# Pigment Network Detection in Dermoscopy Images using Deep Learning

Pedro Filipe Coimbra de Sousa
Instituto Superior Técnico
Universidade de Lisboa
Email: pedrofcsousa@tecnico.ulisboa.pt

***Abstract*—Melanoma is considered to be the most dangerous form of skin cancer. However, if the melanoma is diagnosed in its early stages, it can be easily cured. Some medical techniques have been proposed to improve the performance of early melanoma diagnosis, including dermoscopy, which combines illumination techniques with magnification to obtain a better visualization of the skin lesion. Based on this technique, several medical procedures, such as the ABCD rule and the 7-point checklist, were developed to simplify not only the distinction between the different types of lesions but also the detection of malignant melanomas. These procedures rely on the detection of dermoscopic features and colors in dermoscopy images of the lesion. One of the most relevant dermoscopic structures detected by these procedures is pigment network. Some works were published addressing the automatic detection of this structure, but the majority of them only focus on detecting it and not localizing it, which would be of great importance for medical experts. Thus, this work proposes a system for the automatic detection and segmentation of pigment network, using a deep learning approach. The developed system was based on a well-known convolutional neural network (CNN) architecture called U-Net, which was designed for biomedical image segmentation tasks. This system receives as input a dermoscopy image and generates a binary mask where the presence of pigment network is highlighted. This method was tested against a dataset of 600 images belonging to the ISIC database, achieving a sensitivity of 93.8% and a specificity of 84.2%, which proves the reliability of the proposed system.**

## I. Introduction

Skin cancer is one of the most common forms of cancer and may take the form of a benign or malignant lesion. Melanoma is considered to be the deadliest of malignant skin cancers, mainly due to its ability to metastasize [1]. Although it comprises only 4% of all skin cancers, it is responsible for 80% of skin cancer-related deaths [2]. However, if the melanoma is diagnosed in its early stages, it is easily curable by performing a simple excision of the lesion [1]. Thus, it is crucial to develop reliable automatic systems for melanoma detection.

The dermoscopy technique was developed to improve the performance of early melanoma diagnosis, combining special illumination techniques with magnification to obtain a better visualization of the skin lesion [3]. Previous studies showed that comparing to naked-eye analysis, dermoscopy improves the melanoma diagnostic accuracy by 10-27%. However, this technique can only improve the diagnostic performance if the dermatologists are trained formally [4], otherwise, some lesions can be underdiagnosed, leading to dangerous misclassifications. Based on this technique, some medical procedures were developed to simplify the classification between the different types of skin lesions and detect malignant melanomas. These diagnostic algorithms, such as the ABCD rule [5] and the 7-point checklist [6], rely on the detection of dermoscopic features and colors that are observed in a dermoscopy image of the lesion. One of the most relevant dermoscopic structures analysed by these algorithms is the pigment network [1].

The presence of pigment network is accounted when distinguishing between a melanoma and other skin lesions, being moreover a feature present in all medical algorithms for the diagnosis of melanomas. Furthermore, when an atypical network is present, it commonly results in the lesion being classified as melanoma. Thus, it is essential to develop methods to detect it, since it has a great importance in melanoma diagnosis [7].

Despite its automatic detection being a very complex problem, several works have been published addressing the detection of this structure [8, 9, 10, 11, 12, 13, 14], however, only a few focus not only on detecting it but also localizing it in dermoscopy images, which is of major interest for the dermatologists when diagnosing a lesion. Thus, the aim of this thesis is to carry out the automatic detection and segmentation of pigment network using a deep learning approach. The developed system was based on a well-known Convolutional Neural Network (CNN) architecture, especially designed for biomedical image segmentationt tasks.

The advent of deep learning has had a significant impact on many areas in machine learning [15], dramatically improving the state-of-the-art in different tasks such as object detection, speech recognition, language translation [16] and also dermoscopy image analysis [17]. So far, very few investigations have considered deep learning techniques towards the automatic detection of pigment network, with the exception of some successful works such as the one presented by Kawahara et. al [18]. This is highly related to the fact that deep learning architectures require a large amount of data and only recently were provided databases with enough examples, such as the DermNet [19] and ISIC [20] datasets, that would make this area benefit from the power of deep learning.

## II. CNN ARCHITECTURES FOR IMAGE SEGMENTATION

Although CNNs are widely used in image classification problems, there are many visual tasks, especially in medical image analysis, where the desired output should include localization, providing not only the classes but also additional information regarding the spatial localization of those classes. Thus, it is required that a class label is assigned to each pixel of the image, which corresponds to the main idea behind semantic image segmentation using CNNs [21].

Spatial information is especially important for semantic segmentation tasks. Hence, in 2015, Long et al. [22] proposed the Fully Convolutional Network (FCN) to overcome this limitation. The idea behind their approach was to take advantage of existing CNNs as powerful visual models that are able to learn hierarchies of features. They transformed those existing and well-known classification models, such as AlexNet [23], GoogLeNet [24] and ResNet [25], into convolutional ones, by replacing the final fully connected layers with transposed convolutional ones to output spatial maps instead of classification scores. By applying this operation, the original spatial dimensions of the input image can be recovered while performing semantic segmentation at the same time. This network has made it feasible to train models for pixel-wise semantic segmentation in an end-to-end fashion. Figure 1 depicts the architecture of the FCN.

