Deep Hierarchical Diagnosis of Skin Lesions

Rúben Gomes, Jorge Marques, and Catarina Barata
Instituto Superior Técnico, Lisboa, Portugal

July 2020

Abstract

The early detection of skin cancer is very important because it increases significantly the patient's prognosis, and techniques of data classification are important tools to accomplish it.

Skin lesions can be organized in a hierarchical structure, where in the first level the melanocytic and non-melanocytic lesions are splitted, and in each of these sub-groups the benign and malignant lesions are further discriminated. However, this knowledge has been disregarded by most automatic methods. Thus, in this thesis we propose to investigate this issue.

In this work, we implement and compare the results of two different approaches to diagnose skin lesions' images, based in deep learning: (i) a flat approach, where the inference is done in a single decision which involves all the categories, and (ii) a hierarchical approach that explores the hierarchical structure of skin lesions' organization, where the inference for each example involves more than one decision. Afterwards, a mixed model, which combines these approaches, was implemented in order to capture the strengths of each one.

We verified that the hierarchical model was affected by error propagation of intermediate decisions, in comparison with the flat classifier. The mixed model allowed to improve significantly the results of both approaches.

Keywords: Deep Learning, CNNs, Skin Lesions

1. Introduction

According to the World Health Organization, the incidence of both non-melanoma and melanoma skin cancers has been increasing over the past decades. The diagnosis of skin lesions is done by dermatologists that have been properly trained [3], using dermoscopy which is a non-invasive in vivo method for the microscopic examination of pigmented skin lesions [14]. However, the diagnosis based on a visual inspection of the lesion may be difficult even for experienced physicians, because different categories of skin lesions may look very similar.

For the last decades, a great effort as been made by the research community in the development of computer-aided tools that can be used by dermatologists to help the diagnosis [4]. More recently, large databases, comprising several categories of skin lesions, have been made available to the community. The increase of available data have allowed the use of deep neural networks (DNN), bringing significant improvements [6].

Skin lesions are organized hierarchically, where a distinction is made first between melanocytic and non-melanocytic lesions, and then between benign and malignant. At the last level of this hierarchy comes the final category, as shown in figure 1. Dermatologists divide the lesions diagnosis task into a hierarchical method: first they diagnose it as melanocytic or non-melanocytic, and only then they perform the final diagnosis [3].

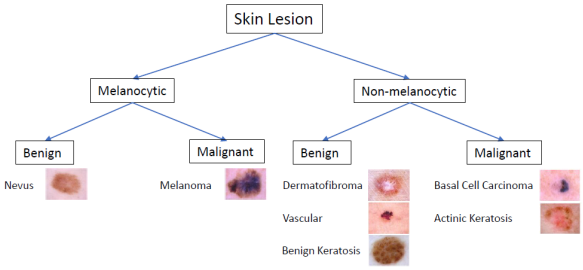

Figure 1: Hierarchical organization of skin lesions with examples. Dermoscopy images taken from [2].

The fact that dermatologists diagnose a skin lesion with a hierarchical method have inspired us to investigate whether there is any benefit to use hierarchical neural networks for lesion classification. Despite the relevance of the hierarchical organization of skin lesions, few works have used this information into their systems. For instance, a previous work [5] showed that structured classification based on a distinction between malignant and benign lesions, followed by the diagnosis of the latter in different classes, leads to better results in ISIC 2017 [8]. However, these results remain to be validated on a larger dataset, comprising more classes

of non-melanocytic lesions.

The main goal of this thesis is to compare the performance of flat classifiers, that try to classify the lesions directly into one of the many possible categories, and hierarchical classifiers, where the classification involves more than one decision, regarding skin lesions classification. Also, it is very important to improve the efficiency with respect to the available data, and this work also aims to study the effects of reducing the amount of data on the models' performance.

This document is organized as follows: section 2 provides a summarized explanation of convolutional neural networks (CNN) and some techniques that have been adopted to improve the models' performance. Section 3 describes the implemented models and the techniques used to diagnose skin lesions' images. Section 4 presents the dataset and performance metrics used in this work. Then, it presents the experimental results and discussion. Finally, in section 5, conclusions and future work topics are presented.

2. Deep Learning Background

Deep learning is a part of a broader family of machine learning algorithms. A deep learning architecture consists in a stack of simple modules, which are used to progressively extract higher level features from the raw input.

The importance of this technique is that the features do not need to be designed by humans with considerable domain expertise, because they are learned from data using a task-driven learning procedure [16]. This technique takes advantage of increases in the amount of available computation and data and it is becoming state-of-the-art in many computer vision tasks.

Due to the importance of CNNs in this work, which is a deep learning based algorithm, we provide a detailed explanation about CNNs.

