Multi-class Skin Disease Classification Based on Deep Convolutional Neural Networks

Pedro Anderson Ferreira Castro

***Abstract*— Skin cancer is a life-threatening condition, with melanoma being a particularly lethal form contributing to the high death rates. Early detection is crucial, as it allows for timely treatment and can significantly reduce mortality. However, identifying skin cancer is difficult because various skin lesions often appear very similar. This work introduces a deep learning-based approach designed to accurately distinguish between different types of skin lesions and cancer using different deep convolutional neural network architectures pre-trained on the famous ImageNet dataset to address the multi-class classification problem. In total, three models are tested: the VGG-16, ResNet-50, and the DenseNet.**

***Keywords*— Skin Cancer, Deep Learning, Convolutional Neural Networks, Transfer Learning, Multi-class classification.**

I. Introduction

The human largest organ, the skin, serves as a crucial barrier between our bodies and the environment, comprising several distinct layers that play vital roles in protection and sensation. Among these layers, the epidermis stands out, encompassing Squamous, Basal, and melanocyte cells. Squamous cells form the outermost layer, while Basal cells provide foundational support, and melanocytes defend against harmful ultraviolet rays by producing melanin. However, despite its protective mechanisms, the epidermis is also vulnerable to genetic mutations triggered by UV exposure, which can lead to the development of various forms of skin cancer. Worldwide, almost 10 million skin cancer deaths took place in 2020. According to the World Health Organization (WHO), it is estimated that, globally, one-third of all diagnosed cancer cases are skin cancer. This means that skin cancer is a significant global public health concern, with approximately 5.4 million new cases diagnosed annually in the United States alone. Melanoma, responsible for three-quarters of all skin cancer-related deaths, claims about 10,000 lives each year in the United States. In Europe, there are over 100,000 cases annually, while Australia sees nearly 15,229 new cases of melanoma each year. Over recent decades, there has been a notable rise in skin cancer incidence. In the United Kingdom, melanoma rates have increased by 119% since the 1990s, and in the United States, by 250% over the same period. Detecting skin cancer early is crucial, however, the diagnosis typically involves the painful, time-consuming, and costly biopsy method, where a tissue sample is extracted for laboratory analysis. This procedure can be burdensome for patients, often requiring repeated hospital visits [1].

Pedro Anderson Ferreira Castro, Department of Electrical and Electronic Engineering (EEL), Federal University of Santa Catarina (UFSC), Florianópolis-SC, e-mail: pedro.a.f.castro77@gmail.com; The code of the project can be accessed at: https://www.kaggle.com/code/pedrocast77/final-project-dl-skin-cancer-classifier

Nowadays, the advances in the deep learning (DL) field have been able to provide models that are helping a lot in medical diagnosis tasks, automating a job that can accelerate the treatment of many patients in health institutions [2]. In this context, the fact that Convolutional Neural Networks (CNN) can learn relevant features by considering both spatial and local information from data (in this case, images), it is possible to use them as a feature extractor that can be passed to a classifier to predict labeled classes. Thus, using them for this application might produce satisfactory results in terms of precision/accuracy.

This project addresses a multiclass classification problem for skin lesion images, using Deep Convolutional Neural Networks (DCNN) using both techniques *Transfer Learning* and *Fine Tuning*, where models are pre-trained on a similar task and then adjusted to a new dataset, using the previously learned "knowledge" as a kick-off to fit the new problem. More specifically, 3 DCNNs models will try to predict between 7 classes of skin lesions and cancer (Melanocytic nevi, Melanoma, Benign keratosis, Basal cell carcinoma, Actinic keratosis, Vascular Lesions, and Dermatofibroma), present on the HAM10000 dataset [3]. The objectives include reproducing the results achieved by [1] for the VGG-16, ResNet-50, and the DenseNet models, implementing the code using PyTorch, and also exploring to improve the performance by hyperparameter tuning. Due to the imbalance in the number of samples from different classes, the application of data augmentation is mandatory to use the same metrics covered in the referred work, so this technique will also be applied. Later, a new metric is proposed (the balanced accuracy) to deal with the unbalance presented by the dataset, and then the models are optimized to achieve the highest value of this metric. Last but not least, the performance of the three architectures is observed under a distribution shift, i.e., when data of a different distribution (a new process of acquisition, different illumination, etc) is shown to a trained model, which usually drops its performance. To address that, selective classification is applied that is when a model is allowed to abstain from low-confidence predictions to avoid potential errors, as presented by the work of [4], for the out-of-distribution (OOD) dataset, that will be, in this case, the PAD-UFES-20 [5], containing 4 of the 7 original classes (Melanocytic nevi, Melanoma, Basal cell carcinoma and Actinic keratoses) present in the in-distribution (ID) data [3].

II. Related Works

Classification of medical images is a task well explored in the literature. Talking specifically about skin cancer classification using the HAM10000 dataset [3], several works

use its data in their research. The paper of [1] applies data preprocessing and augmentation techniques before utilizing a Convolutional Neural Network (CNN) and six transfer learning models (Resnet-50, VGG-16, DenseNet, Mobilenet, Inceptionv3, and Xception) on the dataset to classify the skin lesions. The performance metrics used include precision, recall, F1 score, and accuracy, with the models achieving accuracies of 77% to 90%. Additionally, five different stacking models were developed, but they performed poorly, with the highest accuracy being 78%. Also, the work of [6] explores the use of unsupervised domain adaptation (UDA) to integrate large external datasets for creating reliable classifiers. UDA, with multiple sources, can enhance the training set and bridge domain gaps between different skin lesion datasets, which vary due to distinct acquisition protocols. The study focused on three UDA training schemes: single-source, combined-source, and multi-source UDA, demonstrating their effectiveness in both binary and multi-class classification. It found a strong correlation between test error and label shift in multi-class tasks and showed that UDA can mitigate bias against minority groups and improve fairness in diagnostic systems, while maintaining superior classification performance, even without fairness-focused techniques. This success is potentially due to the increased and well-adapted demographic information from multiple sources. Furthermore, the work of [7] evaluated the current state of the art in the classification of dermoscopic images, based on the ISIC-2019 Challenge [8]. Various deep neural network architectures pre-trained on the ImageNet dataset were adapted to a combined training dataset, including publicly available dermoscopic and clinical images of skin lesions, using transfer learning and model fine-tuning. The performance and applicability of these models for detecting eight classes of skin lesions were examined. Real-time data augmentation techniques, such as random rotation, translation, shear, and zoom, were used to increase the number of training samples. Model predictions were adjusted by multiplying by inverse class frequencies and normalized to better approximate actual probability distributions. Overall prediction accuracy was further enhanced by averaging the predictions of several independently trained models. The best single model has been made available as a web service.

III. Methodology

In this section, the data used, the models' architectures and approaches used, such as the pre-processing techniques, and also the evaluation metrics are presented to achieve the objectives described earlier.

A. Datasets

This project uses 2 datasets for different purposes. The first dataset is the HAM10000, used to train and reproduce the results of [1], while the second one, the PAD-UFES-20, is only used as an OOD test set to verify the case of distribution shift.

