Classification of Orthopoxvirus with Deep Learning in reduced data scenarios using resampling techniques

Darío Santos, Daniel Rivero, and Alejandro Puente-Castro

Faculty of Computer Science, CITIC, Universidade da Coruña, 15071 A Coruña, Spain
Correspondence: dario.santos@udc.es

DOI: *https://doi.org/10.17979/spu.23.c39*

Abstract: In recent years, monkeypox has become a growing global threat, where early diagnosis is essential for its control. This work explores the use of Deep Learning techniques applied to skin image analysis to improve the detection and classification of this disease compared to other similar ones. A highly unbalanced dataset of 770 images is used, so resampling techniques such as SMOTE and SMOTEENN are applied. The objective is not only to compare the performance of different Deep Learning models, but also to measure the impact on classification produced by the use of resampling strategies. It also seeks to identify the best combination to support automatic diagnosis in clinical and epidemiological contexts.

1 Introduction

Monkeypox (mpox) belongs to the large DNA genus known as Orthopoxvirus. This disease is the focus of attention because it is the most recent and least known within the genus, with the rest being very well treated or eradicated, as in the case of smallpox. In addition to mpox, chickenpox and measles will also be analyzed, two diseases with symptoms very similar to mpox, which can complicate diagnosis by healthcare professionals. To correctly detect these diseases, the use of deep learning techniques is proposed, whose models use neural network architectures based on the human brain. These models are trained with large datasets, which are then used to detect diseases. To correctly detect these diseases, the use of deep learning techniques is proposed, whose models employ neural network architectures based on the human brain. These models are trained with large datasets, from which they learn directly, avoiding manual feature extraction. The aim is to achieve accurate and rapid classification of these diseases by analyzing skin images of infected patients. Specifically, there is currently an open and standardized dataset consisting of skin images of patients with monkeypox, chickenpox, and measles, as well as healthy individuals. This dataset will be used to train the models. This collection is quite imbalanced between classes, greatly affecting the effectiveness and accuracy of the classification. Therefore, this study also examines the capabilities of different resampling techniques to improve class identification in this collection. Specifically, the techniques used are Synthetic Minority Oversampling Technique (SMOTE) and Synthetic Minority Oversampling Technique combined with Edited Nearest Neighbors (SMOTEENN).

2 State of the art

Recent literature shows a growing interest in the use of Deep Learning models for monkeypox classification using skin images. Several studies have used both custom CNNs and pre-trained

models (VGG, ResNet, Inception, MobileNet, among others), achieving accuracy metrics above 90%. However, recurring methodological limitations have been identified, especially the inappropriate use of data augmentation in test sets, as in (Alhasson et al., 2023) and (Shateri et al., 2025), or on the original dataset before its partitioning, as in the case of (Bala et al., 2023). This compromises the validity of the results by introducing dependence between training and testing. Additionally, some studies choose to simplify classification into binary problems, such as monkeypox vs. non-monkeypox in (Pal et al., 2023) and (Attallah, 2023), or normal cases vs. cases with pox in (Almutairi, 2022), which favors the results due to the existing imbalance between classes. Despite extensive exploration of classification architectures and strategies, no research has been found that analyzes the impact of balancing techniques such as SMOTE or SMOTEENN on the MSID dataset used in this work. Therefore, the need for this study is justified, which proposes to evaluate these techniques while maintaining a strict separation between training and testing.

3 Materials and methods

3.1 Deep Learning models

During this study, five CNNs or Convolutional Neural Networks will be used and analyzed. These are: AlexNet, ResNet18, DenseNet121, MobileNetV2, and UNet. Additionally, tests will be conducted with a Long Short-Term Memory (LSTM), which is a Recurrent Neural Network, as well as with Vision Transformers (ViT) and a GAN.

