with the majority class, resulted in overall improvements in the performance of the models. This strategy contributed to decreased losses and allowed for better representation of positive cases. Models such as Inception, MobileNetV1, ResNet50V2, and prototypical networks with FSL demonstrated superior performance in identifying positive cases during training.

Balancing the dataset through data augmentation techniques showed a positive impact, increasing precision and reducing losses. This adjustment particularly benefited models such as ResNet50V2 and prototypical networks with FSL, demonstrating an improved ability to identify positive cases correctly. Models such as DenseNet and MobileNet also excelled in this context, achieving robust performance.

Duplication of the dataset, creating an equilibrium with twice as many images, produced a noticeable improvement in training. Models such as InceptionV3, MobileNetV2 and prototypical networks with FSL achieved higher precision rates and lower losses, evidencing improved recall and more efficient positive case identification capability. These results highlighted the superior performance of the prototypical FSL networks, which achieved precision rates as high as 99.76% and losses as low as 0.0075, reflecting their reliability and low error-proneness. Regarding recall, the prototypical networks achieved 80.22% on the C data set, demonstrating a robust ability to identify the most of positive instances.

Advanced techniques can be applied to provide ideas for minimizing segmentation noise in real-time sensing applications, and both pre-and post-segmentation processes can be optimized. Although practices such as image pre-processing and data augmentation have already been implemented to train more robust models, results can be improved by using specialized networks such as UNet and Mask R-CNN. Also, combining multimodal models, including depth sensors, and applying post-processing models can help clean up noise and artifacts.

The following is a detailed series of graphs illustrating the performance of one of the trained algorithms, specifically an FSL algorithm with prototypical networks. This algorithm denotes balance among the three mentioned metrics, making its training the most effective.

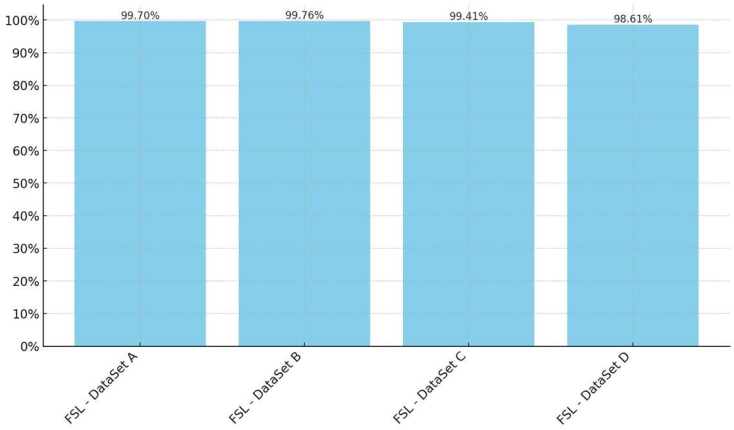

Figure 16. FSL training accuracy. Own source.

As illustrated in Figure 16, minimal differences exist between the outstanding performance in terms os precision when training with different datasets. It has been observed that proportionally increasing the data quantity to ensure an adequate volume for training, without exceeding the threshold that could cause overfitting, optimizes performance. Compared to those obtained by training the algorithm with other datasets, this improvement in results underscores the importance of a careful balance in the data quantity used. This approach prevents overfitting risk and contributes to more effective algorithm generalization, enhancing its ability to make accurate predictions in varied scenarios.