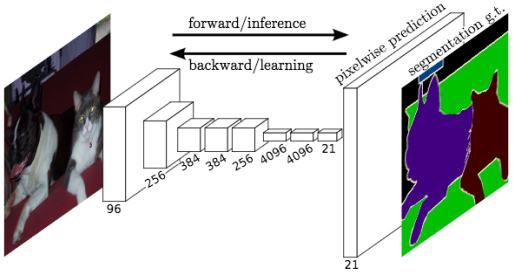



Figure 1: FCN architecture (adapted from [22]); the number below the boxes represent the number of filters of each layer

Since Long et al. popularized the CNN architecture for dense predictions without any fully connected layers, a large number of FCN-based [26, 27, 28, 29, 30, 31, 32] methods have been proposed, promoting the application of deep learning strategies to image semantic segmentation. Recently, U-Net [33] became one of the most popular networks for biomedical image segmentation problems. Essentially, U-Net is a deep convolutional network that learns to segment images in an end-to-end setting, which means that it receives as input an image in its rawest form and produces an output segmentation map. The architecture of this network consists of a contracting path where the image's context is captured and a symmetric expanding path that enables precise localization, which is illustrated in Figure 2.

In Figure 2, each blue box represents a multi-channel feature map, while each white box represents copied feature map. On top of the boxes is indicated the number of channels and the

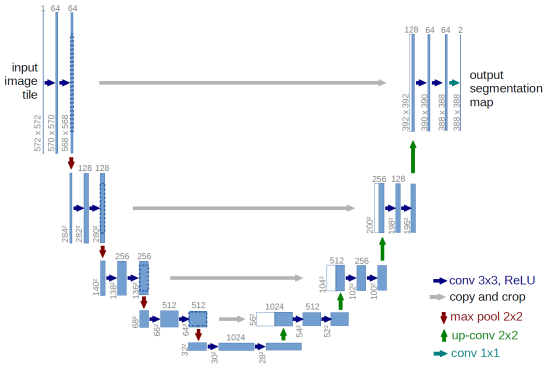



Figure 2: U-Net architecture (adapted from [33]).

size of the image is given at the lower left edge of each box. The arrows show the different operations.

This type of network merges a convolutional network architecture (contracting path on the left side) with a deconvolutional network architecture (expanding path one the right side).

The contracting path is composed of a repetitive pattern of two $3 \times 3$ convolutions, each followed by a Rectified Linear Unit (ReLU) and a $2 \times 2$ max pooling operation with a stride of 2 for downsampling. At each downsampling step, the number of feature channels is doubled.

Regarding the expansive path, every step includes an upsampling operation of the feature map obtained in the contracting path, followed by sequences of 2x2 transposed convolutions, that halves the number of feature map channels. At each step, a concatenation of the resulting feature map with the correspondingly feature map obtained from the contracting path is performed, followed by two $3 \times 3$ convolutions and a ReLU layer after each convolution.

The entire network has 23 convolutional layers, where the last layer is used to map each component feature vector related to the desired number of output classes.

## III. PROPOSED METHODOLOGY

This section presents the techniques that were used in the development of a system to detect pigment network in dermoscopy images. A deep learning approach was taken, where a CNN model was developed and trained using skin lesion images from a publicly available dataset [20].

### A. Overall architecture

The architecture of the proposed system is shown in Figure 3. The main components of the system are two identical pre-processing modules, a CNN architecture module and a training module. The system receives as input a dermoscopy image which is pre-processed before being fed to a trained CNN model, generating as output a binary mask, where white indicates the presence of pigment network and black corresponds to the absence of pigment network. The CNN model is trained using a set of dermoscopy images and their corresponding groundtruth segmentations of pigment network, which are also pre-processed.

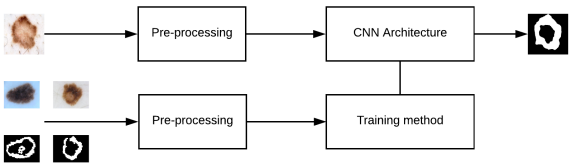



Figure 3: Block diagram of the system's architecture

In the next sections, it will be detailed all the procedures taken in each one of the system's modules.

### B. Pre-Processing

Before feeding the dermoscopy images into the CNN architecture, the raw input images are subjected to a set of pre-processing transformations. In this thesis, the pre-processing procedures applied to the input images are as follows:

- **Image resizing**. The input images were all of different sizes, varying from $576 \times 768$ to $6748 \times 4499$. Thus, to reduce the computational cost and the complexity of the problem, all images of the dataset were resized to a constant value of $256 \times 256$.
- **Image channels reduction**. For some experiments, the input RGB images were converted to grayscale images, reducing the images depth from 3 to 1. The grayscale conversion was performed using the ITU-R 601-2 luma transform [34], which is given by

$$T = 0.299R + 0.587G + 0.114B, \tag{1}$$

  where $R$ denotes the red color channel, $G$ is the green color channel and $B$ is the blue color channel. This transform was chosen mainly due to its implementation easiness.
- **Image normalization** By dividing each input image by 255, each color channel was normalized from a range between 0 and 255 pixel values to a range between 0 and 1 normalized values.

### C. CNN Architecture

Since the goal is to segment the pigment network regions in dermoscopy, the CNN architecture that was chosen is the U-Net [33], which was specifically designed by O. Ronneberger et al. to deal with biomedical image segmentation problems. The architecture of this network is described in Section II and it is illustrated in Figure 2. However, a few changes were made to the architecture of the network.

First, the depth of the channels at each stage of the network, including both convolution and deconvolution parts, was reduced by a quarter, i.e. instead of using 64, 128, 256, 512 and 1024 channels, 16, 32, 64, 128 and 256 channels were used. The reason behind this adjustment is related to limitations of hardware, mainly memory issues (see Section IV-C for the implementation details).