CNNs are a specialized kind of neural network for processing data that has a known grid-like topology, like images [10]. The most common used layers in CNNs are: convolutional, pooling, and fully-connected layers (FCLs).

2.1. Convolutional Layer

A convolutional layer, receives an image I in a 3D-array format, with a width $W(I)$, height $H(I)$, and depth $D(I)$. It performs convolutions to the image I with multiple kernels. A kernel K has width $W(K)$, height $H(K)$, and depth $D(K) = D(I)$. Mathematically, the convolution operation is formally defined by the following expression [10]:

$$S(i,j) = (I * K)(i,j) = \\ = \sum_m \sum_n I(m,n)K(i-m, j-n). \quad (1)$$

In this expression, the depth variable is omitted, since the filters slide spatially across the image, and not across its channels.

The outputs of each convolution are concatenated, forming a new image I'. Then, a non-linear activation function is applied to every pixel of I', resulting in the output of the convolutional layer, often called feature map.

There are several non-linear activation functions: tanh, sigmoid, ReLU, Leaky-ReLU, etc. In this work we adopted ReLU as the activation for every hidden layer.

2.2. Pooling Layer

A pooling layer receives as its input an image I with a width $W(I)$, height $H(I)$ and depth $D(I)$, and it is characterized by a window with width $W(Win)$ and height $H(Win)$. This window will slide across the image, with a specific stride, and for each $H(Win) \times W(Win) \times D(I)$ block of pixels selected, a pooling function is applied. Each pooling function replaces the input image by a summary statistic of the nearby inputs [10]. For instance, the maximum pooling operation reports the maximum input within a rectangular neighborhood, defined by the window.

This operation transforms the input image, I, into another image, I', with reduced spatial size, improving the computational efficiency of the network because the next layer processes an input with smaller dimension. It also helps to make the representation approximately invariant to small translations of the input [10].

2.3. Fully Connected Layer

The last layers of a CNN usually comprise FCLs. Before the first FCL of a CNN, there is an operation that flattens the output of the previous layer (converts the 3D-array into a single 1D-array).

Each FCL connects each neuron to every neuron in the previous layer. It converts one input 1D-array X into one output 1D-array Y. Each element (neuron) in Y is obtained by a linear combination of all elements in X. Finally, a non-linear activation function is applied to all elements of Y, resulting in the layer output.

The last layer of a neural network for classification is a FCL with a softmax as non-linear activation function. The number of neurons in this layer is the number of different classes for the given problem. The softmax function $\sigma : \mathbb{R}^C \rightarrow \mathbb{R}^C$ is defined by the formula:

$$\sigma(\vec{x})_i = \frac{e^{x_i}}{\sum_{j=1}^{C} e^{x_j}}, \quad (2)$$

for $i = 1, ..., C$ and $\vec{x} = (x_1, ..., x_C) \in \mathbb{R}^C$, where C is the number of classes. Each $\sigma(\vec{x})_i$ represents the probability that the input image belongs to the i-th class computed by the CNN. Thus, we have that

$$\sum_{i=1}^{C} \sigma(\vec{x})_i = 1. \quad (3)$$

2.4. Training of Neural Networks

A neural network is composed by parameters (biases and weights) and is able to learn automatically these parameters during the training procedure, in order to optimize a loss function. The loss function, which we aim to minimize, typically measures the error between the output of the network and the ground truth data. This means that training a neural network is the same as solving an optimization problem of the model's parameters:

$$\hat{w} = \arg\min_{w} \ f(w), \quad (4)$$

where w is the model's parameters array and $f(w)$ is the loss function. The optimization problem requires the definition of an optimizer that defines the update rules for the parameters in order to minimize the loss function. The optimization is performed with the gradient algorithm:

$$w_{i+1} = w_i + \Delta w_i, \qquad \Delta w_i = -\eta \nabla_w f(w)|_{w_i}, \quad (5)$$

where η is the learning rate and the gradient $\nabla_w f(w)|_{w_i}$ is typically calculated, using the back-propagation algorithm [10]. This algorithm forwards the input through the network and computes the output. Then, the loss value is computed and the algorithm applies the chain rule, from the output to the input layer, in order to compute each parameter's contribution in the loss [7]. Thus, Δw_i is obtained and used to update the weights for the next iteration. The most commonly used gradient descent variants are SGD, Adam, RMSProp, AdaDelta, and AdaGrad [10], and in this work we used the Adam optimizer that is computationally efficient and combines the advantages of two popular optimization methods: the ability of AdaGrad to deal with sparse gradients, and the ability of RMSProp to deal with non-stationary objective functions [13].

2.5. State-of-the-Art CNNs

AlexNet [15] was published in 2012 and is considered one of the most influential papers published in computer vision. It presents a CNN architecture that achieved great results in ImageNet Challenge [17]. It comprised 5 convolutional and 3 fully connected layers. ReLU is the non-linear activation function used in every hidden layer.