3.2 Metrics

The metrics used in this work for the evaluation of the models, which are also the most commonly used in the studies reviewed in State of the art and in other research works on medical image classification in different domains, are: accuracy, precision, recall and F1-score. In this study, special emphasis is given to the F1-score, since it balances *precision* and *recall*, which is important in contexts where the dataset is unbalanced, as well as in medical problems, where both false positives and false negatives have significant consequences. Subsequently, various statistical methods are applied to the F1-score values obtained, such as ANOVA, Welch's ANOVA, or the Kruskal-Wallis test (always subject to the preconditions and assumptions of each method). These techniques are used to determine whether there are significant differences in the performance of the models. Tukey's HSD (Honestly Significant Difference) is then applied to compare the models in case significant differences exist. Finally, confidence intervals for the F1-score are calculated using the Student's t-distribution. These analyses provide a more robust evaluation of the results obtained.

3.3 Resampling techniques for class balancing

Resampling techniques aim to balance the classes of a dataset and prevent models from leaning towards categories with more examples. There are three main approaches: oversampling, undersampling, and a combination of both.

- SMOTE: generates synthetic samples in minority classes through interpolation, which reduces bias and overfitting.
- ENN: removes inconsistent instances in majority classes to improve class separation, although it may excessively reduce the training set.
- SMOTEENN: combines both strategies, first applying SMOTE and then cleaning with ENN.

3.4 Dataset description

In this work, the "Monkeypox Skin Images Dataset" (MSID) developed by Bala et al. (2023), available on Kaggle (Bala, 2022), is used. It consists of 770 RGB images in PNG format with dimensions of 224 × 224 pixels, classified into four categories: *Chickenpox*, *Measles*, *Monkeypox*, and *Normal*. The images come from different parts of the human body, ranging from close-ups of the skin lesions to more distant views. The dataset includes images from various sources and shows geographic, phenotypic, and anatomical diversity. An important limitation of the dataset is the class imbalance, since the *Normal* category contains more than three times the number of examples of *Measles*. This imbalance may bias the models toward the majority classes, making it difficult to detect the underrepresented ones. To mitigate this problem, resampling techniques such as SMOTE and SMOTEENN are proposed in order to improve the ability of the models to correctly classify all diseases.

3.5 Data preparation

The different preprocessing transformations were carried out using the utilities provided by the Torchvision package. The images were resized to 224 × 224 pixels, as this is the dimension required by many of the architectures used. Next, normalization was applied based on the mean ($[0.485, 0.456, 0.406]$) and standard deviation ($[0.229, 0.224, 0.225]$) of the three RGB channels of each pixel in the ImageNet dataset. This transformation is applied because this is the dataset used for the pretraining of most of the models employed. It helps maintain the distribution of RGB values with which the models were pretrained, on which the weights depend, leading to better learning. For the application of SMOTE and SMOTEENN, each image is flattened into a one-dimensional vector, converting it into an *array*. Once applied, it is necessary to reconstruct the data back into the original format.

4 Results

PyTorch implementations were used for the models AlexNet, ResNet, MobileNet, DenseNet, ViT (16×16), and LSTM, with the first five pretrained on the ImageNet database. For UNet, a publicly available implementation was used (Medium, 2024). Finally, the GAN model was manually implemented using PyTorch functions in order to build a discriminator that classifies the images into different classes. A *stratified 5-fold cross-validation* strategy was employed to increase the reliability of the performance estimation and minimize the risk of random partitioning affecting the evaluation. Five folds were used as they provide a balance between computational cost and statistical stability. Stratification also helps ensure a more robust evaluation. For the GAN model, a publicly available training implementation was used (Lindernoren, 2025). In this process, the generator was trained so that its generated images were as realistic as possible, while in this specific case, the discriminator was trained to classify both original and generated images. Real labels were assigned to original images, while random labels were assigned to generated images. The hyperparameters used, obtained through a previous exploratory process, can be seen in Table 1. These allow the model to be trained in a balanced way, without overfitting.

Table 1: Hyperparameter configuration used for the training of the models in the first iteration.