The other alteration that has been done is that the final layer was reduced from two layers to only one layer. The pixel-wise classification is binary, i.e. each pixel is classified as foreground (white) or background (black), so it is possible to use only one output layer with a sigmoid activation function (2) and reduce the number of parameters of the network.

$$f(\hat{y}) = \frac{1}{1 + e^{-\hat{y}}}. \tag{2}$$

Thus, instead of the output pixel $p(x, y)$ belonging to the class of the node with the highest probability $p_{ij}$, it is considered foreground if its probability is above a threshold $\lambda$ and background if it is below that given threshold $\lambda$, as shown in equation (3).

$$p(x, y) = \begin{cases} 1, \ if \ p_{ij} \geq \lambda \\ 0, \ otherwise \end{cases} \tag{3}$$

### D. Training method of the network

The training methodology follows the block diagram depicted in Figure 4. Since the training is done from scratch, the weights of the model are randomly initialized. The training data is split into mini-batches, wherein each iteration (smaller loop in the figure), one mini-batch is feedforwarded through the CNN architecture. Then, the generated binary mask is compared with the desired binary output using a loss function (see next section for details of the used loss functions). After that, the derivative of the loss function is computed and the error is backpropagated through the network from the end to the start. Once all derivatives are computed, the weights are updated using a gradient-based optimization algorithm called Adam [35] and another mini-batch is feedforwarded through the CNN and the procedure repeats. After all the mini-batches have been feedforwarded once through the CNN (what is called an epoch), the training set is shuffled (bigger loop in the figure which contains the dashed line) and the aforementioned mini-batch training is repeated. The described procedure is performed for a chosen number of epochs. The training data is shuffled at each epoch to make sure that the model remains general and overfit less. By shuffling the data after each epoch, the risk of creating batches that are not representative of the overall dataset decreases and the estimate of the gradient will be better; it has been observed that if the order in which the mini-batches are visited is changed for each epoch, a faster convergence is obtained [36].

*1) Definition of hyperparameters:* For the learning process, some parameters must be carefully considered in order to achieve the best possible performance. Thus, the parameters to be defined before the training algorithm are the following:

- **Batch size**. The batch size is the number of samples fed to the network in one training iteration, in order to make one update to the model parameters. Since the entire dataset cannot be propagated into the neural network at once for memory limitations, it is divided into batches, which makes the overall training procedure require less memory and become faster. It should be highlighted that the higher the batch size is, the more memory will be needed and the slower is the training procedure.

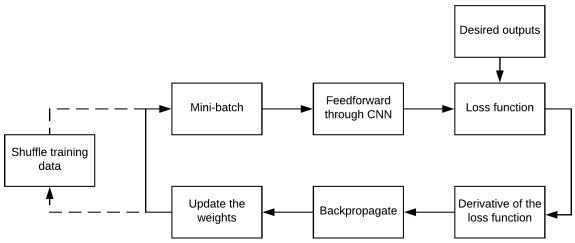



Figure 4: Training methodology. The small loop corresponds to each mini-batch training (full line). The bigger loop is performed once per epoch (loop including dashed line), after all mini-batches have been feedforwarded once in the current epoch.

- **Epochs**. The number of epochs denotes how many times the entire dataset has passed forward and backward through the neural network, i.e., one epoch is when every image has been seen once during training. Nevertheless, this concept should not be confused with iterations. The number of iterations corresponds to the total number of forward and backward passes, with each pass using a batch and depends on the the batch size, the number of epochs and number of training images. It is computed as follows:

$$\#iterations = \frac{\#epochs \times \#training\ images}{batch\ size} \quad (4)$$

- **Loss function**. The loss function evaluates the inconsistency between the predicted value $\hat{y}$ and the groundtruth label $y$ in every batch.
For the purpose of this thesis, two loss functions were tested, namely the Cross-Entropy (CE) and the Weighted Cross-Entropy (WCE). The cross-entropy loss function for binary classification is given by

$$\mathcal{L}(y,\hat{y}) = -\sum_{i=1}^{N}\left[y_i\, log\Big(\frac{e^{\hat{y}_i}}{1+e^{\hat{y}_i}}\Big) + (1-y_i)\, log\Big(\frac{1}{1+e^{\hat{y}_i}}\Big)\right], \quad (5)$$

where $N$ denotes the total number of training images.
Since the training dataset is imbalanced (see Section IV-D), i.e., the number of foreground pixels (pixels where pigment network is present) is much smaller than the number of background pixels (pixels where pigment network is absent), it is necessary to find a way to overcome this issue. To tackle this problem, it is introduced a weight as a multiplicative coefficient for the positive class, i.e. foreground pixels, in the loss function. Thus, the weighted cross-entropy loss function for binary classification is as follows:

$$\mathcal{L}(y,\hat{y}) = -\sum_{i=1}^{N}\left[y_i\, log\Big(\frac{e^{\hat{y}_i}}{1+e^{\hat{y}_i}}\Big)\, w + (1-y_i)\, log\Big(\frac{1}{1+e^{\hat{y}_i}}\Big)\right], \quad (6)$$

where $w = \frac{\#total\ pixels}{\#foreground\ pixels}$.