1) HAM10000: The HAM10000 dataset [3] is a large collection of multi-source dermatoscopic images of common pigmented skin lesions. It comprises 10015 images which can serve as a training set for academic machine learning purposes. Cases include a representative collection of all important diagnostic categories in the realm of pigmented lesions (the target classes, as mentioned before): Actinic keratoses and intraepithelial carcinoma / Bowen's disease (**akiec**), basal cell carcinoma (**bcc**), benign keratosis-like lesions (solar lentigines / seborrheic keratoses and lichen-planus like keratoses, **bkl**), dermatofibroma (**df**), melanoma (**mel**), melanocytic nevi (**nv**) and vascular lesions (angiomas, angiokeratomas, pyogenic granulomas and hemorrhage, **vasc**). All dermoscopic images have a resolution of 600 x 450 pixels, and the channel is three. The images are taken by dermatoscopy instrument. The instrument is a type of magnifier that is used to take pictures of skin lesions. Those classes are then processed using 1-hot encoding to pass to the different DCNN models. Figure 1 shows some examples of each possible class. It is possible to also see data distribution over each class in the Figure 2.

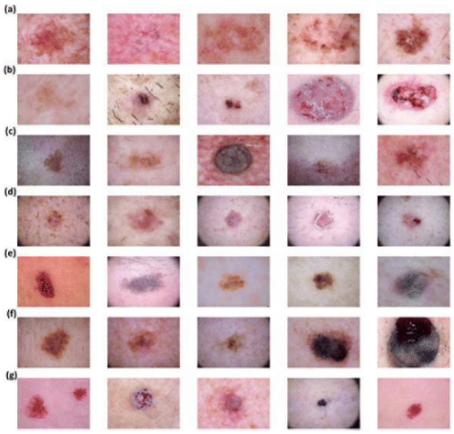


Fig. 1: Sample skin cancer images from HAM10000 dataset (a) Actinic keratosis (b) Basal cell carcinoma (c) Benign keratosis-like lesions (d) dermatofibroma (e) Melanocytic nevi (f) Melanoma (g) Vascular lesions. Source: M. S. Akter et al [1].

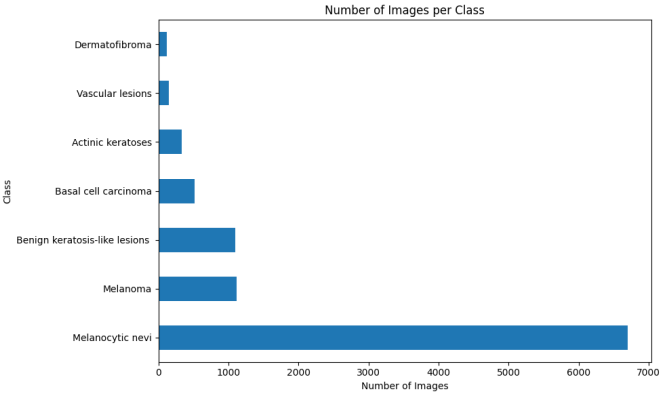


Fig. 2: Distribution of samples over each class present on the HAM10000 dataset. Source: author.

2) PAD-UFES-20: The PAD-UFES-20 dataset, collected through the Dermatological and Surgical Assistance Program

(in Portuguese: Programa de Assistência Dermatológica e Cirurgica - PAD) at the Federal University of Espírito Santo (UFES-Brazil), contains 2,298 samples of six different types of skin lesions [5]. Each sample includes a clinical image and up to 22 clinical features such as patient age, lesion location, Fitzpatrick skin type, and lesion diameter. The dataset encompasses Basal Cell Carcinoma (**BCC**), Squamous Cell Carcinoma (**SCC**), Actinic Keratosis (**ACK**), Seborrheic Keratosis (**SEK**), Bowen's disease (**BOD**), Melanoma (**MEL**), and Nevus (**NEV**), with BOD grouped under SCC, resulting in three skin cancers (BCC, MEL, SCC) and three skin diseases (ACK, NEV, SEK). Approximately 58% of the samples are biopsy-proven, particularly all BCC, SCC, and MEL cases, while others are clinically diagnosed by a consensus of dermatologists. The dataset comprises 1,373 patients, 1,641 lesions, and 2,298 images in *.png* format, captured using various smartphone devices. The associated metadata, available in a CSV document, includes up to 26 features per lesion, with each image/sample referencing the corresponding patient and lesion. Figures 3 and 4 show some examples of the data and also the samples' distribution over the matching classes of both datasets, which represents only a fraction of the whole data present on PAD-UFES-20.

Fig. 3: Random examples of skin lesion/cancer images from PAD-UFES-20 dataset with their respective labels. Source: author.

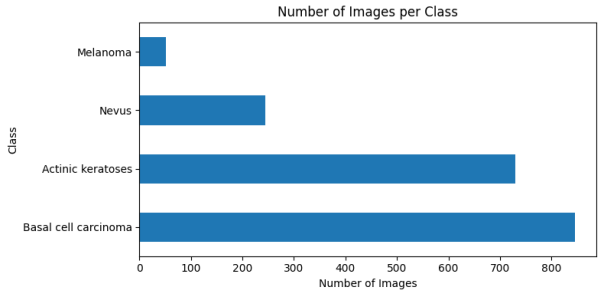


Fig. 4: Distribution of samples over the matching classes of both datasets, present on the PAD-UFES-20 dataset. Source: author.

B. Pre-Processing

Before passing the data to the convolutional models, a pre-processing step is taken. Firstly, for the ID dataset (HAM10000), the data are split into 3 sets: training, validation, and test, corresponding to 7130, 1783, and 1102 images, respectively. Additionally, a small portion of the training data is used to create a small training set (35%) for optimization of the hyperparameters with more time efficiency. The splits are performed using the stratification of the classes' distribution, to maintain its proportions across all sets. After that, the mean and the standard deviation of the samples in the training set are computed to be further used to normalize the data. Referring to the paper of [1], a bunch of transformations are done on the training set, such as random rotation (with 20º degrees), horizontal and vertical flips, random zoom (using random resized crop, with a scale from 0.7 to 1), random width height shift (using random affine, with parameters zero rotation and translation of 10% on each axis), and ZCA whitening (with epsilon equals to $1 * 10^{-5}$). Following the common transformations to all sets, the images are resized to 120x120, normalized, and converted to PyTorch tensors. For the OOD dataset, ordinal mapping is done by matching the classes' numbers for the common classes and assigning new values for the classes that aren't present on the ID data. The mismatched classes are then removed and the new test set is created using the same common transformations applied to the ID test set. Notice that the samples of the OOD data are not used to calculate the mean and the standard deviation to pass to the transformers, the objective here is to only verify how the models, without any training on the new data, will perform.

C. Models

Aiming to approximate the results obtained by [1], three models' architectures are chosen to be trained on the HAM10000 data: the VGG-16, the ResNet-50, and the DenseNet. All models are pre-trained on the famous ImageNet competition [9] and then adapted to the new task by changing the classification layers of the architectures. After that, they are trained and optimized (by choosing a new set of hyperparameters) on the data for a few epochs to learn how to distinguish the 7 classes that are present. Loading the weights of a previous task to another is called Transfer Learning and Fine-tuning is the process of optimizing the model to the new task, using the previous weights as a good starting point.