VGG [18] was published in 2015, and presents six CNN configurations whose main difference is the number of layers. The authors concluded that increasing the number of layers helped to improve the results.

ResNet [11] and DenseNet [12] were published in 2017 and present architecture design techniques to avoid the vanishing gradients problem. ResNet introduced a building block for residual learning with shortcut connections, improving the flow of information throughout the network. The architecture proposed in DenseNet connects all the layers directly with each other, in order to ensure the maximum information flow between the layers in the network. With these techniques it was possible to implement deeper neural networks without degrading their performance.

In addition to a good network architecture design, there are other techniques that can optimize the results. Some of the techniques used in these works are: dropout [12, 15, 18, 19], data augmentation [11, 12, 15, 18], scale jittering [18] and ensemble of models [11, 18, 19].

3. Methodology

In this work, we compared three approaches: (i) a flat classifier, that performs a single decision, whose output includes all the final categories, (ii) a hierarchical classifier, that makes intermediate decisions, and then predicts the final category, mimicking the method followed by a dermatologist, and (iii) a mixed model that combines the previous approaches, trying to capture the strengths of each one.

3.1. Flat Classifier

The flat classifier does not incorporate the hierarchical knowledge regarding skin lesions, and performs a single decision for each image, whose output includes all the possible categories.

This model receives a skin lesion image as input and comprises several convolutional layers inspired in state-of-the-art architectures. The last layer is a fully-connected layer (FCL) with softmax activation, whose number of units is the number of different categories in the dataset, as shown in figure 2. To improve generalization, dropout with 50% probability was applied before the decision layer as in [5].

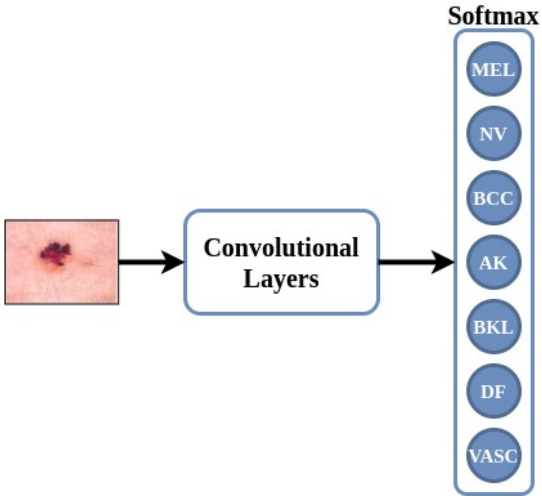

Figure 2: Architecture of the flat classifier. The last layer is a FCL with seven units, corresponding to the seven different categories in the dataset used in this work.

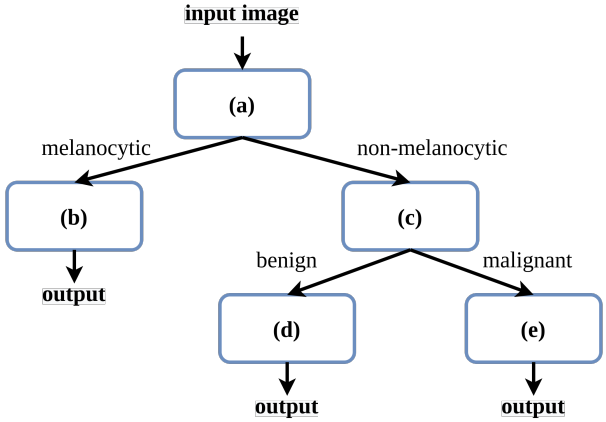

Figure 3: Hierarchical inference procedure of a skin lesion. For simplicity, we refer to a classifier by its letter. For instance the classifier that diagnoses non-melanocytic malignant lesions will be referred by **(e)**.

We compared three state-of-the-art CNN architectures that achieved great results in the ImageNet Challenge [17]: VGG [18], ResNet-101 [11], and DenseNet-121 [12]. We trained each model using two approaches: from scratch in 50 epochs, and using the CNN pre-trained on the ImageNet dataset and fine-tuned it for 20 epochs.

3.2. Hierarchical Model

The hierarchical model exploits the hierarchical organization of skin lesions. When predicting the category of each skin lesion image, it makes intermediate decisions, and then predicts the final output, mimicking the approach followed by a dermatologist. There are three levels of decision: (i) melanocytic vs. non-melanocytic, (ii) benign vs. malignant, and (iii) final diagnosis.

The dataset [9] used in this work has only one class of melanocytic benign lesions, and only one class of melanocytic malignant lesions, therefore the decision level benign vs. malignant can be omitted on this case. Figure 3 illustrates the hierarchical inference procedure of a skin lesion in our dataset, using five different classifiers. We implemented two hierarchical models:

The first model comprises five different networks. Each one preserves the flat classifier architecture, except for the output layer which is adapted according to the classes that it aim to classify. Each network is trained independently from the others with a specific training set. For instance, classifier c) is trained with all the non-melanocytic lesions.