- **Optimizer**. The optimizer used in this work was the Adam [35], which is a gradient-based optimization algorithm that computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients; the name Adam is derived from adaptive moment estimation. This optimizer was chosen since it has little memory requirements and the hyperparameters $\beta_1$ and $\beta_2$ have intuitive interpretations, requiring little or no tuning.
Besides storing an exponentially decaying average of past squared gradients $v_t$ like RMSprop, Adam also keep an exponentially decaying average of past gradients $m_t$, similar to momentum [37], according to:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \quad (7)$$

$$v_t = \beta_2 m_{t_1} + (1-\beta_2) g_t^2, \quad (8)$$

where $m_t$ and $v_t$ are the estimates of the first moment and the second moment of the gradients respectively, $\beta_1, \beta_2 \in [0,1]$ are hyperparameters that control the exponential decay rate of these moving averages and $g_t = \nabla_{\theta_t} f_t(\theta)$ denotes the gradient, i.e. the vector of partial derivatives of the objective function $f_t$ with respect to the parameters $\theta$ (weights and biases). After that, the learning rate $\eta_t$ is updated using the external learning rate $\eta$, according to

$$\eta_t = \eta \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t} \quad (9)$$

and the parameters $\theta_t$ are adjusted using the Adam update rule given by

$$\theta_t = \theta_{t-1} - \eta_t \frac{m_t}{\sqrt{v_t} + \hat{\epsilon}}, \quad (10)$$

where $\hat{\epsilon}$ corresponds to a small constant for numerical stability.
- **Learning rate**. The learning rate parameter controls the step size for which the weights of a model are updated regarding the loss gradient. The lower its value is, the slower the convergence is but it is ensured that it is not missed any local minimum.

The hyperparameters values chosen are defined in Section IV-D.

*2) Model Evaluation:* As referred in section III-D, the CNN was trained for a fixed number of epochs. However, during training, the model was validated at every 10 epochs, in order to select the best model configuration. Figure 5 illustrates how each model was evaluated to obtain the best possible one.

As depicted in Figure 5, the model is learned from the training data through a learning algorithm. To evaluate how well the model can perform, a predicted algorithm, using the learned weights of the model, generates predictions from a new set of data (validation data) that was not used during the training procedure. Essentially, the new set of data allows the

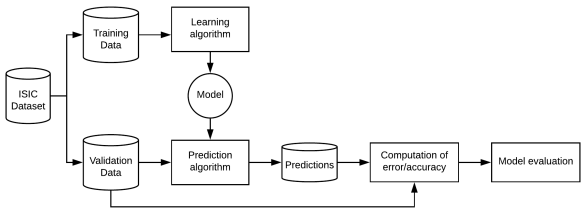



Figure 5: Model evaluation procedure

validation of the model's performance in order to understand if the model is able to generalize to new inputs. After that, the model's performance is evaluated by using a metric that compares the predictions with the desired outputs of the validation data. The metric chosen was the Balanced Error (BE) which is given by

$$BE = 1 - \frac{SE + SP}{2}, \tag{11}$$

where $SE$ is the sensitivity (percentage of correctly classified foreground pixels) and $SP$ is the specificity (percentage of correctly classified background pixels).

In the next chapter, the results of this metric are presented using the validation data as the input of the system. After obtaining for each experiment the model that achieves the lowest BE, these models are used to detect pigment network in the test data.

## IV. Implementation and Results

This section presents the results obtained for the proposed system as well as some implementation aspects.

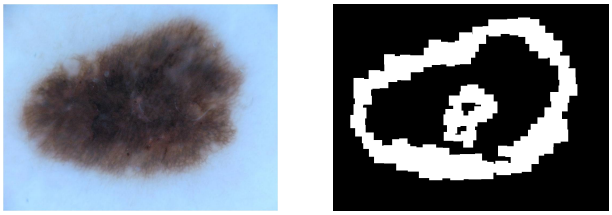
### A. Dataset

The International Skin Imaging Collaboration (ISIC) 2017 Challenge [38] dataset [20] for Skin Lesion Towards Melanoma Detection was used throughout this work .

This dataset of dermoscopy images was very recently made publicly available and it contains 2750 RGB images which are pre-partitioned into 2000 training images, 150 validation images and 600 test images. Furthermore, it is also provided corresponding superpixel masks and superpixel-mapped expert annotation of the presence or absence of pigment network. These superpixel masks were converted to binary segmentation masks, where white denotes the pixels where pigment network is present (foreground) and black corresponds to the background. Figure 6 depicts an example where pigment network is present and its corresponding manual segmentation in a binary mask.

The dataset is composed by malignant and benign lesions, where pigment network may or may not be present. Table I shows the number of dermoscopy images that contain pigment network for each subset.

It should be noted that only recently this database was made publicly available, providing a large number of dermoscopy images with superpixel-level annotations performed by expert dermatologists, which is why this thesis could only be developed in the present days.



(a) (b)

Figure 6: (a) Original image with pigment network; (b) Binary segmentation mask (extracted from the ISIC 2017 dataset [20]).

| Subset | Pigment Net. | Non-pigment Net. | Total images |
|---|---|---|---|
| Training | 1131 | 869 | 2000 |
| Validation | 64 | 86 | 150 |
| Test | 333 | 267 | 600 |

Table I: ISIC 2017 Dataset [20] distribution

### B. Performance metrics

This section addresses the procedures used to quantitatively evaluate the performance of the proposed system for detection of pigment network.

The proposed network detection system generates a binary segmented image that is compared with the groundtruth segmentation using pixel-based statistics [1]. The pixels $p(x, y)$ can be classified as True Positives (TP), False Positives (FP), True Negatives (TN) or False Negatives (FN), where

- **#TP**: Number of correctly detected as pigment network pixels.
- **#FP**: Number of wrongly detected as pigment network pixels.
- **#TN**: Number of correctly undetected pixels.
- **#FN**: Number of wrongly undetected pixels.