1) VGG-16: The VGG-16 model is a convolutional neural network (CNN) architecture known for its depth and simplicity. It was introduced by the Visual Geometry Group (VGG) from the University of Oxford [10]. It comprises 16 layers, including 13 convolutional layers with small 3x3 filters, followed by 3 fully connected layers. The model uses max-pooling to reduce spatial dimensions and ReLU activations for non-linearity. VGG-16 was designed to improve image classification tasks, achieving high accuracy on the ImageNet dataset, but it is computationally intensive due to its large number of parameters. The Figure 5 shows the original architecture. To modify it to the problem aborded in this project and also referring to some of its done by [1], the classifier part

is removed and replaced by a new one, consisting of a fully connected layer with 4096 neurons as input (following the output of last convolutional layer flattened) and 1024 as output (1/4 of the input), followed by a ReLU activation layer, passed to a Dropout layer with 0.4 rate, finally passed to another fully connected layer, with 1024 inputs and 7 outputs neurons (the number of classes).

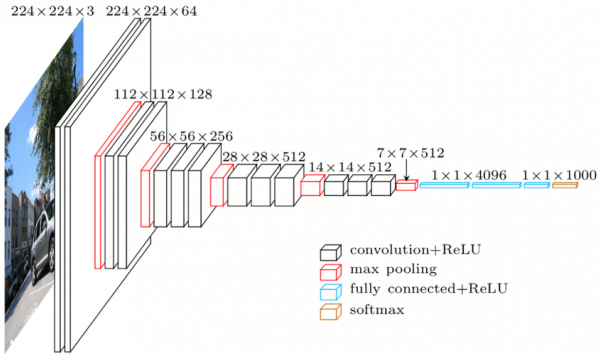


Fig. 5: VGG-16 architecture's schematic. Source: Manolis Loukadakis et al [11].

2) ResNet-50: The ResNet-50 model is a convolutional neural network (CNN) architecture known for its depth and the introduction of residual learning. It was introduced by Kaiming He and his colleagues from Microsoft Research [12]. It consists of 50 layers, including convolutional layers, identity blocks, and shortcut connections that mitigate the vanishing gradient problem. The model uses global average pooling to reduce spatial dimensions and ReLU activations for non-linearity. ResNet-50 was designed to improve image classification tasks, achieving high accuracy on the ImageNet dataset while maintaining computational efficiency compared to similarly deep networks. Figure 6 shows the original architecture. Modifying it to the problem approached in this project and also referring to some of its done by [1], the classifier part is removed and replaced by a new one, consisting of a fully connected layer with 2048 neurons as input (following the output of last convolutional layer flattened) and 1024 as output (1/2 of the input), followed by a ReLU activation layer, passed to a Dropout layer with 0.4 rate, finally passed to another fully connected layer, with 1024 inputs and 7 outputs neurons (the number of classes).

3) DenseNet: The DenseNet model is a convolutional neural network (CNN) architecture known for its dense connectivity and efficiency. It was introduced by Gao Huang and his colleagues from Cornell University [14]. DenseNet consists of multiple dense blocks, where each layer is connected to every other layer in a feed-forward fashion, promoting feature reuse and reducing the number of parameters. The model uses batch normalization and ReLU activations for non-linearity. DenseNet was designed to improve image classification tasks, achieving high accuracy on the ImageNet dataset while being computationally efficient. Figure 7 shows the original fundamental "dense block" of a DenseNet model, where the chosen variance of the model to this work was the DenseNet-161, selected by its similar performance on the ImageNet

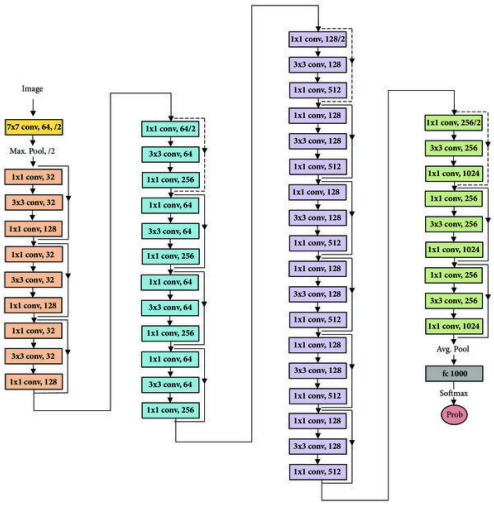


Fig. 6: ResNet-50 architecture's schematic. Source: Ali Qamar Bhatti et al [13].

dataset to the InceptionV3 model, the one which got the best results on the paper of [1]. Altering it to the problem presented in this project and also referring to some of its done by [1], the classifier part is removed and replaced by a new one, consisting of a fully connected layer with 2208 neurons as input (following the output of last convolutional layer flattened) and 1104 as output (1/2 of the input), followed by a ReLU activation layer, passed to a Dropout layer with 0.4 rate, finally passed to another fully connected layer, with 1024 inputs and 7 outputs neurons (the number of classes).

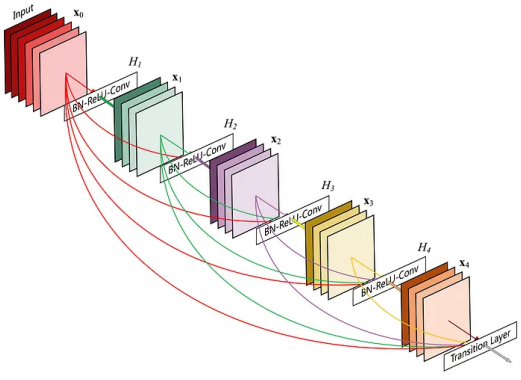


Fig. 7: A 5-layer dense block, the fundamental block of a DenseNet model. Source: Gao Huang et al [14].

All modifications regarding the number of neurons were made by the author's choice since those values are not given by [1]. Also, in dealing with a multi-class problem, the models used as their criterion the *torch.nn.CrossEntropyLoss*, which computes the cross entropy loss between input logits and target. To optimize the models the Adam algorithm was used, with default parameters.

D. Evaluation Metrics

When proposing comparisons over different model's architectures, it is important to establish metrics to check the performance of each model on the proposed task. For this work, 4 metrics were adopted: precision, recall, accuracy, and F1-Score. Nonetheless, those metrics are adopted to compare the models' performance in the study of [1]. To follow what has been done in the literature for this specific task, the normalized (or balanced) multi-class accuracy and the average AUC across all diagnoses, according to the ISIC-2019 Challenge [8] where the HAM10000 was also included, are also applied.

1) Precision: It is important to specify how many of the predicted positive values are correct. Precision measures this aspect and is particularly useful when the number of False Positives is high. Precision is defined as:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1}$$

where TP is the number of True Positives and FP is the number of False Positives.

2) Recall: Also known as Sensitivity or True Positive Rate, is the ratio of correctly predicted positive observations to all observations in actual class. It answers the question "Of all the items that are positive, how many did we correctly identify as positive?". The Recall is defined as:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

where FN is the number of False Negatives.

3) Accuracy and Balanced Accuracy: Accuracy is a metric that measures the overall correctness of a model's predictions. It is defined as the ratio of correctly predicted observations (both true positives and true negatives) to the total number of observations. Accuracy answers the question "Out of all the samples, how many were correctly predicted?". Accuracy is particularly useful when the classes are balanced. It's calculated as follows:

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

where TN is the number of True Negatives.