Hyperparameter	**Value**
Maximum epochs	50
Stopping strategy	*Early stopping* (no improvement in 8 epochs)
Optimizer	SGD (Stochastic Gradient Descent)
Loss function	Cross Entropy Loss
Initial learning rate	0.001
Learning rate reduction	Factor 0.1 if no improvement in 3 epochs
Weight decay	10^{-4}

4.1 Results with the original dataset

Table 2: Overall results on the test set with the original data

Model	**Accuracy (%) (mean ± std)**	**Precision (mean ± std)**	**Recall (mean ± std)**	**F1-score (mean ± std)**
AlexNet	88.57 ± 1.67	0.8881 ± 0.0156	0.8857 ± 0.0167	0.8842 ± 0.0154
ResNet	89.48 ± 1.50	0.8977 ± 0.0138	0.8948 ± 0.0150	0.8927 ± 0.0137
MobileNet	90.52 ± 2.34	0.9088 ± 0.0241	0.9052 ± 0.0234	0.9035 ± 0.0230
LSTM	49.61 ± 3.66	0.4489 ± 0.0608	0.4961 ± 0.0366	0.4401 ± 0.0420
ViT	90.52 ± 1.67	0.9077 ± 0.0181	0.9052 ± 0.0167	0.9037 ± 0.0161
DenseNet	90.13 ± 1.56	0.9033 ± 0.0181	0.9013 ± 0.0156	0.8993 ± 0.0166
UNet	40.00 ± 2.08	0.3040 ± 0.1423	0.4000 ± 0.0208	0.2562 ± 0.0498
GAN	56.98 ± 7.13	0.5944 ± 0.0548	0.5698 ± 0.0713	0.5623 ± 0.0747

The metrics obtained on the test set for each model using the original dataset can be seen in Table 2. It shows the mean and standard deviation of the studied metrics, calculated from the results obtained in the 5 folds. After performing the corresponding analyses of the metrics, it was concluded that ViT was the best model, with an average *F1-score* of 0.9037 or 90.37% across the 5 folds. Moreover, this result is quite reliable, since for a 95% confidence level, the interval has a width of only 0.045. ViT achieved very good values thanks to the use of patches, which allow it to model long-range relevant patterns, such as skin lesions distributed throughout the image. UNet did not perform well because this architecture is more oriented toward segmentation tasks. LSTM has better performance with sequential data. The low values of the GAN are due to the fact that the generated images had random labels, which penalized the discriminator when classifying images similar to a certain class but without that class label, leading to poor training. As can be seen in Figure 1, which shows the average *F1-score* per class across the 5 folds obtained with ViT, the majority classes *Monkeypox* and *Normal* achieved higher values. This was expected and demonstrates that the model fails to learn the characteristics of the minority classes, being unable to classify them correctly. It will be analyzed whether the subsequent use of resampling techniques can help improve the results.

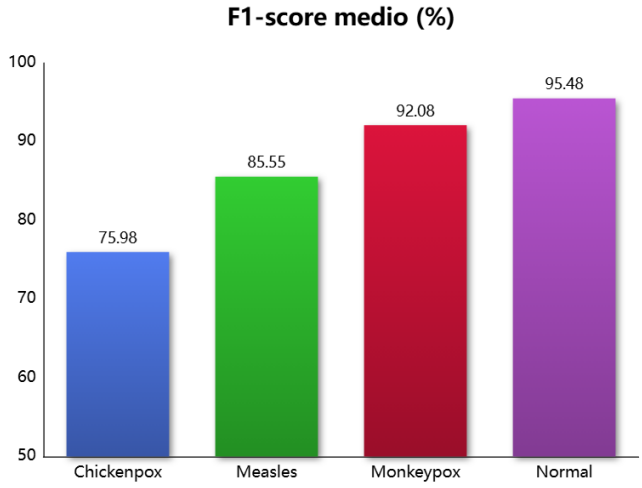

Figure 1: Bar chart with the average *F1-score* of each class for ViT

4.2 Results using SMOTE

As detected, the imbalance present in the dataset causes the models to focus more on learning the characteristics of the majority classes, producing poor performance for the underrepresented classes. Therefore, we propose analyzing the benefits obtained by balancing the different classes using synthetic samples. For this purpose, the SMOTE oversampling technique is applied, considering the 3 nearest neighbors for sample cleaning. The metrics obtained on the test set for each model using SMOTE on the training set, while keeping the test set with real data and the original class distribution, are shown in Table 3.