The second model relies on a single network which comprises a CNN, whose output is flattened with a global average pooling operation. Then, a dropout layer is added with probability of 50% and five fully connected layers are stacked on top. Since the features are shared by the five classifiers, the model is trained with the whole training set, which poses an additional challenge: each kind of lesion should only update a part of the classifiers. For instance, a melanoma lesion (melanocytic and malignant) should only update the classifiers a) and b). This was accomplished by setting the loss to zero when a lesion should not update a classifier. The architecture of this model is shown in figure 4.

3.3. Mixed Model

During the inference stage, the hierarchical model has to make more than one decision, to reach the final class. If at least one decision is incorrect, the final diagnose will be automatically incorrect. Therefore, it is very important that in each decision, the hierarchical model is very confident, to prevent error propagation.

Our approach to assess the confidence of each decision was to evaluate the difference between the two largest softmax probabilities. If the difference is large, we assume that the classifier is very confident in the output class, however, if the difference is small, we assume that the classifier is

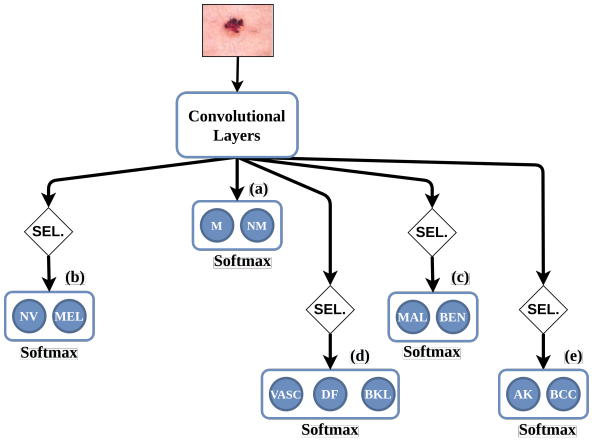

Figure 4: Training architecture of the hierarchical model, with a single feature extractor.

undecided between those two classes.

Each of the five classifiers is associated with a threshold $\eta \in [0, 1]$ (or $[0, 100]\%$). During the inference stage, if at any classifier's output, the difference between the two largest softmax probabilities is less then its threshold, the image is sent to the flat model, otherwise it is classified with the hierarchical model.

We denote $\vec{\eta}$ to represent the array of five η_i, for each classifier. In order to tune η, we used the following method:

1. Vary each η_i from 0% to 100% with a step of 5%, using the same threshold in the five classifiers and initialize each η_i with the the threshold that led to the best performance in the validation set.

2. Initialize a array Δ which defines how each η_i will be changed in the following steps.

3. Assign δ with the next value of Δ.

4. Evaluate the model in the validation set, with $\vec{\eta_1} = (\eta_a - \delta, \eta_b, \eta_c, \eta_d, \eta_e)$, and $\vec{\eta_2} = (\eta_a + \delta, \eta_b, \eta_c, \eta_d, \eta_e)$

5. Update $\vec{\eta}$ with the best combination among $\vec{\eta_1}$, $\vec{\eta_2}$, and $\vec{\eta_3} = (\eta_a, \eta_b, \eta_c, \eta_d, \eta_e)$

6. Repeat from step 4. but now changing the next η_i, until testing/updating every η_i.

7. Repeat from step 3. with the next value of Δ, until all values are tested.

In this work we used $\Delta = (10, 5, 2)\%$.

3.4. Techniques to Improve Performance

In order to improve the performance of the models described previously, same strategies were adopted.

In order to prevent overfitting, we applied data augmentation to the training set. For each training image, two artificial images were generated, corresponding to the vertical flip and a 90º rotation (counter clockwise) of the original image.

Also, the training set is unbalanced with respect to the number of examples for each category, therefore we adopted the same strategy as [5], assigning a weight on the loss function for each class:

$$w_C = \frac{N}{N_C}, \tag{6}$$

where w_C is the weight of class C, N is the number of images in the training set, and N_C is the number of training images from class C. This technique prevents the model to be biased towards the most frequent categories, just because there are more training examples of this category in the dataset.

3.5. Dataset with Reduced Dimensions

The medical images necessary for the development of these methods are often not available. This is more and more frequent given the strict data protection policy. Additionally, many algorithms (especially, supervised learning algorithms) require labels created by doctors who have other and more important tasks and responsibilities. Therefore, there is a need to develop algorithms that are robust in the presence of little data. In this work we evaluate the robustness of the implemented models with respect to the amount of the available data, by comparing the models' results obtained when trained in the original dataset and in a new dataset, about half the size. The original dataset was split in half, and the same strategies and procedures were applied during this analysis (i.e., data augmentation was performed with the same strategies, class weights were used and every training hyperparameter was maintained). The results were evaluated with the flat and hierarchical models. Then, we compared the benefits of using these models in the original dataset and in the dataset with reduced dimensions.