Balanced (or normalized) multi-class accuracy is a metric used to evaluate the performance of a classifier in a multi-class classification problem. It considers the accuracy of each class, weighting them equally regardless of their prevalence in the dataset. Specifically, it calculates the mean of the true positive rate (sensitivity) and the true negative rate (specificity). This metric ensures that the performance of the classifier is evaluated in a balanced manner across all classes, rather than being dominated by the most prevalent classes. The normalized (or balanced) multi-class accuracy can be defined as:

$$\text{Balanced Accuracy} = \frac{1}{2}\left(\frac{\text{TP}}{\text{TP} + \text{FN}} + \frac{\text{TN}}{\text{TN} + \text{FP}}\right) \tag{4}$$

where TP, TN, FP, and FN stand for true positives, true negatives, false positives, and false negatives, respectively.

4) F1-Score: It is the weighted average of Precision and Recall. Therefore, it takes both false positives and false negatives into account. F1-Score is more useful than accuracy, especially if you have an uneven class distribution. The equation for the F1-Score is:

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

5) Average AUC: The average AUC (Area Under the Curve) across all diagnoses is a metric used to evaluate the performance of a multi-class classifier by averaging the AUC scores for each individual class. The AUC score for a class represents the area under the Receiver Operating Characteristic (ROC) curve, which is a plot of the true positive rate against the false positive rate at various threshold settings as exemplified in the Figure 8. By averaging these AUC scores across all classes, we obtain a single measure that reflects the classifier's ability to discriminate between the classes on average. So, we can calculate the average AUC as:

$$\text{Average AUC} = \frac{1}{N}\sum_{i=1}^{N} \text{AUC}_i \tag{6}$$

where N is the number of classes and AUC_i be the AUC for class i.

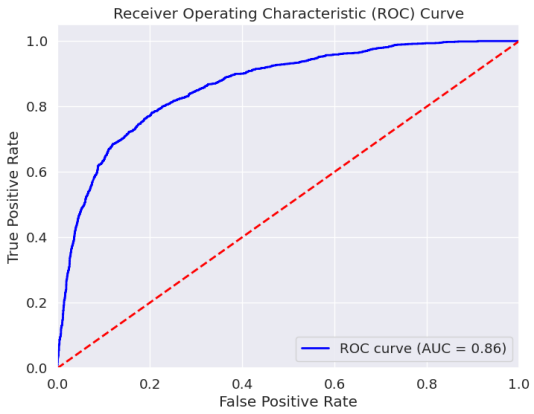


Fig. 8: ROC curve example for a binary classification problem. Source: Pedro A. F. Castro [15].

IV. Experiments and Results

This section covers all the experiments made in this work. First, attempting to reproduce the results obtained by [1], tests are conducted to optimize the hyperparameters of the Dropout layer and the learning rate by using a scheduler. After this, a test of regularization is done and finally, with the optimal values, the final results regarding this approach are obtained. Subsequently, we further tune the DCNNs based on the balanced accuracy, a metric that makes more sense to deal with unbalanced data, performing exploratory tests for removing/adding what makes more sense to this specific task. Finally, with the optimized models, we approach a case of distribution shift, giving new test data (PAD-UFES-20) and analyzing the performance over those conditions, applying

selective classification to experiment if it's feasible to apply the models without re-training it on the new distribution.

A. Replicated Results

Since none of the hyperparameter values, except the learning rate of each model, were given in the work of [1], those optimal values need to be found by experiments. Following this idea, several tests were conducted to optimize those hyperparameters' values. Augmentation parameters were set as default or by choosing between 1-3 values and seeing how the validation accuracy would grow, but without a refined grid search. The models were trained for 30 epochs, according to the referred paper.

1) Dropout Optimization: To optimize the dropout probability p, we fixed a model (in this case, the DenseNet) and tested the following values on the small training dataset: 0.3,0.4,0.5, and 0.6. Since the goal metric in the work of [1] is accuracy, 0.4 gave the most stable curve and highest value of accuracy on the validation set. Figures 9 and 10 illustrate the accuracy curves of each run with a different p value for the training (small) and validation sets. It is important to notice that using the learning rate given by the authors of [1], the performance dropped, so it was changed to use the same as the others.

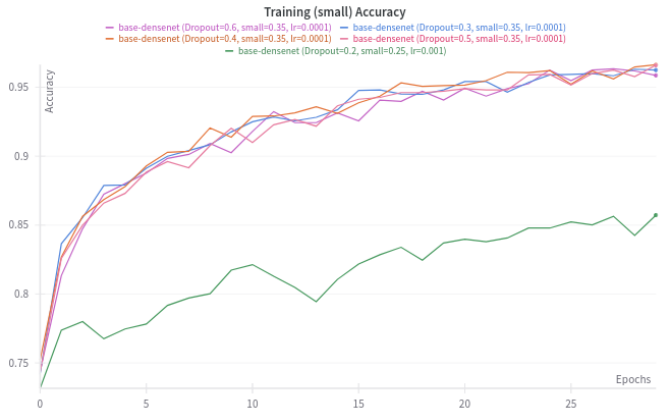


Fig. 9: Accuracy plot for the training set (small) over the epochs. Each line uses a different value of p, the dropout probability. Source: author.

2) Scheduler Optimization: Aiming to improve the performance even more, a common approach is to use a scheduler, where the learning rate is updated given certain conditions. In this case, the scheduler applied was the *ReduceLROnPlateau*, which reduces the learning rate when a metric has stopped improving for a determined number of epochs. This is called patience. As the goal metric of the research done by [1] is accuracy, the mode was set to 'max', meaning that it will monitor if it still increasing or don't. Figures 11 and 12 show the optimization of the patience value, while Figure 13 presents the learning rate update over the epochs. The patience of 10 was chosen since it gave the highest value on the validation set. Both tests used a factor of 0.1.

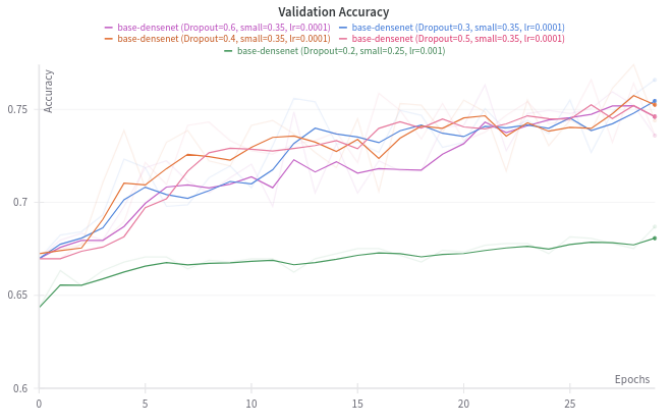


Fig. 10: Accuracy plot for the validation set over the epochs. Each line uses a different value of p, the dropout probability. Source: author.

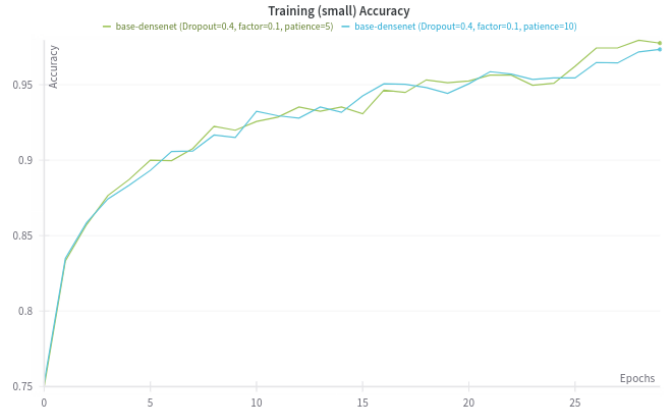


Fig. 11: Accuracy plot for the train (small) set over the epochs. Each line uses a different number of epochs as patience. Source: author.