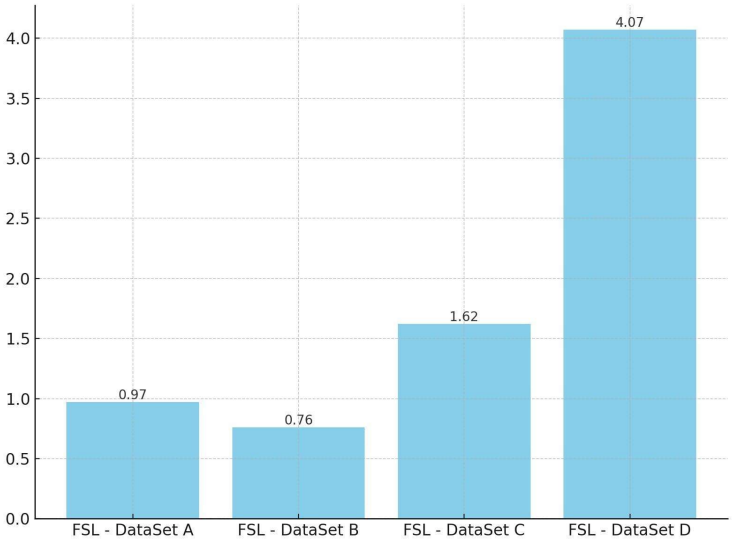

Figure 17. FSL loss. Own source

A similar trend is observed in the precision context, where the algorithm trained with the improved dataset through proportional data augmentation shows a subtle advantage. Surprisingly, the unmodified dataset showed better loss results than at least two enriched datasets. This finding suggests that data augmentation, though beneficial in many scenarios, is unnecessary for achieving satisfactory loss results. This analysis reinforces the idea that the effectiveness of data augmentation strategies can vary considerably, depending on the specific nature of each dataset and the inherent characteristics of the applied algorithms.

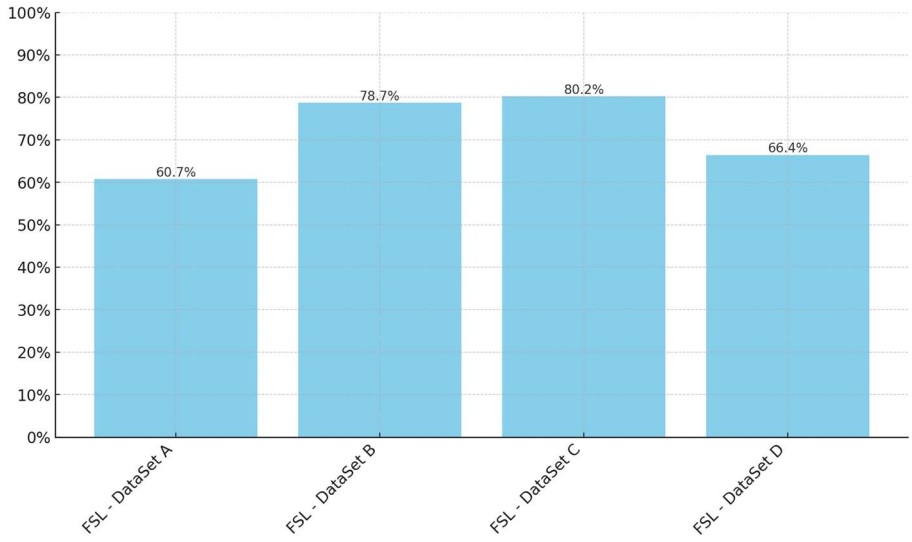

Figure 18. FSL recall. Own source.

Figure 18 shows notable differences in recall values obtained by the model trained with enriched datasets through augmentation techniques. Despite this, the dataset that did not experience data augmentation presented a reasonable recall level. This phenomenon highlights how data augmentation can positively influence the model's ability to correctly identify positive cases, although the improvement may be marginal in some cases.
On the other hand, in the specific analysis of the FSL algorithm implemented with prototypical networks, it stands out that its optimal performance is achieved when trained with a moderately sized dataset containing the necessary information for making accurate predictions. This finding suggests that, for this type of algorithm, the quality and relevance of the dataset can be more determinant than the sheer volume of data, highlighting the importance of careful data selection and preparation to maximize model effectiveness.

4 Evaluation

We use the recall, precision, and F_1 score for the model evaluations. Tables 1 to 4 show the algorithm results differentiated by obtained data (A, B, C, and D datasets) and discriminating by augmentation technique applied.
For each model, these metrics were calculated using the confusion matrix and considering the values of TP (true positives), TN (true negatives), FP (false positives), and FN (false negatives).

Table 1. Results of Algorithms - Dataset A. Own source.

Algorithm Results – Dataset A			
ALGORITHM	**RECALL (%)**	**PRECISION (%)**	**F_1-SCORE**
DenseNet121	0,5405	0,7053	0,5644
DenseNet201	0,5455	0,6648	0,5675
ResNeXt	0,6832	0,6413	0,6583
Inception	0,6366	0,6357	0,5782
DenseNet169	0,6028	0,6317	0,6039
EfficientNet	0,6006	0,6285	0,5859
Xception	0,6559	0,6210	0,6210
MobileNet V1	0,4511	0,6206	0,4876
Prototypical FSL	0,6074	0,6075	0,6074
ResNet101	0,5569	0,5791	0,5397
WideResNet	0,4954	0,5638	0,5039
MobileNet V2	0,4065	0,5466	0,4350
Resnet50 V2	0,4363	0,5141	0,4430

Table 2. Results of Algorithms - Dataset B. Own source.