4. Results & Discussion

In this section we introduce the dataset and metrics used to compare the models' performance. Finally, we summarize and discuss the models' performance. In all the experiments, we set the initial learning rate to 10^{-5}, and the learning rate was divided by 10 at 50 %, and 75 % of the total number of training epochs, as in [12]. The training was carried out by optimising the categorical cross entropy,

using the Adam optimizer [13], with a mini-batch of size 10.

4.1. Dataset ISIC 2018

The dataset used in this work is the one released for the ISIC 2018 challenge [9]. It comprises 10,015 images with ground truth, used for training and validation, and 1,512 images without ground truth data (test set). The models' evaluation on the test set is performed on an online platform [1], with the hyper-parameters tuned in the validation set. This dataset comprises seven categories of skin lesions, as shown in figure 5: melanoma (MEL), nevus (NV), basal cell carcinoma (BCC), aktinic keratosis (AKIEC), benign keratosis (BKL), dermatofibroma (DF), and vascular (VASC).

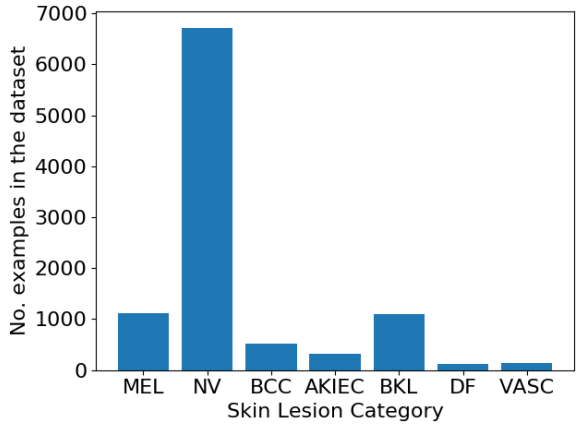

Figure 5: Distribution of the number of examples of each skin lesion category in ISIC 2018 dataset.

This dataset was randomly divided in training (∼80%) and validation (∼20%) sets, resulting in a training set comprising 8,004 images and a validation set with 2,011 images.

In order to prevent overfitting, data augmentation was performed on the training set, as described in section 3.4, resulting an augmented training set that comprises 24,012 images.

4.2. Performance Metrics

We have adopted two metrics to compare the models' results on the validation set. The metric that guided our decisions is the Balanced Accuracy ($BACC$):

$$BACC = \frac{1}{C}\sum_{i=1}^{C} p(\hat{y}=i|y=i) = \frac{1}{C}\sum_{i=1}^{C} P_{ii}, \quad (7)$$

where C is the number of different classes, and P_{ii} is the element from the normalized confusion matrix P at row i and column i.

We also computed the metric that we interpreted to be the primary metric used at ISIC 2018 Challenge, $P_{SUCC.}$, which represents the probability of picking randomly an image from the evaluation set and correctly predicting its class:

$$P_{SUCC.} = \sum_{i=1}^{C} p(\hat{y}=i|y=i)\cdot p(y=i) = \\ = \sum_{i=1}^{C} P_{ii}\cdot p(y=i) \quad (8)$$

4.3. Comparison of state-of-the-art CNNs

As stated in section 3.1, we compared the flat classifier with three state-of-the-art CNNs: VGG [18], ResNet-101 [11], and DenseNet-121 [12]. For each architecture, the CNN was evaluated when trained from scratch and with transfer learning and always achieved better results when trained with transfer learning. Table 1 shows the performance of the CNNs pre-trained on the ImageNet dataset [17] and fine-tuned for 20 epochs.

Table 1: Scores obtained with flat classifiers in the validation set, trained with transfer learning ordered from the best to the worst.

CNN	BACC	$P_{SUCC.}$
DenseNet-121	81.0 %	82.6 %
ResNet-101	78.6 %	80.9 %
VGG-19	69.0 %	70.8 %

Since the DenseNet-121 presents better results, all the following architectures were based on this CNN.

4.4. Comparison of Hierarchical Models

As explained in section 3.2, we compared two approaches to implement a hierarchical model. The results are summarized in table 2, as well as the number of parameters used in each model.

Table 2: Comparison of the performance obtained with the hierarchical models in the validation set. The models are identified with "ns" or "s", if they were implemented with non-shared or shared features, respectively.

	BACC	$P_{SUCC.}$	#params
Hier-s	77.6 %	82.9 %	7,048,779
Hier-ns	77.3 %	82.5 %	35,198,795

The hierarchical model with shared features among the five classifiers requires less resources to train and store the model's weights, achieving a better performance in the validation set. However, both models achieved a worse performance when compared to the flat classifier's performance.