3) Regularization: Regularization is often used when the model is overfitting on the training set. There are tons of ways to do it, even using the hyperparameters of the transformations when performing data augmentation. This was only a test to see if any improvement would be perceived on the validation set using L2 regularization, also known as weight decay on the Adam optimizer. Figures 14 and 15 display that the lower the value of the hyperparameter, the better the result, so the decision was to use the value of $1 * 10^{-5}$. Additionally, the scheduler was turned off to see only the influence of the weight decay.

4) Test set Results: After all the tests presented above, noticing that the model appears to still learning, the number of epochs was changed to 50, 20 more than used by [1]. Using the best values for the hyperparameters tuned, the different models are now trained and tested. Figures 16 and 17 show both, the accuracy of training (full set) and validation data, and the loss for those sets, of each model. With the models fully trained, we check their performances now on the test set, observing also the other metrics presented before. Also,

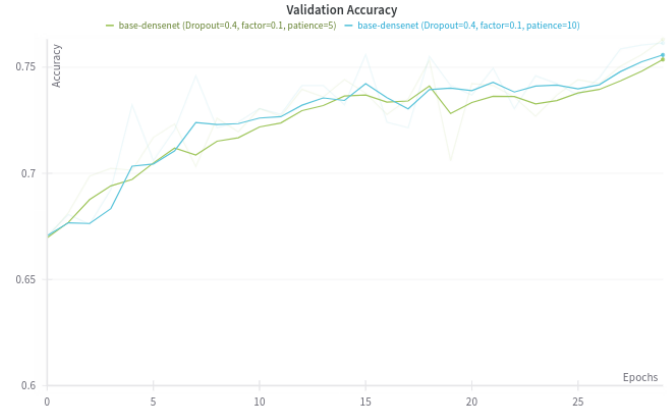


Fig. 12: Accuracy plot for the validation set over the epochs. Each line uses a different number of epochs as patience. Source: author.

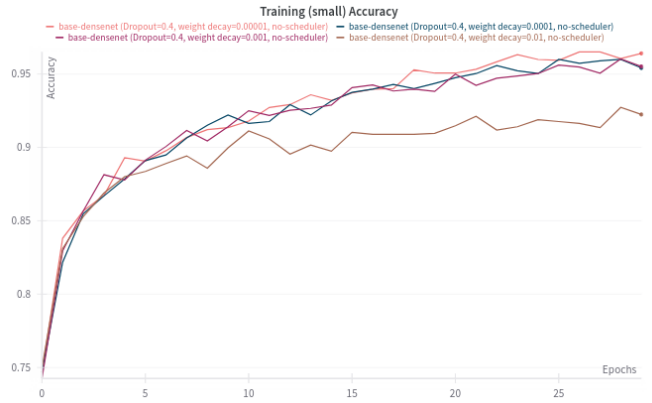


Fig. 14: Accuracy plot for the train (small) set over the epochs. Each line uses a different weight decay value. Source: author.

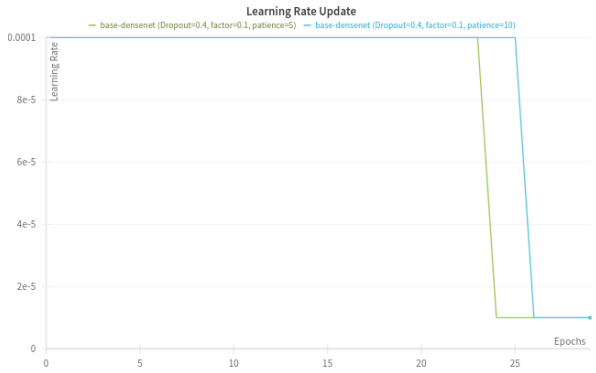


Fig. 13: Learning rate value update plot over the epochs. Each line uses a different number of epochs as patience. Source: author.

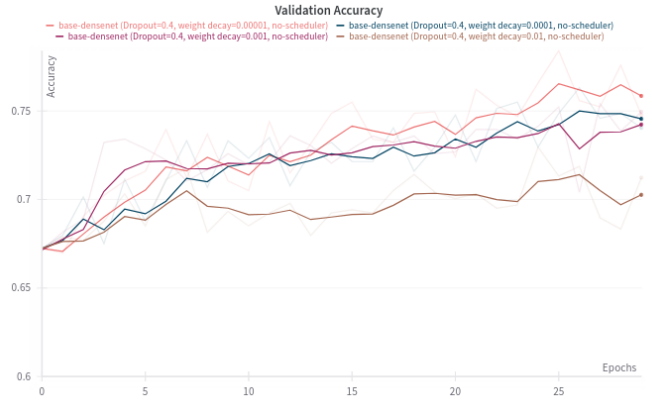


Fig. 15: Accuracy plot for the validation set over the epochs. Each line uses a different weight decay value. Source: author.

the confusion matrix of each of the models is plotted, to evaluate the performance over each class. The results obtained are shown in Table I while the confusion matrices are shown in Figure 18. The highlighted results in Table I indicate that the DenseNet model achieved the best performance, with accuracy about 4% higher than the ResNet-50 and 0.02% than the VGG-16, which got the second place in this project. Even training for more epochs, the models still don't match the metrics values obtained by [1]. Nevertheless, although the results are not the same, the goal metric isn't the best choice for the case of unbalanced data, let alone for medical applications. Figure 18 shows that the Melanoma class was often misclassified by all three models, which can be dangerous when dealing with diagnosis. Therefore, a better approach to this problem needs to be made, since the goal metric is inflated, and the models' performances are not reliable for a real case.

B. Improving the Models

The approach of the previous section wasn't ideal for medical applications or even for dealing with unbalanced data, we now propose the optimization of the models, inspired by the ISIC challenge, based on the balanced accuracy as the goal metric, and the macro AUC for tied performances[8]. Now all the hyperparameters set before need to be re-optimized and an analysis of what may be useful to increase the goal metric is done, removing or adding what was previously used, regarding their effects on the performances. The models are now trained for 80 epochs, the scheduler patience is applied to all tests.