Algorithm Results – Dataset B			
ALGORITHM	**RECALL (%)**	**PRECISION (%)**	**F_1-SCORE**
EfficientNet	0,7692	0,8013	0,7725
WideResNet	0,7849	0,7923	0,7865
Prototypical FSL	0,7872	0,7872	0,7872
ResNet101	0,7435	0,7870	0,7640
DenseNet201	0,7750	0,7853	0,7772
Xception	0,7643	0,7797	0,7602
DenseNet121	0,7602	0,7693	0,7541
DenseNet169	0,7158	0,7684	0,7310
ResNeXt	0,7164	0,7442	0,7141
MobileNet V2	0,6773	0,7149	0,6837
MobileNet V1	0,6319	0,6812	0,6328
Inception	0,4892	0,6769	0,4946
Resnet50 V2	0,6336	0,6702	0,6360

Table 3. Results of Algorithms - Dataset C. Own source.

Algorithm Results – Dataset C			
ALGORITHM	**RECALL (%)**	**PRECISION (%)**	**F_1-SCORE**
DenseNet121	0,7896	0,8107	0,7945
Prototypical FSL	0,8022	0,8022	0,8022
Inception	0,7535	0,7931	0,7542
Xception	0,7755	0,7878	0,7763
DenseNet201	0,7622	0,7870	0,7640
ResNet101	0,747	0,7862	0,7488
MobileNet V2	0,7021	0,7161	0,7032
ResNeXt	0,6795	0,7144	0,6724
WideResNet	0,6266	0,7117	0,5995
MobileNet V1	0,6822	0,7103	0,6847
EfficientNet	0,67	0,7079	0,6622
Resnet50 V2	0,6489	0,6752	0,6499
DenseNet169	0,3513	0,6101	0,3000

Table 4. Results of Algorithms - Dataset D. Own source.

Algorithm Results – Dataset D			
ALGORITHM	**RECALL (%)**	**PRECISION (%)**	**F_1-SCORE**
DenseNet201	0,6788	0,7028	0,6796
WideResNet	0,6699	0,6968	0,6716
ResNet101	0,6587	0,6908	0,6600
DenseNet121	0,6533	0,6845	0,6535
EfficientNet	0,6331	0,6806	0,6371
Inception	0,7535	0,6742	0,6536
Prototypical FSL	0,6640	0,6640	0,6640
ResNeXt	0,6225	0,6562	0,6241
DenseNet169	0,5976	0,6316	0,5924
Xception	0,5616	0,6315	0,5639
MobileNet V2	0,5662	0,5994	0,5679
MobileNet V1	0,5583	0,5883	0,5555
Resnet50 V2	0,5106	0,5684	0,5097

Next, the applied technique in the datasets that offer the best results in terms of precision is identified. Table 2 shows that DenseNet121 and FSL with prototypical networks present a higher precision value than other techniques, so

they are chosen for further studies. Algorithms InceptionV3 and DenseNet201 are also selected as the subsequent best-performing algorithms in this technique

Once the algorithms DenseNet121, FSL with prototypical networks, InceptionV3, and DenseNet201 are selected for their superior performance in precision. A study evaluates their behavior when trained with different data percentages. Previously, these algorithms were trained with 70% of the data. This study proposes that the algorithms with 20%, 40%, and 60% of the data from dataset B to observe how their performance varies with different training data quantities. Each algorithm was trained with these percentages, as shown in Table 5.

Table 5. Results of Testing Data Percentages. Own source.