The performance of the proposed system is assessed by obtaining the number of TP, TN, FN and FP in test images and by computing the Sensitivity (SE), Specificity (SP), Dice Score , Accuracy (ACC) and Balanced Accuracy (BA) as follows:

- **Sensitivity**, also known as recall, measures the proportion of actual positives that are correctly identified.

$$SE = \frac{\#TP}{\#TP + \#FN} \tag{12}$$

- **Specificity** measures the proportion of actual negatives that are correctly identified.

$$SP = \frac{\#TN}{\#TN + \#FP} \tag{13}$$

[1]It could be possible to assess the performance of the detection algorithm using the superpixel-wise classification instead of the pixel-wise classification. However, this is not the most natural way of evaluating the algorithm since the CNN architecture classifies all the image pixels in an independent way.

- **Dice coefficient** compares the pixel-wise agreement between the groundtruth and its corresponding predicted segmentation mask, measuring how similar these objects are.

$$Dice = \frac{2\,\#TP}{2\,\#TP + \#FP + \#FN}. \tag{14}$$

- **Accuracy** measures the proportion of correct predictions.

$$ACC = \frac{\#TP + \#TN}{\#TP + \#TN + \#FP + \#FN} \tag{15}$$

- **Balanced accuracy** is computed as the average of the sensitivity and specificity.

$$BA = \frac{SE + SP}{2} \tag{16}$$

During the training procedure, the performance metric used (balanced error) was computed using the validation set to obtain the best model configuration. The test set was only used to make the final assessment of the proposed system after the best model configuration was obtained.

### C. Implementation aspects

The proposed method was implemented in Python 3.6.4 based on Tensorflow 1.8.0. Since it involves many complex operations with some of which computationally expensive either in processing time or memory requirements, a Graphics Processing Unit (GPU) was required to train and evaluate the models, using CUDA libraries developed by NVIDIA to compile and perform the parallel computations on the GPU. For this purpose, a personal computer with an Intel Core i7 processor with 8GB of RAM and an NVIDIA GeForce GT 740M with 2GB of memory was used. Even though it has a compute capability of only 3.5, it highly reduces the training time compared to only using a Central Processing Unit (CPU). Instituto Superior Técnico (IST) provided a computer with Intel Core i7 processor with 8GB of RAM (without GPU) where some of the training procedures were undertaken. The training time of the proposed system using only CPU took, on average, 53 hours. When using the personal GPU, the training time took, on average, 14 hours (the training time for all experiments was between 13 and 15 hours). Thus, the hardware available has restricted the number of experiments performed and the number of configurations obtained.

### D. System optimization

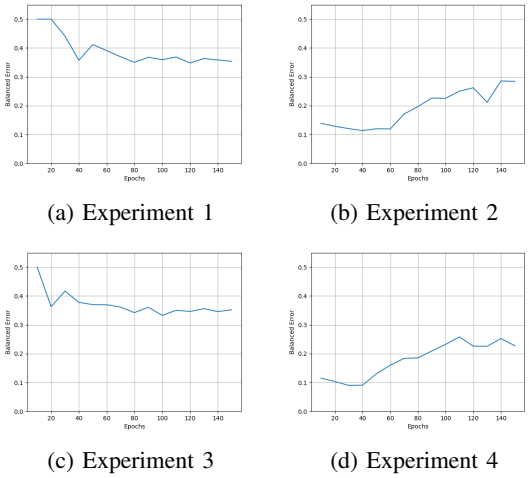
The model was trained with a batch size of 2, not only due to memory limitations but also, because a small batch size brings a higher generalization ability and a faster overall training procedure. Regarding the epochs, a number of 150 epochs was chosen. This choice was based on the examination of the behaviour of the balanced error vs the number of epochs plots using the validation data as the input of the system. It was found that the minimum balanced error was reached before 150 epochs. The optimization process during training was made using a very small learning rate of $\eta = 10^{-4}$ to guarantee a reliable training procedure. Regarding the optimizer hyperparameters, default values of $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\hat{\epsilon} = 10^{-8}$ were used. For the pixel-wise classification, a threshold of $\lambda = 0.5$ was chosen.

As explained in III-D1, to tackle the imbalanced data issue, a weighted cross-entropy loss function that introduces a weight as a multiplicative coefficient for the positive class was used. The value of this weight was obtained inverting the value of the foreground pixels/total pixels ratio using the training set images. Thus, it was found that only $4.5\%$ of the total number of pixels (all images included) corresponded to foreground pixels, which led to a weight of 22.16.

After several preliminary experiments performed, the following ones were selected to be discussed:

- **Experiment 1**. The model was trained using greyscale dermoscopy images as input and cross-entropy as loss function.
- **Experiment 2**. The model was trained using greyscale dermoscopy images as input and weighted cross-entropy as loss function.
- **Experiment 3**. The model was trained using RGB dermoscopy images as input and cross-entropy as loss function.
- **Experiment 4**. The model was trained using RGB dermoscopy images as input and weighted cross-entropy as loss function.

In section III-D2, it was stated that the model parameters were saved every 10 epochs and evaluated using the validation set to obtain the best model configuration. Furthermore, it was introduced that the assessment of the best model would be made using the balanced error as the metric and the validation data as the input of the system. Figure 7 shows the results of this metric applied to each one of the models' outputs resulting from each one of the four experiments.




(a) Experiment 1 (b) Experiment 2

(c) Experiment 3 (d) Experiment 4

Figure 7: Balanced error over epochs for all the tested configurations using validation data.