1) Augmentation Optimization: When doing augmentation to handle the problem of unbalanced data or just to prevent overfitting, it is important to investigate what transformations might be useful, since applying it randomly can just make the

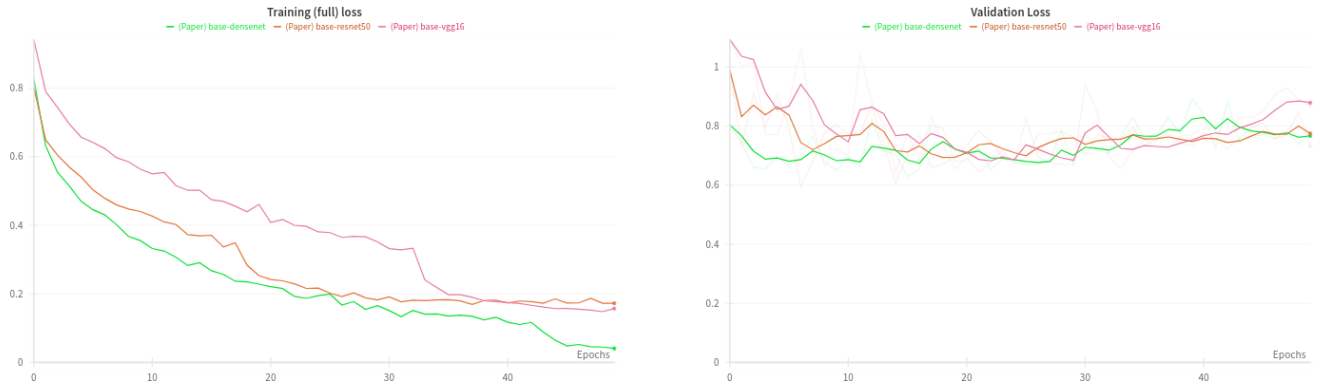
Metrics	Models		
	VGG-16	**ResNet-50**	**DenseNet**
Balanced Accuracy	**0.6980**	**0.5849**	**0.6905**
Accuracy	**0.7958**	**0.7722**	**0.8167**
Avg-AUC	0.8250	0.7626	**0.8234**
Weighted Precision	0.8046	0.7628	0.8189
Weighted Recall	0.7958	0.7722	0.8167
Weighted F1	0.7992	0.7644	0.8173

TABLE I: Results of the models on the test set, imitating the conditions of the paper [1]. Source: author.

(a) Training accuracy of models.

(b) Validation accuracy of models.

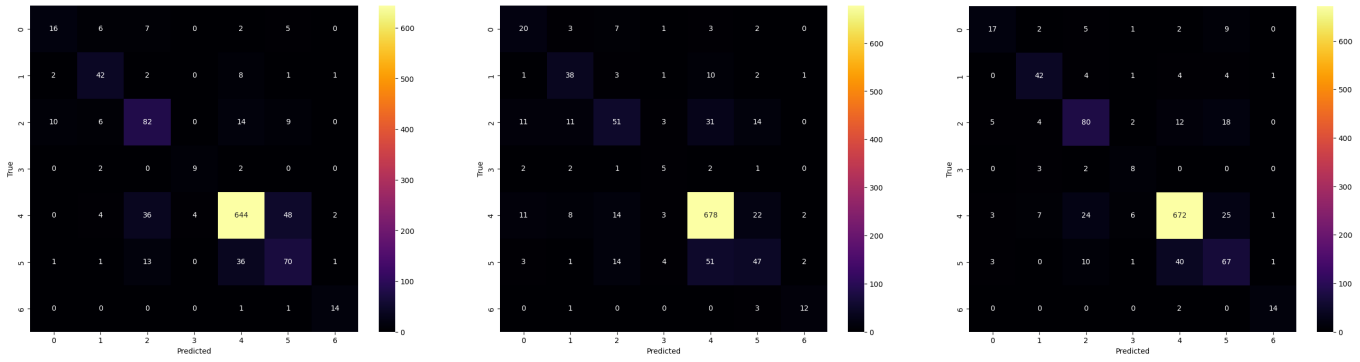
Fig. 16: Accuracy plots for the training (full) and validation sets over the epochs, for all 3 models chosen. Source: author.



(a) Training loss of models.

(b) Validation loss of models.

Fig. 17: Loss plots for the training (full) and validation sets over the epochs, for all 3 models chosen. Source: author.



(a) VGG-16 confusion matrix.

(b) ResNet-50 confusion matrix.

(c) DenseNet confusion matrix.

Fig. 18: Confusion matrix plots for each model, with the settings mimicking the work of [1], on the test set. The class numbers are mapped as: 0 - Actinic keratoses, 1 - Basal cell carcinoma, 2 - Benign keratosis-like lesions, 3 - Dermatofibroma, 4 - Melanocytic nevi, 5 - Melanoma, and 6 - Vascular lesions. Source: author.

data too difficult for the model. Thus, variations of the used transformations are applied and tested to see how the models would behave. We fixed the VGG-16 model due to its observed speed for training in comparison to the other 2 DCNNs and tested it on 5 profiles with different transformations. Figure 19 illustrates the performance for each profile, while Table II has the transformations used for each profile. Profile 5, containing the ZCA whitening appears to be the worst set of transformations, which might be one of the reasons the models performed poorly. Given the results obtained, the best profile was the first one, due to its fast convergence and stability. From now on, this one will be applied to the training data.

2) Second Dropout Optimization: Same as before, the dropout layers present in the models need to be optimized,

Augmentation Profile	Transformations
1	Rotation, Flip, Resize, Normalization
2	Rotation, Flip, Resize, Normalization, Zoom
3	Rotation, Flip, Resize, Normalization, Width and Height Shift
4	Rotation, Flip, Resize, Normalization, ZCA Whitening, Width and Height Shift
5	Rotation, Flip, Resize, Normalization, Width and Height Shift, Color Jitter

TABLE II: Description of the profiles tested using different transformations for data augmentation. The "Flip" transformation uses both horizontal and vertical flips. Source: author.

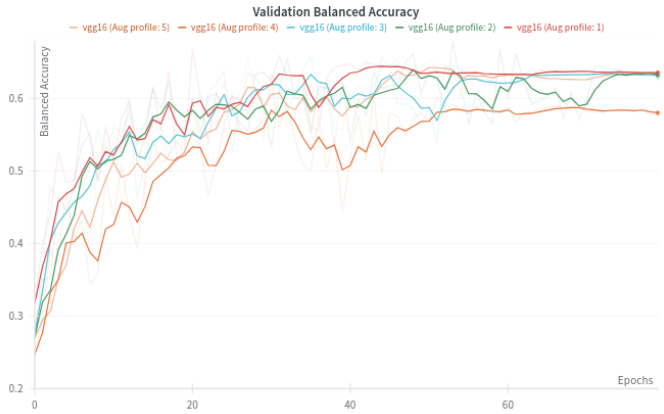


Fig. 19: Balanced accuracy plot for the validation set over the epochs. Each line uses a different augmentation profile, explained in Table II. Source: author.

with the best p value. More tests are performed and, as shown by Figure 20, the best choice was to put it to 0.2, since it achieved the highest value for the balanced accuracy on the validation set.

3) Optimized Models and Final Results: After tuning the hyperparameters. the models are now trained using the full training data. Figure 21 exhibits the balanced accuracy over the training and validation sets, while Figure 22 shows their respective losses. Finally, the test set is now passed to the models, enabling the comparison of what was obtained before. Starting by looking at Table III, it is noticeable that the DenseNet surpasses the other two models, clearing being the best one when talking about performance. But this is not the only thing: comparing with the results shown in Table I, the general performance has increased, even surpassing the results reported by [1], when analyzing the accuracy again. This highlights the importance of choosing proper suitable metrics, considering the particularities of the approached problem. Furthermore, the confusion matrices of each model, illustrated in Figure 23, also reinforce it, by that fewer predictions of the Melanoma class are misclassified.