Results of Testing Data Percentages				
Algorithm	**Data (%)**	**RECALL (%)**	**PRECISION (%)**	**F_1-SCORE**
DenseNet121	20	0,6856	0,7182	0,6759
Prototypical FSL	20	0,7976	0,7976	0,7976
DenseNet201	20	0,7591	0,7927	0,7631
InceptionV3	20	0,6690	0,6954	0,6583
DenseNet121	40	0,7341	0,7547	0,7314
Prototypical FSL	40	0,7851	0,7851	0,7851
DenseNet201	40	0,6960	0,7591	0,6892
InceptionV3	40	0,7111	0,8002	0,7062
DenseNet121	60	0,7589	0,8085	0,7609
Prototypical FSL	60	0,7897	0,7898	0,7897
DenseNet201	60	0,7172	0,7621	0,7186
InceptionV3	60	0,7592	0,7636	0,7604
DenseNet121	70	0,7896	0,8107	0,7945
Prototypical FSL	70	0,8022	0,8022	0,8022
DenseNet201	70	0,7622	0,7870	0,7640
InceptionV3	70	0,7535	0,7931	0,7542

To verify the performance of the models trained with different data percentages, a completely different dataset than the initially implemented one is used, pre-processed, and cleaned for use in each of the four models generated with each defined data percentage.

The *RoCoLe* dataset was used exclusively to perform tests on the generated models and evaluate their generalization capability when processing images with features different from those of the training set. These differences make *RoCoLe* the right candidate for implementation in this type of evaluation.

The *RoCoLe* dataset [23] contains images like those worked on in the project. Specifically, images in different stages of rust, it is already labeled and taken from the underside of the leaf, which is a very important factor. After implementing the models, this information is represented in Table 6.

Table 6. Results, Testing Data Percentages, RoCoLe Dataset. Own source.

.

Results of Testing Data Percentages				
Algorithm	**Data (%)**	**RECALL (%)**	**PRECISION (%)**	**F_1-SCORE**
DenseNet121	20	0,3874	0,4075	0,3378
Prototypical FSL	20	0,4735	0,4735	0,4735
DenseNet201	20	0,4828	0,5443	0,4514
InceptionV3	20	0,4570	0,4085	0,4006

DenseNet121	40	0,4340	0,3868	0,3994
Prototypical FSL	40	0,5198	0,5197	0,5197
DenseNet201	40	0,4183	0,4337	0,3872
InceptionV3	40	0,3041	0,4186	0,2425
DenseNet121	60	0,4170	0,3965	0,3671
Prototypical FSL	60	0,5204	0,5204	0,5204
DenseNet201	60	0,4242	0,4719	0,4068
InceptionV3	60	0,3472	0,3301	0,3026
DenseNet121	70	0,5044	0,4176	0,4500
Prototypical FSL	70	0,4948	0,4949	0,4948
DenseNet201	70	0,4081	0,3789	0,3658
InceptionV3	70	0,3569	0,3356	0,2786

The data presented in Tables 5 and 6 show a decrease in precision variables, which can be inferred as due to the change in coffee variety, image noise, or image quality, as the dataset used for model training is a dataset of images of the Caturra variety, while the *RoCoLe* dataset images are of the Robusta variety.

In addition, training was performed using 10% of the dataset to evaluate the model's behavior against overfitting. This training took 10 minutes and 45 seconds on the Kaggle platform, utilizing a GPU P100 with 73 GB of storage, 29 GB of RAM, and 16 GB of dedicated GPU memory. The results obtained presented slightly lower metrics compared to those in Table 5, with 77.22% recall, 78.46% precision, and 77.45% F1 score, compared to 75.91% recall, 79.27% precision and 76.31% F1 score of the model trained with a more extensive data set. Despite this difference, the results indicate that the model retains good generalization ability even when a smaller fraction of the data is used.

Analysis of Robustness to Segmentation Variability

Table 5 shows that the algorithms with prototypical FSL and DenseNet201 networks maintain stable precision with different percentages of training data. For this reason, these two algorithms and the models generated with 20% of the data are chosen to carry out a new series of tests. It is proposed that the robustness of the models already trained be analyzed to classify images with variations in their segmentation. For this purpose, the behavior of the models is analyzed against manually segmented pictures. Then some unsegmented photos are added to the dataset and evaluated using the trained model.

First Test

As a first step, 20 coffee leaf images from the RoCoLe dataset were manually segmented to meet the characteristics of the original dataset, such as similarity in framing, cropping, illumination, and image quality. These images were segmented in an assisted manner using Facebook's segmentation website, Segment Anything (SAM), thus extracting the region of interest cleanly. Subsequently, we used these 20 pre-processed images to implement the algorithms, obtain the predictions, and calculate this implementation's recall, Precision, and F_1-Score metrics.

Table 7. Test of manually segmented images. Own source.