The final decisions of a hierarchical classifier are performed in the last level of the hierarchy. How-

ever, the errors produced in previous levels of decision are propagated to the subsequent levels and prevent the subsequent decisions from being correct. This error propagation phenomenon caused a decrease in the performance of the hierarchical models.

4.5. Mixed Model using Hierarchical with Non-shared Features

In order to reduce the number of predictions affected by the error propagation phenomenon in the hierarchical model, a mixed model of a flat and a hierarchical classifier was implemented, as described in section 3.3. In this section present the results obtained with the mixed model using the best flat classifier and the hierarchical model with non-shared features.

We assess if for any of the classifiers' outputs the difference between the two largest softmax probabilities is less than a threshold. For the positive cases, the image is sent to the flat model, otherwise it is classified with the hierarchical model. We denote $\vec{\eta} = (\eta_a, \eta_b, \eta_c, \eta_d, \eta_e)$, whose elements correspond to the threshold values from classifiers **(a)** to **(e)**. The hyper-parameters η_i were tuned using the validation set.

As described in section 3.3 we started to vary the five thresholds from 0% to 100%, with a step of 5%, and annotated the performance of these models. Figures 6 and 7 reports the scores obtained with these models in the validation set.

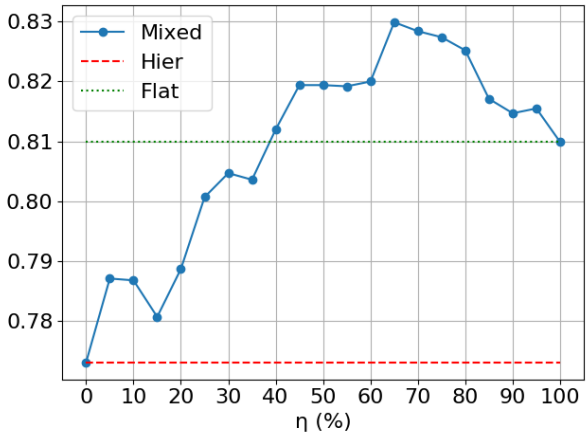

Figure 6: $BACC$ obtained with the mixed model in the validation set, using different thresholds shared by the five classifiers.

We observe for both metrics that initially, by increasing η, the performance has an upward behaviour. After reaching the maximum value, performance starts to degrade. After this value, the classifiers of the hierarchical model are confident enough and it is not advantageous to send the image to the flat classifier. We observe that the mixed model performs better than the flat classifier, in terms of $BACC$, only if $\eta \geq 40\%$. In terms

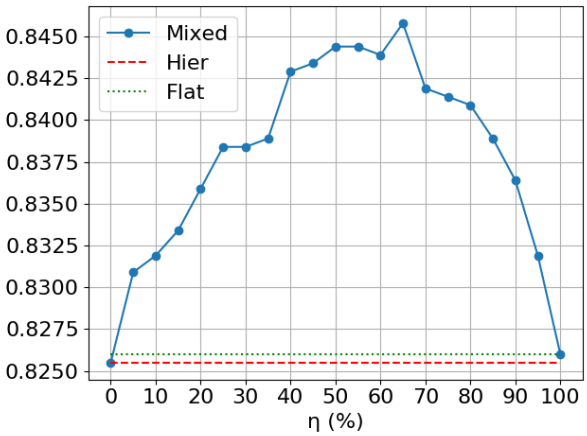

Figure 7: $P_{SUCC.}$ obtained with the mixed model in the validation set, using different thresholds shared by the five classifiers.

of $P_{SUCC.}$, the mixed model outperforms the other methods independently of the chosen η.

When $\eta = 0$, the mixed model is equivalent to the hierarchical model because no image is forwarded to the flat classifier. On the contrary, when it is equal to 100%, we have a flat classifier because all the images are forwarded to the flat classifier.

Then, we adopted the strategy described in section 3.3 to evaluate the mixed model with different combinations of $\vec{\eta}$, not necessarily having the same threshold shared by the five classifiers. Table 3 presents the scores obtained with the best mixed model sharing the same threshold for the five classifiers (65%) and the mixed model obtained with $\vec{\eta} = (63, 65, 68, 65, 65)\%$ that was achieved with the strategy described in section 3.3, that slightly improved the performance.

Table 3: Scores obtained with the mixed models, using the hierarchical model with non-shared features and the best flat model.

	BACC	$P_{SUCC.}$
$\vec{\eta} = (63, 65, 68, 65, 65)\%$	83.3 %	84.6 %
$\vec{\eta} = (65, 65, 65, 65, 65)\%$	83.0 %	84.6 %

The mixed model using the hierarchical model with non-shared features and the best flat model allowed to improve their performances. The strategy to apply different thresholds for each classifier increased slightly the performance of the mixed model.