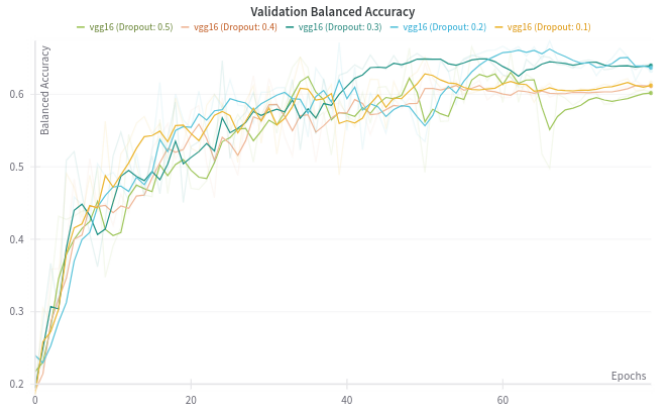


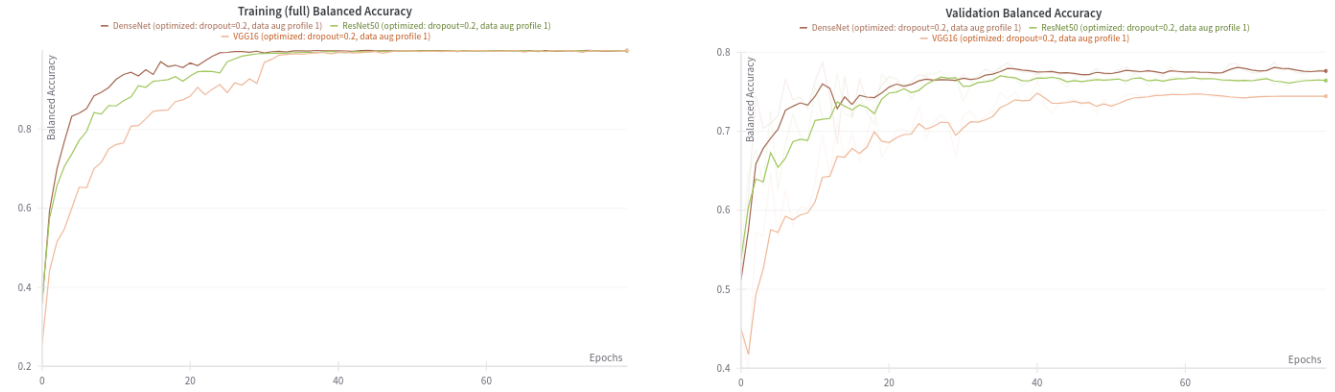
Fig. 20: Balanced accuracy plot for the validation set over the epochs. Each line uses a different dropout p probability. Source: author.

Metrics	Models		
	VGG-16	ResNet-50	DenseNet
Balanced Accuracy	**0.7392**	**0.7505**	**0.7982**
Accuracy	**0.8412**	**0.8557**	**0.8802**
Avg-AUC	0.8492	0.8570	**0.8825**
Weighted Precision	0.8389	0.8541	0.8768
Weighted Recall	0.8412	0.8557	0.8802
Weighted F1	0.8377	0.8539	0.8775

TABLE III: Results of the models on the test set, now optimized using the balanced accuracy as goal metric. Source: author.

C. Data Shift and Selective Classification

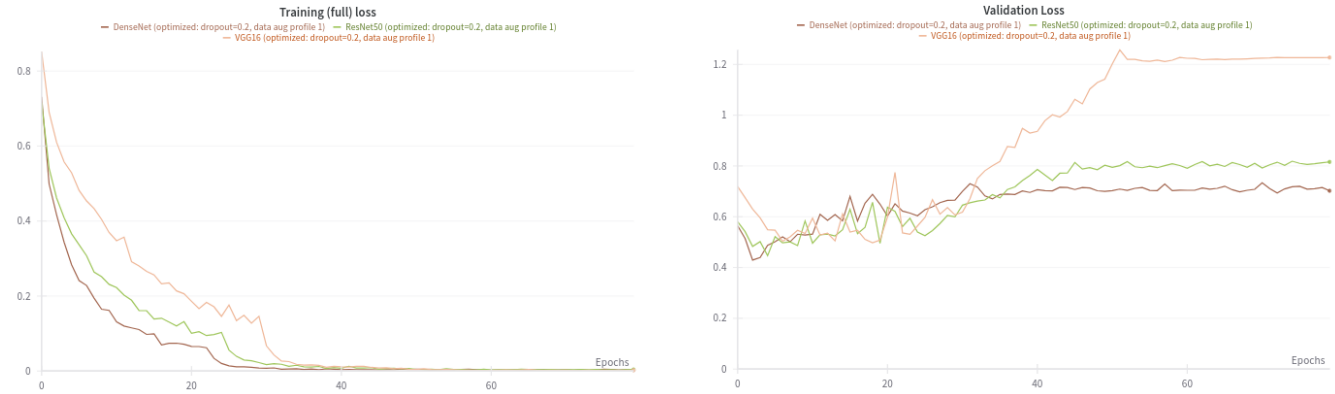
Having the models optimized, we now study the case of data shifting, by giving new data from another distribution, the PAD-UFES-20 [5], which uses smartphone images instead of a dermatoscopy. The performance for all models dropped significantly, with the VGG-16, the ResNet-50, and the DenseNet achieving accuracies of 0.2779, 0.3111, and 0.3025, respectively, and also 0.3254, 0.3583, and 0.3470 balanced accuracy scores. Those scores are very low, so, trying to solve it, we apply selective classification to reduce the coverage of the dataset to increase its performance by picking only samples with confidence, in this case, given by the Maximum Softmax Probability (MSP), above a certain threshold. Similar to what is done on the Receiver Operating Characteristic Curve (ROC curve), it is important to set an operating point for each model, which is done by plotting the Risk Coverage curve, where the y-axis is the risk associated with a given coverage of the data. The risk, in this case, is calculated by both, using $1-accuracy$ and $1-balanced\ accuracy$, the thresholds are chosen based on the needs of whoever uses the model in its final application, for instance, by choosing the amount of risk accepted to the specific task. The plots of those 2 curves can be seen in Figure 24, where it is possible to notice that, for both graphs, the models are allowed to cover about 20% of the data before giving a 50% of risk using the accuracy and even less when using the balanced accuracy, which is far from ideal.



(a) Training balanced accuracy of models.

(b) Validation balanced accuracy of models.

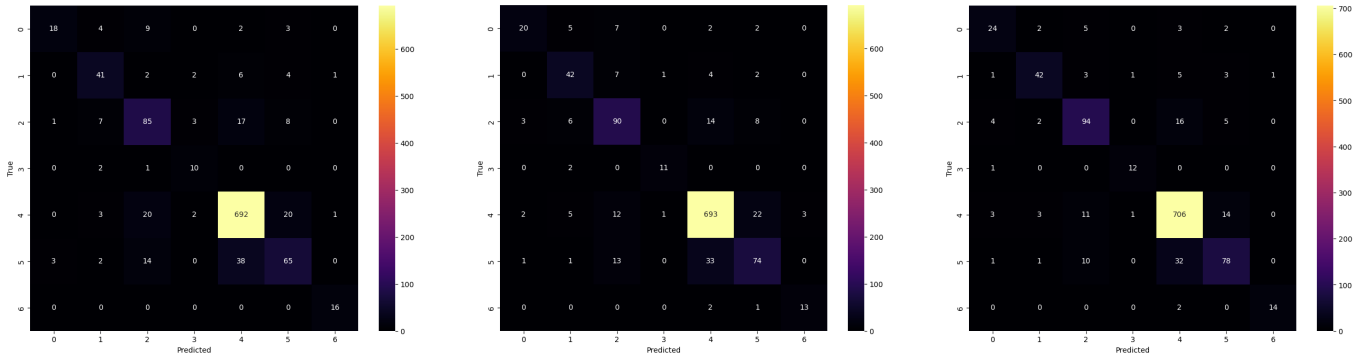
Fig. 21: Balanced accuracy plots for the training (full) and validation sets over the epochs, for all 3 models chosen. Source: author.