Test with 20 manually segmented images			
Algorithm	**RECALL (%)**	**PRECISION (%)**	**F_1-SCORE**
Dataset RoCoLe			
Prototypical FSL	0,4735	0,4735	0,4735
DenseNet201	0,4828	0,5443	0,4514
SEGMENTED IMAGES MANUALLY			
Prototypical FSL	0,5923	0,5923	0,5922
DenseNet201	0,5500	0,7927	0,5333

The results presented in Table 7 show that the precision values of each model are slightly higher with this type of manually segmented images. When comparing these results with the precision values of the RoCoLe dataset extracted from Table 6 corresponding to the percentage of training data 20, it can be observed that this new test considerably improves the precision of the evaluated models. It, can be interpreted as a better performance of the models with images that follow the pattern of the training dataset and are susceptible to the variations that the photos may present in their segmentation process.

Second Test

A second test was performed based on the results obtained in the first test. It combined the original pre-processed dataset with images extracted from the RoCoLe dataset without pre-processing. The data selection for the noise study of the DenseNet201 and Prototypical Networks with FSL algorithms corresponds to 10% of the test data used throughout the project. Figure 19 shows the original percentage of data (Blue bar) and the noise induced by the images without the RoCoLe dataset pre-processing (Yellow bar).

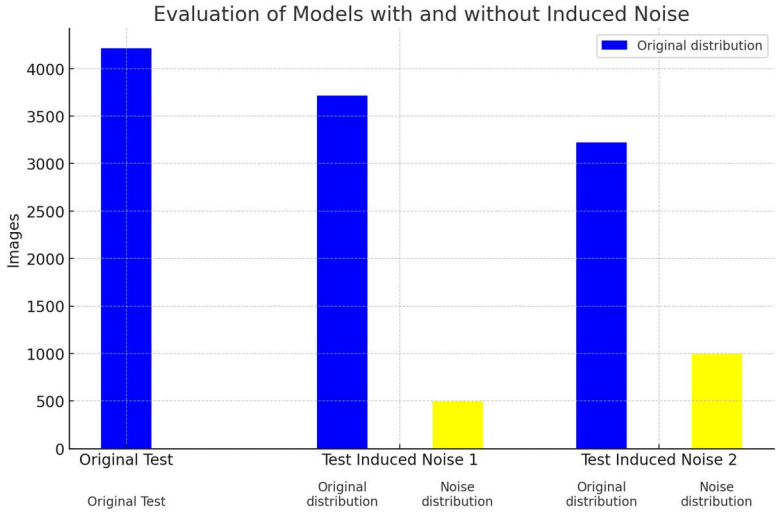

Figure 19. Data used in robustness analysis. Own source.

In Table 8, the "Original Test" column data corresponds to 10% of the original dataset used to train the algorithms. The "Induced Noise Test 1" and "Induced Noise Test 2" columns expose noise percentages for Original, RoCoLe and dataset respectively. Each induced test complements each other to achieve 10% of the original dataset size in order to compare them.

Table 8. Evaluation of models with and without induced noise. Own source.

Evaluation of models with and without induced noise					
	Original Test	**Induced Noise Test 1 (Original Dataset)**	**Induced Noise Test 1 (RoCoLe Dataset)**	**Induced Noise Test 2 (Original Dataset)**	**Induced Noise Test 2 (vDataset)**
Total by dataset	4215	3716	498	3221	997
Percent	10	8.82	1.18	7.64	2.36

The procedure to determine the metrics of the algorithms consisted of constructing the confusion matrix and, calculating the relevant metrics from it. The "Original Test" set was evaluated as a first step to establish a baseline of the model performance without any noise. Then, the "Induced Noise Test 1" was performed, where 498 images from the "Original Test" set were replaced by unsegmented images from the RoCoLe dataset. It allowed quantifying the affectation in the model performance when processing images outside its training. Finally, the "Induced Noise Test 2" was performed, which integrated 997 photos from the RoCoLe dataset to observe the effect of a more considerable noise on the dataset provided to the model.

Table 9. Results evaluation of algorithms with and without induced segmentation noise. Own source.