The results obtained in the validation step are summed up in Table II. The table specifies the minimum balanced error

obtained for each one of the experiments and at which epoch it was reached.

| Experiment | Color | Loss | Minimum BE | Epoch |
|---|---|---|---|---|
| 1 | Greyscale | CE | 34.8% | 120 |
| 2 | Greyscale | WCE | 11.4% | 40 |
| 3 | RGB | CE | 33.3% | 100 |
| 4 | RGB | WCE | **9.0%** | 30 |

Table II: System performance in the validation set

The results presented in Table II suggest that using RGB dermoscopy images instead of greyscale ones as input guarantees slightly better results. Furthermore, it also suggests that using the weighted cross-entropy loss function highly improves the results comparing to using the standard cross-entropy as loss function. However, the validation set is too small to be able to draw conclusions from these validation results. Thus, the model parameters obtained at the epoch where the minimum balanced error was achieved were used to assess the performance of each model using the dermoscopy images belonging to the test set as the input of the proposed system.

### *E. Results*

The performance assessment of each one of the models was obtained by comparing every automatically generated binary mask, i.e. the output of the proposed system, with its corresponding groundtruth mask. For this purpose, the test set was used as the input of the proposed system. The statistical results for the network detection system at a pixel level are presented in Table III.

| Color | Loss | SE | SP | Dice | ACC | BA |
|---|---|---|---|---|---|---|
| Greyscale | CE | 38.8% | **97.4%** | 34.4% | **95.1%** | 68.1% |
| Greyscale | WCE | **94.9%** | 81.6% | 28.1% | 82.0% | 88.2% |
| RGB | CE | 44.9% | 97.0% | **40.3%** | **95.1%** | 71.0% |
| RGB | WCE | 93.8% | 84.2% | 32.9% | 84.5% | **89.0%** |

Table III: System performance in the test set

Table III shows that, as seen in the validation step, the proposed system obtains slightly better results when training the model with RGB images. The balanced accuracy slightly increases, however, the dice coefficient increases $\sim 5\%$, which may indicate that the generated binary masks get more similar to the groundtruth by using RGB images as the input of the proposed system during training. Regarding the loss function, it can be seen that the results highly improve when using the weighted cross-entropy instead of the standard cross-entropy. Even though specificity decreases $\sim 15\%$, sensitivity increases $\sim 50\%$, which implies that more pixels with pigment network are detected. Using the WCE as loss function results in obtaining more pixels classified as *with pigment network* than the ones which actually contain it, but on the other hand, the proposed system gets much better in the task for which it was designed, i.e. detecting pigment network in dermoscopy images. The balanced accuracy increases by roughly $20\%$ which proves the benefit of using this weighted loss function. Accuracy decreases when choosing the WCE as loss function over CE. However, accuracy is not a relevant metric when dealing with segmentation tasks, mainly because it is not sensitive to the fact that the classes are imbalanced. For instance, in Section IV-D it was found that only $4.5\%$ of the total number of pixels of the dermoscopy training images corresponded to foreground pixels. This means that if one used them as the input of the proposed system and the output generated masks showed only background pixels, the accuracy of the model would still be $95.5\%$, when it did not detect pigment network as it was designed for.

To understand how the number of epochs affects the results, Table IV shows the results obtained using the successive saved models during the learning procedure. For this purpose, the model using RGB images as input and the WCE as loss function was used.

| Epoch | Sensitivity | Specificity | Dice Coefficient | Accuracy |
|---|---|---|---|---|
| 10 | **95.8%** | 81.8% | 28.5% | 82.3% |
| 20 | 94.9% | 83.6% | 30.5% | 84.0% |
| 30 | 93.8% | 84.2% | 30.9% | 84.5% |
| 40 | 93.7% | 84.5% | 31.4% | 84.9% |
| 50 | 85.9% | 89.1% | 36.6% | 89.0% |
| 60 | 79.6% | 91.2% | 38.9% | 90.8% |
| 70 | 75.0% | 91.6% | 38.1% | 91.0% |
| 80 | 72.4% | 92.4% | 38.9% | 91.6% |
| 90 | 70.3% | 92.9% | 39.5% | 92.0% |
| 100 | 62.1% | 94.2% | 39.7% | 93.0% |
| 110 | 56.0% | **95.1%** | 39.4% | **93.6%** |
| 120 | 67.1% | 93.5% | **39.8%** | 92.5% |
| 130 | 64.0% | 93.8% | 39.2% | 92.7% |
| 140 | 61.8% | 94.3% | 39.7% | 93.1% |
| 150 | 66.3% | 93.6% | 39.7% | 92.5% |

Table IV: Results over training for Experiment 4

By inspecting Table IV, the first noticeable thing is that using the balanced error in the validation step was a good metric to obtain the model that would guarantee the best results. Furthermore, by inspecting the results over the epochs, one may reckon that during training the model loses its capacity to detect pigment network, leading the results to approach the ones obtained when using the cross-entropy as loss function. This means that even though using the weight as a multiplicative coefficient of the minority class (foreground pixels) helps to detect more pixels actually containing pigment network, it does not emphasize the minority class enough compared to the majority class (background pixels), which may imply that the model is not learning equally from both classes and other ways of dealing with imbalanced data should be pursued. The Dice coefficient increases over the number of epochs mainly due to the decrease in the number of FP identified. Even though the number of TP decreases, the reduction in the number of FP represents a much higher number of pixels that stop being wrongly classified, which is why the increase of the Dice coefficient does not entail much relevance.

Nevertheless, the proposed system achieves an interesting performance on the task for which it was designed, achieving a $SE = 93.8\%$ and $SP = 84.2\%$.