(a) Training loss of optimized models.

(b) Validation loss of optimized models.

Fig. 22: Loss plots for the training (full) and validation sets over the epochs, for all 3 optimized models. Source: author.



(a) VGG-16 optimized confusion matrix.

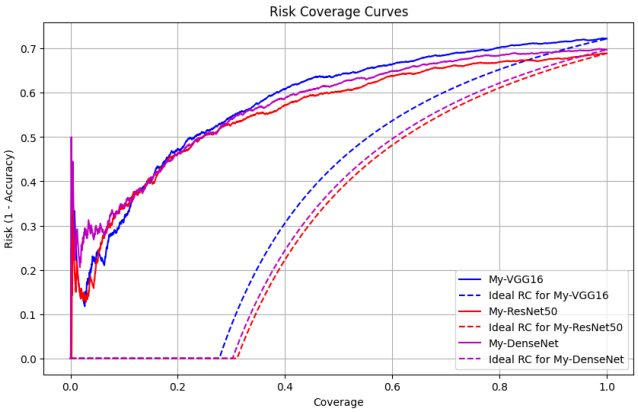
(b) ResNet-50 optimized confusion matrix.

(c) DenseNet optimized confusion matrix.

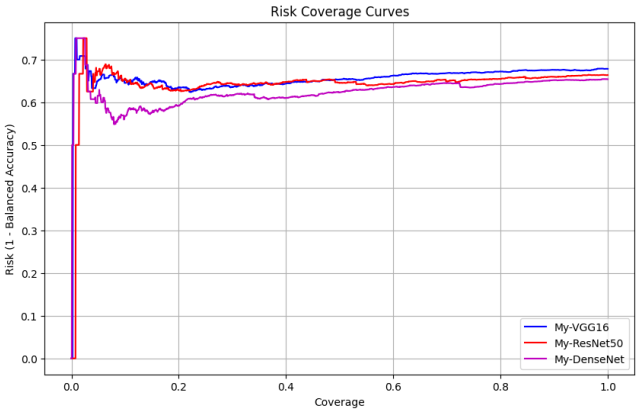
Fig. 23: Confusion matrix plots for each model, with the settings changed to optimize the balanced accuracy, on the test set. The class numbers are mapped as: 0 - Actinic keratoses, 1 - Basal cell carcinoma, 2 - Benign keratosis-like lesions, 3 - Dermatofibroma, 4 - Melanocytic nevi, 5 - Melanoma, and 6 - Vascular lesions. Source: author.

V. Conclusion

This project addresses a multi-class classification problem, where three distinct architectures were trained on the HAM10000 [3] dataset to predict 7 classes of skin lesions, with 3 of them being skin cancer cases. The first part of the work tries to reproduce the results obtained by [1], checking that the approach was not optimal from a practical perspective. Later on, the goal metric was changed from accuracy to balanced accuracy, giving better and more realistic results, after testing what is best suitable for this task. All three models got satisfactory performance with a balanced accuracy score above 72% and superior performance when compared to what is reported by the referred paper if the accuracy score is considered. Last but not least, a study case was



(a) RC curve of optimized models, with risk being given by $1 - Accuracy$.



(b) RC curve of optimized models, with risk being given by $1 - Balanced\ Accuracy$.

Fig. 24: Risk coverage curve plots for the optimized models. Figure 24b doesn't have the ideal curve due to not being a trivial problem determining how to weigh each coverage's losses since the prevalence changes every time a new sample is included. Source: author.

conducted to analyze the behavior of the optimized models under distribution shift, using the PAD-UFES-20 [5] as a new test set, where it was noticed that the values for the goal metric dropped significantly. Selective classification was applied to the models, using the MSP as their confidence estimator, but the RC graphs show that they're able to make predictions just for a small fraction of the whole data. This behavior can probably be explained by the difference between the domains being substantially high since the performance of the models dropped too much, which means that their confidence is very low about the new set, or even because of the normalization step because the mean and standard deviation used are taken from the training data of the ID dataset, which could be non-optimal values for normalizing the OOD data.

VI. Conflict of Interest

The author declares that this final project is not part of another subject, internship, or undergraduate or postgraduate completion work, having been developed exclusively for the Machine Learning course (EEL410250-41000056DO/ME - 20241), under single authorship.

References

[1] M. S. Akter, H. Shahriar, S. Sneha, and A. Cuzzocrea, "Multi-class skin cancer classification architecture based on deep convolutional neural network," in *2022 IEEE International Conference on Big Data (Big Data)*, 2022, pp. 5404–5413.

[2] M. Bakator and D. Radosav, "Deep learning and medical diagnosis: A review of literature," *Multimodal Technologies and Interaction*, vol. 2, no. 3, 2018. [Online]. Available: https://www.mdpi.com/2414-4088/2/3/47

[3] P. Tschandl, "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," 2018. [Online]. Available: https://doi.org/10.7910/DVN/DBW86T

[4] L. F. P. Cattelan and D. Silva, "How to fix a broken confidence estimator: Evaluating post-hoc methods for selective classification with deep neural networks," 2024.

[5] A. G. Pacheco, G. R. Lima, A. S. Salomão, B. Krohling, I. P. Biral, G. G. de Angelo, F. C. Alves Jr, J. G. Esgario, A. C. Simora, P. B. Castro, F. B. Rodrigues, P. H. Frasson, R. A. Krohling, H. Knidel, M. C. Santos, R. B. do Espírito Santo, T. L. Macedo, T. R. Canuto, and L. F. de Barros, "Pad-ufes-20: A skin lesion dataset composed of patient data and clinical images collected from smartphones," *Data in Brief*, vol. 32, p. 106221, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S235234092031115X

[6] J. Wang, Y. Zhang, Z. Ding, and J. Hamm, "Achieving reliable and fair skin lesion diagnosis via unsupervised domain adaptation," 2024.

[7] J. Steppan and S. Hanke, "Analysis of skin lesion images with deep learning," 2021.

[8] ISIC Challenge. [Online]. Available: https://challenge.isic-archive.com/landing/2019/

[9] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1409.1556

[11] M. Loukadakis, J. Cano, and M. O'Boyle, "Accelerating deep neural networks on low power heterogeneous architectures," 01 2018.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[13] A. Bhatti, M. Umer, S. Adil, M. Ebrahim, D. Nawaz, and F. Ahmed, "Explicit content detection system: An approach towards a safe and ethical environment," *Applied Computational Intelligence and Soft Computing*, vol. 2018, 07 2018.

[14] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2018. [Online]. Available: https://arxiv.org/abs/1608.06993

[15] P. A. F. Castro, "Ml binary classification with sklearn," https://www.kaggle.com/code/pedrocast77/ml-binary-classification-with-sklearn, 2024, accessed: 2024-06-07. [Online]. Available: https://www.kaggle.com/code/pedrocast77/ml-binary-classification-with-sklearn