Results evaluation of algorithms with and without induced segmentation noise

ALGORITHM	RECALL (%)	PRECISION (%)	F_1 SCORE
ORIGINAL TEST			
DenseNet201	0,76	0,7927	0,7631
Prototypical Networks + Few Shot Learning	0,80	0,7976	0,7976
INDUCED NOISE TEST 1			
DenseNet201	0,76	0,73	0,6912
Prototypical Networks + Few Shot Learning	0,80	0,5667	0,5666
INDUCED NOISE TEST 2			
DenseNet201	0,76	0,672	0,6105
Prototypical Networks + Few Shot Learning	0,80	0,4589	0,4589

Table 9 shows the results of the induced noise tests. A decrease in the precision of the algorithms is noted as noise is added to the data sets. However, the precision of the prototypical networks with FSL is affected more than that of DenseNet201. This second test demonstrates that the generated models are susceptible to the segmentation noise in the images the model processes once trained. However, the dense networks are based on convolutional neural networks; they proved to be more robust than the prototypical networks with FSL concerning the segmentation noise that may be present in the images being processed.

5 Discussion and Future Perspectives

Our study has successfully demonstrated the effectiveness of deep learning and image processing techniques in assessing Coffee Leaf Rust (CLR) severity, while the integration of Few-Shot Learning (FSL) represents a significant breakthrough in overcoming the limitations of traditional CNN-based approaches. FSL's ability to achieve high precision with limited datasets offers transformative potential for agricultural disease management, providing an adaptable solution that maintains robustness against data variability while optimizing foliar damage detection and quantification.

Building on these foundations, we identify several critical pathways for advancement. First, we propose developing a mobile API to deploy these models directly to farmers' smartphones, creating accessible tools for precision agriculture that support real-time crop management decisions. This mobile implementation will require careful evaluation of processing times to ensure practical feasibility for field use.

While the FSL model showed superior performance on curated and balanced datasets, its recall on the RoCoLe dataset was notably lower. This indicates potential limitations in generalization and robustness, particularly under noisy conditions. In contrast, DenseNet models demonstrated better stability across such variations.

The research also highlights the need to expand beyond CLR-specific solutions. We will investigate model adaptation for other coffee diseases (including iron spot and rooster's eye) and potentially different crops, while addressing key challenges in field image quality, coffee variety recognition, and disease characteristic differentiation. The generalizability demonstrated by our FSL approach suggests promising potential for these expansions.

Although FSL models achieved high performance on curated datasets, their lower recall on the RoCoLe dataset and greater sensitivity to segmentation noise suggest they may require further fine-tuning or domain adaptation to match the robustness of DenseNet in real-world conditions.

A particularly innovative direction involves integrating climate data (temperature, humidity) with visual analysis to enable predictive capabilities for disease outbreaks. This multi-modal approach, combined with our work on disease progression stage recognition, could significantly enhance early warning systems and preventive measures.

Together, these advances position our research as an innovative benchmark in the evolution of smart agriculture, offering farmers more precise and accessible tools to address complex phytosanitary challenges while improving overall agricultural management practices. The success of our FSL implementation lays the groundwork for the development of robust and adaptable solutions that meet the diverse needs of modern coffee farming.

6 Conclusions

During the training of the algorithms, class imbalance was observed to significantly influence the models. The classes used for training had the following distribution: 2,153 images for class 0, 8,717 for class 1, 1,232 for class 2, 428 for class 3, and only 100 for class 4. Given this disparity, it was essential to apply image data augmentation techniques, such as generating new samples by transforming existing ones. Class balancing is crucial when working with highly unbalanced data sets, as it ensures a fair representation and prevents the model from being biased towards the majority classes, thus improving generalization. To optimize this process, a single type of data augmentation in varying proportions was applied to the original data set. This approach allowed us to evaluate how different levels of augmentation affected the performance of the algorithm, which helped identify the right balance between improving data diversity and model efficiency. As a result, the capacity of the algorithms improved without compromising their generalization capability.

One of the main aspects observed is that the evaluator must specialize in the specific pathology monitored, since lack of specialization increases variability in the results. Comparing the performance of human evaluators in rust classification with deep learning models is not feasible, because human classification is highly subjective and depends on multiple external factors. In fact, in the experiments conducted in this work, it was found that the evaluations of human evaluators did not reach a consensus. At the same time, the deep learning models presented consistency and facilitated the replication of the results.

While prototypical networks with FSL provide clear advantages in efficiency and generalization with limited data, the results show that their recall decreases and their robustness is lower than DenseNet when tested on external or noisy datasets like RoCoLe. This highlights the need to further improve their adaptability and resilience to image variability for real-world applications.