Figures 8 and 9 show examples of satisfactory and poor seg-

mentations performed by the proposed system in the detection of pigment network. In these figures, the left column corresponds to the original image, the middle column is the output of the proposed system and the right column corresponds to the groundtruth binary mask. Figure 8 shows that even when hair is present in the dermoscopic image, the proposed system can still perform a satisfactory detection of pigment network. Furthermore, it also explains the decrease of the specificity when using the WCE as loss function instead of the CE; the automatically generated masks show that the regions with pigment network are slightly wider than the manually segmented regions of pigment network performed by the experts. Figure 9 shows that when there is no sufficient contrast between the lesion area (where pigment network is located) and the background, the proposed system cannot perform a satisfactory segmentation of pigment network. However, it should be noted that the cases where the proposed system underperformed were rare.

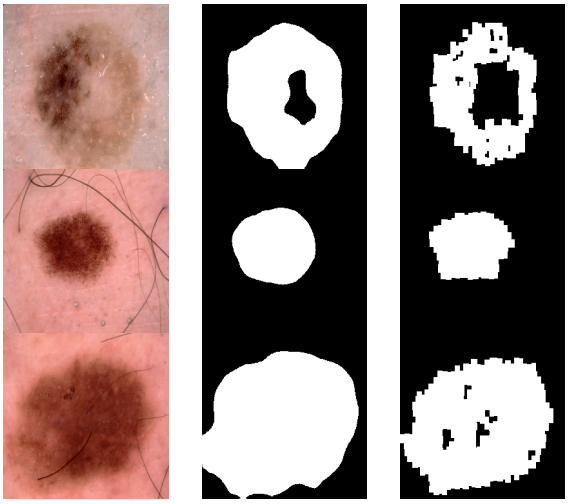


Figure 8: Satisfactory segmentation results: original image (left), automatic segmentation (center) and groundtruth segmentation (right)

## V. Conclusion

The aim of this thesis was the development of an automatic system for the detection of pigment network in dermoscopy images, based on a deep learning approach. This work was only possible thanks to the publication of the ISIC 2017 dataset, which contained 2750 dermoscopy images with superpixel-mapped annotations performed by expert dermatologists.

The proposed system is comprised of two identical pre-processing modules, a CNN architecture module and a training module. The pre-processing modules are responsible for resizing and normalizing the input images as well as converting them to greyscale for some experiments. One of the modules is used to pre-process the training images before the training procedure, while the other one is used to pre-process the images before they are forwarded through the proposed system

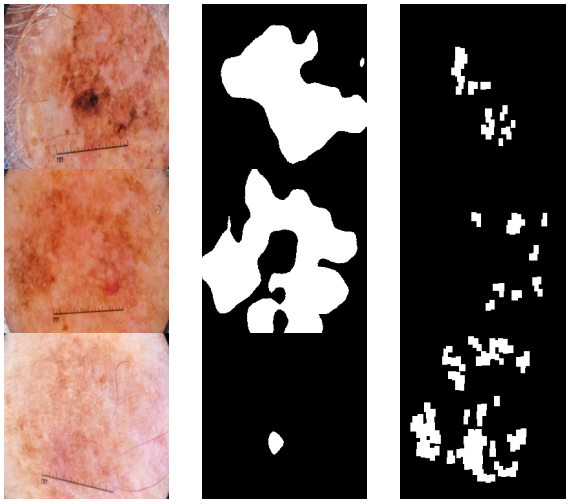


Figure 9: Poor segmentation results: original image (left), automatic segmentation (center) and groundtruth segmentation (right)

to obtain the corresponding segmentation of pigment network. The CNN architecture that was used is based on the U-Net, which was specifically designed for biomedical image segmentation tasks, in order to obtain automatically generated binary masks. The training of the model is performed from scratch, using the training images belonging to the ISIC dataset. During the training procedure, the dataset is divided into small mini-batches (2 images) as a matter of efficiency and memory limitations. Furthermore, a weighted loss function (weighted cross-entropy) was used to deal with the imbalanced data, which introduces a multiplicative weight for the minority class (pixels with pigment network). To obtain the best model configuration, the model parameters were validated every 10 epochs using the balanced error in the validation step and the configuration that achieved the best score was chosen.

The quantitative evaluation of the proposed system was performed by comparing each generated binary mask with its corresponding groundtruth mask, using as input the test images of the ISIC dataset. In general, it was found that using RGB images as the input of the proposed system and during the training procedure slightly improves the results. Regarding the loss function, it was found that using the WCE helps to deal with the imbalanced data, guaranteeing highly better results, mainly on the detection of foreground pixels.

The results obtained were quite promising, proving that deep learning methods can help medical experts to make faster and more accurate diagnoses. The proposed system achieved a $SE = 93.8\%$ and a $SP = 84.2\%$ by considering RGB images as the input of the system as well as the WCE as the loss function. These results prove that the developed system is a useful tool for the automatic detection of pigment network in dermoscopy images.

Despite the encouraging results obtained for the developed system, the proposed solution may benefit from the following suggestions:

- **Original depth of the channels**. Using the full depth of the channels at each stage of the network as proposed by the authors of U-Net [33] would result in a higher number of features learned by the network, which could improve the performance of the system. However, this should be combined with data augmentation, so that the model does not end up fitting the training data too well.
- **Data augmentation**. The performance of a deep CNN highly depends on the data that it has been trained with. The CNN architecture used has a very high number of parameters, which means that the model should be fed with a proportional amount of examples during training. For instance, by applying minor alterations to the existing data set, such as flips, translations or rotations, the model would be able detect foreground pixels even if they are placed in different orientations, which would make the model invariant to these alterations. This would also help to prevent overfitting, which is a common problem when the model is exposed to too few examples, learning patterns that do not generalize well to new data.
- **Transfer Learning**. Instead of training the entire CNN from scratch, it could be positive to try transfer learning by using pre-trained networks on very large datasets such as the Dermnet dataset [19] in order to initialize the weights safely.
- **Dropout**. This regularization technique could be used to reduce overfitting. At every iteration, it randomly selects some nodes, which are ignored during training. This means that their contribution is temporally removed. Thus, the other neurons have to handle the representation required to make the predictions for the missing nodes. The effect is that the network gets less sensitive to the specific weights of neurons, which makes the network capable of better generalization [39].
- **Detect other structures**. This CNN architecture could also be used to obtain models capable of detecting other dermoscopic structures, such as streaks, negative networks and milia-like cysts.
- **Distinguish between typical and atypical pigment network**. The presence of atypical network commonly results in the lesion being classified as a melanoma. Thus, it would be of great importance to distinguish between both types of pigment network, which would increase the value of the proposed system.