Table 3: Overall results on the test set using SMOTE in the training set

Model	**Accuracy (%) (mean ± std)**	**Precision (mean ± std)**	**Recall (mean ± std)**	**F1-score (mean ± std)**
AlexNet	68.31 ± 2.64	0.8184 ± 0.0107	0.6831 ± 0.0264	0.6942 ± 0.0225
ResNet	77.66 ± 3.35	0.8307 ± 0.0218	0.7766 ± 0.0335	0.7831 ± 0.0316
MobileNet	75.06 ± 2.48	0.7998 ± 0.0206	0.7506 ± 0.0248	0.7429 ± 0.0298
LSTM	44.81 ± 4.33	0.4666 ± 0.0519	0.4481 ± 0.0433	0.4285 ± 0.0525
ViT	67.92 ± 9.95	0.8208 ± 0.0272	0.6792 ± 0.0995	0.7000 ± 0.0871
DenseNet	84.16 ± 3.17	0.8561 ± 0.0242	0.8416 ± 0.0317	0.8406 ± 0.0315
UNet	38.96 ± 9.43	0.5168 ± 0.1366	0.3896 ± 0.0943	0.3602 ± 0.1166
GAN	53.24 ± 6.37	0.5734 ± 0.0415	0.5324 ± 0.0637	0.5325 ± 0.0657

The metrics worsened considerably compared to the first experiment with the original data. After analyzing these metrics, DenseNet was found to be the best model, with an average *F1-score* of 0.8406 or 84.06% across the 5 *folds*, as shown in Table 3, compared to 90.37% *F1-score* in the previous experiment. Additionally, for a 95% confidence level, the interval has a width of 0.09 for DenseNet. This model achieved the best results due to several key characteristics that make it effective for image classification, such as dense connectivity between layers. This

allows the reuse of learned features, helps mitigate gradient vanishing, and achieves better information propagation. SMOTE may generate a large number of samples in ambiguous regions where different classes overlap, causing the models to misclassify them and causing lower metric values.

Furthermore, an experiment was conducted to demonstrate the effect on metrics when SMOTE is applied before splitting the data. That is, with synthetic samples generated and fully balanced classes also in the test set. A mean *F1-score* of 0.9622 or 96.22% was obtained with MobileNet, compared to 84.06% *F1-score* when SMOTE was applied correctly, but these values are not reliable. When SMOTE is applied before partitioning, synthetically generated samples that depend on others (due to interpolation) may appear in the test set. Therefore, this test set is not independent from the training set, inflating results since the model encounters biased data. The test set should be an independent collection with samples never seen by the model, maintaining the same class proportion as the original dataset.

4.3 Results using SMOTEENN

SMOTEENN is proposed to remove noisy or ambiguous samples that can be generated by SMOTE, analyzing whether this would help the model better learn the different classes. The metrics obtained on the test set for each model using SMOTEENN on the training set, while keeping the test set with real data and the original class distribution, are shown in Table 4.

Table 4: Overall results on the test set using SMOTEENN in the training set

Model	Accuracy (%) mean ± std	Precision mean ± std	Recall mean ± std	F1-score mean ± std
AlexNet	42.08 ± 6.80	0.8007 ± 0.0194	0.4208 ± 0.0680	0.4293 ± 0.0844
ResNet	36.10 ± 6.17	0.8038 ± 0.0480	0.3610 ± 0.0617	0.3679 ± 0.0764
MobileNet	44.68 ± 7.01	0.8175 ± 0.0295	0.4468 ± 0.0701	0.4834 ± 0.0705
LSTM	20.39 ± 1.86	0.4453 ± 0.1153	0.2039 ± 0.0186	0.1381 ± 0.0302
ViT	20.52 ± 4.08	0.4917 ± 0.1585	0.2052 ± 0.0408	0.1346 ± 0.0501
DenseNet	57.92 ± 3.75	0.8302 ± 0.0228	0.5792 ± 0.0375	0.6191 ± 0.0380
UNet	15.84 ± 1.91	0.0379 ± 0.0089	0.1584 ± 0.0191	0.0588 ± 0.0135
GAN	45.84 ± 3.59	0.6188 ± 0.0204	0.4584 ± 0.0359	0.4827 ± 0.0331

The results with SMOTEENN worsened significantly compared to SMOTE or the original dataset. After analyzing the corresponding metrics, DenseNet was found to be the best model, with an average *F1-score* of 0.6191 or 61.91% across the 5 *folds*. Additionally, for a 95% confidence level, the interval has a width of 0.11 for DenseNet.