Experiments with varying proportions of the training dataset revealed that DenseNet121, DenseNet201 and prototypical networks with FSL maintain good performance even with limited data, demonstrating their potential for small-scale applications. In particular, DenseNet121 and DenseNet201 showed consistent accuracy even when trained with only 20% of the original data, taking advantage of their dense connectivity and pre-training on ImageNet to generalize efficiently. Surprisingly, DenseNet201 outperformed DenseNet121 in data-poor scenarios (54.43% vs. 40.75% accuracy with 20% data), highlighting the advantage of its deeper architecture for feature reuse. Meanwhile, prototypical networks with FSL achieved 79.27% accuracy with the same constraints, demonstrating good performance under ideal conditions, though further refinement is needed to improve robustness under real-world variability. These results underscore the importance of architecture selection in limited data contexts and validate the role of FSL in effective detection of coffee tree leaf diseases, especially when dense networks, which although not originally designed for small datasets, demonstrate their adaptability through transfer learning and hierarchical feature extraction.

Acknowledgements

We would like to express our sincere gratitude to Isabel Merle for her dedication and commitment in collecting the data used in this study. Her meticulous and professional work, carried out as part of her doctoral research with CIRAD, was fundamental to the development and quality of this work. We deeply appreciate her effort and collaboration at every stage of the process.

References

[1] Lasso, E., Corrales, D. C., Avelino, J., de Melo Virginio Filho, E., & Corrales, J. C. (2020). Discovering weather periods and crop properties favorable for coffee rust incidence from feature selection approaches. *Computers and Electronics in Agriculture*, *176*. https://doi.org/10.1016/j.compag.2020.105640.

[2] Soares, A. da S., Vieira, B. S., Bezerra, T. A., Martins, G. D., & Siquieroli, A. C. S. (2022). Early Detection of Coffee Leaf Rust Caused by Hemileia vastatrix Using Multispectral Images. *Agronomy*, *12*(12). https://doi.org/10.3390/agronomy12122911.