4.6. Mixed Model using Hierarchical with Shared Features

The same procedures were adopted to develop a mixed model, using the hierarchical model with a single CNN as feature extractor.

Table 4 presents the scores obtained with the best mixed model sharing the same threshold for the five classifiers (55%) and the mixed model obtained with $\vec{\eta} = (65, 55, 55, 40, 55)\%$ that was ob-

tained with the strategy described in section 3.3.

Table 4: Scores obtained with the mixed models, using the hierarchical model with shared features and the best flat model.

	BACC	$P_{SUCC.}$
$\vec{\eta} = (65, 55, 55, 40, 55)\%$	82.9 %	85.2 %
$\vec{\eta} = (55, 55, 55, 55, 55)\%$	82.6 %	85.3 %

4.7. Final Evaluation on the Test Set

Table 5 summarizes the performance achieved by the models in the validation ($BACC$ and $P_{SUCC.}$) and in the held-out test set (score). It is ordered from the best to the worst model in terms of $BACC$ achieved in the validation set.

Table 5: Performance achieved by the models on validation and test sets. The performances obtained with mixed and hierarchical models are identified with "ns" or "s", if the hierarchical model is implemented with non-shared or shared features, respectively.

Model	**BACC**	$P_{SUCC.}$	**Score (test)**
Mixed-ns	83.3 %	84.6 %	74.0 %
Mixed-s	82.9 %	85.2 %	73.0 %
Flat	81.0 %	82.6 %	71.2 %
Hier-s	77.6 %	82.9 %	69.5 %
Hier-ns	77.3 %	82.5 %	71.3 %

The flat model performs better than the hierchical models in the validation set. Regarding the test set, it performs better than the hierarchical model with shared features, and presents similar results to those of the hierarchical model with non-shared features. Such behavior may be due to the fact that a pure hierarchical network is penalized with the wrong decisions of levels prior to the final decision.

With respect to the hierarchical models, the one with a single CNN as feature extractor performed slightly better in the validation set, but achieved the worst performance in the test set, suggesting a worst capacity of generalization. These classifiers are penalized with the wrong decisions of levels prior to the final decision and the flat classifier achieves a greater $BACC$ in validation set. Also, the flat classifier's score obtained in the test set is similar to the best score achieved by a hierarchical model.

The introduction of a criterion that allows forwarding the image to the flat classifier, when the hierarchical network is not sufficiently confident, allowed the implementation of a mixed model, which achieved the better performance when compared to the corresponding flat and hierarchical models.

Except for the hierarchical model with shared features, the scores obtained in the test set present a similar behaviour as the scores obtained in the validation set, suggesting that the models are capable of generalization, even when tuning hyperparameters (η) on the validation set.

Figure 8 presents the normalized confusion matrix obtained in the validation set with the mixed model implemented with the best flat classifier and the hierarchical model with non-shared features, using $\vec{\eta} = (65, 65, 65, 65, 65)\%$.

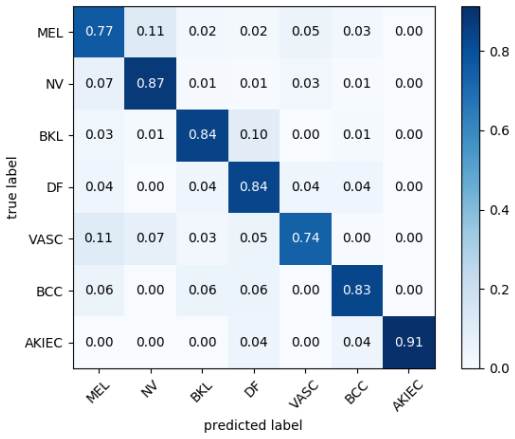

Figure 8: Confusion matrix obtained in the validation set with the mixed model, using $\vec{\eta} = (65, 65, 65, 65, 65)\%$.

This model presents good results, since the lowest score is 74 % for vascular (VASC) lesions. By inspecting the confusion matrix, we observe that melanoma (MEL) is sometimes predicted as nevus (NV), and vice-versa. Also, the vascular lesions that are incorrectly diagnosed, most of the times are predicted as MEL or NV lesions.

4.8. Evaluation of Models using Dataset with Reduced Dimensions

Since the amount of available data may influence the models' performance, we decided to evaluate them in a new dataset, about half the size. The dataset released for the ISIC 2018 challenge comprises 10,015 images with ground truth. Each image of this dataset was included in the a new training set with probability of 40 %, and in a new validation set with a probability of 10 % (each image could only belong at most to one set). Finally, we applied data augmentation to the training set, as described in section 3.4.