5 Conclusions

The study successfully achieved all its objectives through the development of an experimental framework that enabled the comparison of Deep Learning models and the analysis of resampling techniques. After reviewing the state of the art, advances and limitations in monkeypox classification were identified, and architectures capable of distinguishing between chickenpox, measles, monkeypox, and normal cases with high accuracy were proposed, reaching an F1-score of 90.37% with stratified K-fold cross-validation. The main challenge identified was the class imbalance, which motivated the study of SMOTE and SMOTEENN. However, their application did not improve the results. Nevertheless, the objective of analyzing these techniques

was fulfilled, and it was also demonstrated how, when misapplied, they can artificially inflate metrics, rendering them unreliable. The modular, reusable system developed shows great potential as support for clinical diagnosis and in educational environments.

6 Future work

Some potential future directions to improve the developed framework and the accuracy in classifying skin diseases include the creation of a graphical user interface to facilitate its use by professionals without technical knowledge. Additionally, the integration of clinical and genomic data could allow the development of more precise multimodal models, as well as the expansion of the dataset through collaboration with medical centers that can provide real cases. Another avenue is the use of multitask models capable of simultaneously predicting different aspects of the disease. Finally, the application of decentralized learning approaches, such as Federated Learning, would enable training networks without sharing sensitive data and developing models adapted to local needs.

Bibliography

H. F. Alhasson, E. Almozainy, M. Alharbi, N. Almansour, S. S. Alharbi, and R. U. Khan. A deep learning-based mobile application for monkeypox detection. *Applied Sciences*, 13(23):12589, 2023.

S. A. Almutairi. Dl-mdf-oh2: optimized deep learning-based monkeypox diagnostic framework using the metaheuristic harris hawks optimizer algorithm. *Electronics*, 11(24):4077, 2022.

O. Attallah. Mondial-cad: Monkeypox diagnosis via selected hybrid cnns unified with feature selection and ensemble learning. *Digital Health*, 9:20552076231180054, 2023.

D. Bala. Monkeypox skin images dataset (msid), 2022. URL *https://www.kaggle.com/dsv/3971903*.

D. Bala, M. S. Hossain, M. A. Hossain, M. I. Abdullah, M. M. Rahman, B. Manavalan, N. Gu, M. S. Islam, and Z. Huang. Monkeynet: A robust deep convolutional neural network for monkeypox disease detection and classification. *Neural Networks*, 161:757–775, 2023.

E. Lindernoren. Repositorio con el entrenamiento de gan en pytorch, 2025. URL *https://github.com/eriklindernoren/PyTorch-GAN/blob/master/implementations/gan/gan.py*. Accedido el 9 de junio de 2025.

Medium. Mastering u-net: A step-by-step guide to segmentation from scratch with pytorch, 2024. URL *https://medium.com/@fernandopalominocobo/mastering-u-net-a-step-by-step-guide-to-segmentation-from-scratch-with-pytorch-6a17c5916114*.

M. Pal, A. Mahal, R. K. Mohapatra, A. J. Obaidullah, R. N. Sahoo, G. Pattnaik, S. Pattanaik, S. Mishra, M. Aljeldah, M. Alissa, et al. Deep and transfer learning approaches for automated early detection of monkeypox (mpox) alongside other similar skin lesions and their classification. *ACS omega*, 8(35):31747–31757, 2023.

A. Shateri, N. Nourani, M. Dorrigiv, and H. Nasiri. An explainable nature-inspired framework for monkeypox diagnosis: Xception features combined with ngboost and african vultures optimization algorithm. *arXiv preprint arXiv:2504.17540*, 2025.