[3] Boulent, J., Foucher, S., Theau, J., & St-Charles, P. L. (2019). Convolutional neural networks for the automatic identification of plant diseases. Frontiers in Plant Science, 10, 941. https://doi.org/10.3389/fpls.2019.00941.
[4] Plazas, J.E., Rojas, J.S., Corrales, D.C., Corrales, J.C. (2016). Validation of Coffee Rust Warnings Based on Complex Event Processing. In: Gervasi, O., et al. Computational Science and Its Applications -- ICCSA 2016. ICCSA 2016. Lecture Notes in Computer Science(), vol 9789. Springer, Cham. https://doi.org/10.1007/978-3-319-42089-9_48
[5] Guo, L. ., Marlisah, E. ., Ibrahim, H. ., & Manshor, N. . (2023). A review of few-shot image recognition using semantic information . Review of Computer Engineering Research, 10(2), 55–69. https://doi.org/10.18488/76.v10i2.3472
[6] Merle, I., Tixier, P., de Melo Virginio Filho, E., Cilas, C., & Avelino, J. (2020). Forecast models of coffee leaf rust symptoms and signs based on identified microclimatic combinations in coffee-based agroforestry systems in Costa Rica Forecast models of coffee leaf rust symptoms and signs based on identified microclimatic combi-nations in coffee-based agroforestry systems in Costa. *Crop Protection*, *130*, 105046. https://doi.org/10.1016/j.cropro.2019.105046ï.
[7] Barrios, M., Cerda, R., de Sousa, K., Diagramación, E., & Jiménez, R. (n.d.). *Prevención y control de la roya del café*.
[8] Terven, J., Cordova-Esparza, D. M., Ramirez-Pedraza, A., & Chavez-Urbiola, E. A. (2023). *Loss Functions and Metrics in Deep Learning*. http://arxiv.org/abs/2307.02694
[9] Escobedo, R., Heras, J., Milella, A., & Marani, R. (2023). 50. Data augmentation techniques for grape bunch segmentation in natural images. Precision agriculture ’23, 409-414
[10] Shorten, C., Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. J Big Data 6, 60 (2019). https://doi.org/10.1186/s40537-019-0197-0
[11] J. T. Barron, "A General and Adaptive Robust Loss Function," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019, pp. 4326-4334, doi: 10.1109/CVPR.2019.00446. keywords: {Training;Adaptive systems;Image synthesis;Neural networks;Manuals;Minimization;Robustness;Deep Learning;3D from Single Image;Computer Vision Theory;Image and Video Synthesis;Low-level Vision;Statistic},
[12] Álvarez-Sánchez, D.-E. ., Arévalo, A. . ., Benavides , I. F. ., Salazar-González, C. ., & Betancourth, C. . . (2023). Use of Trained Convolutional Neural Networks for Analysis of Symptoms Caused by Botrytis fabae Sard. Revista De Ciencias Agrícolas, 40(1), e1198. https://doi.org/10.22267/rcia.20234001.198
[13] Nanni, L., Paci, M., Brahnam, S., & Lumini, A. (2021). Comparison of Different Image Data Augmentation Approaches. Journal of imaging, 7(12), 254. https://doi.org/10.3390/jimaging7120254
[14] SMART FARMING USING DEEP LEARNING. (2023). International *Research Journal of Modernization in Engineering Technology and Science*. https://doi.org/10.56726/IRJMETS39923
[15] Hayit, T., Erbay, H., Varçın, F., Hayit, F., & Akci, N. (2021). Determination of the severity level of yellow rust disease in wheat by using convolutional neural networks. *Journal of Plant Pathology*, *103*(3), 923–934. https://doi.org/10.1007/s42161-021-00886-2
[16] Chavarro, A. F., Renza, D., & Ballesteros, D. M. (2023). Influence of Hyperparameters in Deep Learning Models for Coffee Rust Detection. *Applied Sciences (Switzerland)*, *13*(7). https://doi.org/10.3390/app13074565
[17] Sun, J., Cao, W., Fu, X., Ochi, S., & Yamanaka, T. (2023). Few-shot learning for plant disease recognition: A review. *Agronomy Journal*. https://doi.org/10.1002/agj2.21285
[18] Xinfeng Li, Shuai Xiao, Paul Kumar, and Bunyamin Demir "Data-driven few-shot crop pest detection based on object pyramid for smart agriculture," *Journal of Electronic Imaging* 32(5), 052403 (23 March 2023). https://doi.org/10.1117/1.JEI.32.5.052403
[19] Argüeso, D., Picon, A., Irusta, U., Medela, A., San-Emeterio, M. G., Bereciartua, A., & Alvarez-Gila, A. (2020). Few-Shot Learning approach for plant disease classification using images taken in the field. *Computers and Electronics in Agriculture*, *175*. https://doi.org/10.1016/j.compag.2020.105542
[20] Mu, J., Feng, Q., Yang, J., Zhang, J., & Yang, S. (2024). Few-shot disease recognition algorithm based on supervised contrastive learning. *Frontiers in Plant Science*, *15*. https://doi.org/10.3389/fpls.2024.1341831
[21] Faisal, M., Ashtiani, S.-H. M., Teo, J., & Ragu, N. (n.d.). *Sustainable Food Processing, a section of the journal Frontiers in Sustainable Food Systems Object detection and classification using few-shot learning in smart agriculture: A scoping mini review*.
[22] Tassis, L. M., & Krohling, R. A. (2022). , *6*, 55–67. https://doi.org/10.1016/j.aiia.2022.04.001
[23] Parraga-Alava, J., Cusme, K., Loor, A., & Santander, E. (2019). RoCoLe: A robusta coffee leaf images dataset for evaluation of machine learning based methods in plant diseases recognition. *Data in Brief*, *25*. https://doi.org/10.1016/j.dib.2019.104414

[24] Huaxiang Song, Yong Zhou. Simple is best: A single-CNN method for classifying remote sensing images[J]. Networks and Heterogeneous Media, 2023, 18(4): 1600-1629. doi: 10.3934/nhm.2023070
[25] Song, H., et al.: Efficient knowledge distillation for hybrid models: a vision transformer-convolutional neural network to convolutional neural network approach for classifying remote sensing images. IET Cyber-Syst. Robot. e12120 (2024). https://doi.org/10.1049/csy2.12120
[26] Song, H. (2023). A Leading but Simple Classification Method for Remote Sensing Images. *Annals of Emerging Technologies in Computing*, *7*(3), 1–20. https://doi.org/10.33166/AETiC.2023.03.001