Table 6 summarizes the results obtained in the test set with the models trained and validated in the original and reduced datasets. A relative deviation is computed in order to assess the performance decrease in each approach.

Table 6: Comparison of the scores achieved by the flat and hierarchical approaches on the test set, with the models trained and validated in the original or reduced datasets. "s" stands for shared features and "ns" stands for non-shared features.

Approach	Original	Reduced	Rel. Dev.
Flat	71.2 %	70.9 %	- 0.421 %
Hier-ns	71.3 %	62.9 %	- 11.8%
Hier-s	69.5 %	46.1 %	- 33.7 %

We conclude that reducing the size of the dataset affected mostly the hierarchical models, in particular the hierarchical model with shared features. On the other hand, the flat approach almost maintained its performance, which is impressive since it was trained and validated in a halved dataset. The hierarchical approaches require enough examples of each class in order to capture the hierarchical structure of the dataset. If this is not possible, the flat approach may be the preferred one.

5. Conclusions

The main goal of this thesis was to compare the performance of flat and hierarchical models, regarding the classification of skin lesion images. Also, we wanted to assess the effects of using a hierarchical model in a smaller dataset. The objectives have been accomplished.

Regarding the classification of dermoscopy images, flat classifiers do not take advantage of their hierarchical structure and the classification relies on a single decision including all the final categories.

On the other hand, a hierarchical model takes advantage of the hierarchical structure of skin lesions but suffer from the fact that they have to make decisions prior to reach a final category. This intermediate decisions lead to the error propagation phenomenon causing a decrease in accuracy when compared to the one obtained by the flat model. Moreover, a hierarchical classifier performs better in larger datasets.

In order to capture the strengths of both approaches, a mixed model was implemented, which followed a criterion to forward the dermosocopy image to the flat classifier, when the hierarchical classifier was not confident enough. This model allowed to improve the performance, when comparing to the flat and hierarchical results. Moreover, this approach presented good results and a good generalization capability, even when fine-tuning some hyper-parameters in the validation set.

Finally, we compared three state-of-the-art CNNs: VGG-19, ResNet-101, and DenseNet-121. DenseNet-121 outperformed the others in all the experiments. In DenseNet, each layer receives directly all the feature maps obtained with the preceding layers. This provides a strong gradient flow and improves the features' propagation to the last layer.

5.1. Future Work

As a follow-up to this work, some aspects can be studied. For instance, we provide a few examples:

- Structure the dataset using a different hierarchical organization with the intermediate decision levels in the following order: (i) benign vs. malign, and (ii) melanocytic vs. non-melanocytic, instead of (i) melanocytic vs. non-melanocytic, and (ii) benign vs. malign.
- The dataset used in this work only comprised a single category for melanocytic benign lesions, as well as for melanocytic malignant lesions, which allowed us to omit the decision level melanocytic vs. non-melanocytic. We think that should be interesting to assess the effects of using a hierarchical model in a dataset that specified the different categories of benign melanocytic lesions.

References

[1] ISIC Challenge.

[2] Task 3: Lesion Diagnosis — ISIC 2018.

[3] G. Argenziano and H. P. Soyer. *Interactive atlas of dermoscopy*. Edra Medical Publishing & New Media, 2000.

[4] C. Barata, M. E. Celebi, and J. Marques. A survey of feature extraction in dermoscopy image analysis of skin cancer, 2019.

[5] C. Barata and J. Marques. Deep Learning for Skin Cancer Diagnosis with Hierarchical Architectures, 2019.

[6] M. E. Celebi, N. Codella, and A. Halpern. Dermoscopy Image Analysis: Overview and Future Directions. {*IEEE*} *Journal of Biomedical and Health Informatics*, 23(2):474–478, 2019.

[7] F. Chollet. *Deep Learning with Python*. Manning Publications, 2017.

[8] N. Codella, D. Gutman, M. Celebi, B. Helba, M. A. Marchetti, S. W. Dusza, A. Kalloo, K. Liopyris, N. Mishra, H. Kittler, and A. Halpern. Skin Lesion Analysis Toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC). 10 2017.

[9] N. Codella, V. Rotemberg, P. Tschandl, M. Celebi, S. Dusza, D. Gutman, B. Helba,

A. Kalloo, K. Liopyris, M. Marchetti, H. Kittler, and A. Halpern. Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC). 2 2019.

[10] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. 12 2015.

[12] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely Connected Convolutional Networks. 8 2016.

[13] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. 12 2014.

[14] H. Kittler, H. Pehamberger, K. Wolff, and M. Binder. Diagnostic accuracy of dermoscopy. *The Lancet Oncology*, 3(3):159–165, 2002.

[15] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. 2012.

[16] Y. Lecun, Y. Bengio, and G. Hinton. Deep Learning. 2015.

[17] 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. 9 2014.

[18] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. 9 2014.

[19] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.