## REFERENCES


[1] "Dermoscopy tutorial," http://www.dermoscopy.org/atlas/base.htm, accessed: 2018-09-11.

[2] A. J. Miller and M. C. Mihm Jr, "Melanoma," *New England Journal of Medicine*, vol. 355, no. 1, pp. 51–65, 2006.

[3] A. W. Kopf, M. Elbaum, and N. Provost, "The use of dermoscopy and digital imaging in the diagnosis of cutaneous malignant melanoma," *Skin Research and Technology*, vol. 3, no. 1, pp. 1–7, 1997.

[4] C. M. Grin, A. W. Kopf, B. Welkovich, R. S. Bart, and M. J. Levenstein, "Accuracy in the clinical diagnosis of malignant melanoma," *Archives of dermatology*, vol. 126, no. 6, pp. 763–766, 1990.

[5] W. Stolz, "ABCD rule of dermatoscopy: a new practical method for early recognition of malignant melanoma," *Eur. J. Dermatol.*, vol. 4, pp. 521–527, 1994.

[6] G. Argenziano, G. Fabbrocini, P. Carli, V. De Giorgi, E. Sammarco, and M. Delfino, "Epiluminescence microscopy for the diagnosis of doubtful melanocytic skin lesions: comparison of the abcd rule of dermatoscopy and a new 7-point checklist based on pattern analysis," *Archives of dermatology*, vol. 134, no. 12, pp. 1563–1570, 1998.

[7] G. Argenziano, H. P. Soyer, S. Chimenti, R. Talamini, R. Corona, F. Sera, M. Binder, L. Cerroni, G. De Rosa, G. Ferrara *et al.*, "Dermoscopy of pigmented skin lesions: results of a consensus meeting via the internet," *Journal of the American Academy of Dermatology*, vol. 48, no. 5, pp. 679–693, 2003.

[8] M. G. Fleming, C. Steger, J. Zhang, J. Gao, A. B. Cognetta, C. R. Dyer *et al.*, "Techniques for a structural analysis of dermatoscopic imagery," *Computerized medical imaging and graphics*, vol. 22, no. 5, pp. 375–389, 1998.

[9] M. Anantha, R. H. Moss, and W. V. Stoecker, "Detection of pigment network in dermatoscopy images using texture analysis," *Computerized Medical Imaging and Graphics*, vol. 28, no. 5, pp. 225–234, 2004.

[10] G. Betta, G. Di Leo, G. Fabbrocini, A. Paolillo, and P. Sommella, "Dermoscopic image-analysis system: estimation of atypical pigment network and atypical vascular pattern," in *Medical Measurement and Applications, 2006. MeMea 2006. IEEE International Workshop on*. IEEE, 2006, pp. 63–67.

[11] M. Sadeghi, M. Razmara, T. K. Lee, and M. S. Atkins, "A novel method for detection of pigment network in dermoscopic images using graphs," *Computerized Medical Imaging and Graphics*, vol. 35, no. 2, pp. 137–143, 2011.

[12] P. Wighton, T. K. Lee, H. Lui, D. I. McLean, and M. S. Atkins, "Generalizing common tasks in automated skin lesion diagnosis," *IEEE Transactions on Information Technology inBiomedicine*, vol. 15, no. 4, p. 622, 2011.

[13] C. Barata, J. S. Marques, and J. Rozeira, "A system for the detection of pigment network in dermoscopy images using directional filters," *IEEE transactions on biomedical engineering*, vol. 59, no. 10, pp. 2744–2754, 2012.

[14] J. L. G. Arroyo and B. G. Zapirain, "Detection of pigment network in dermoscopy images using supervised machine learning and structural analysis," *Computers in biology and medicine*, vol. 44, pp. 144–157, 2014.

[15] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "A brief survey of deep reinforcement learning," *arXiv preprint arXiv:1708.05866*, 2017.

[16] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[17] A. C. F. Barata, E. M. Celebi, and J. Marques, "A survey of feature extraction in dermoscopy image analysis of skin cancer," *IEEE Journal of Biomedical and Health Informatics*, 2018.

[18] J. Kawahara and G. Hamarneh, "Fully convolutional neural networks to detect clinical dermoscopic features," *IEEE Journal of Biomedical and Health Informatics*, 2018.

[19] "Dermnet," http://www.dermnet.com/, accessed: 2018-10-10.

[20] "Isic archive," https://www.isic-archive.com/.

[21] N. M. Zaitoun and M. J. Aqel, "Survey on image segmentation techniques," *Procedia Computer Science*, vol. 65, pp. 797–806, 2015.

[22] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[26] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1520–1528.

[27] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2650–2658.

[28] W. Liu, A. Rabinovich, and A. C. Berg, "Parsenet: Looking wider to see better," *arXiv preprint arXiv:1506.04579*, 2015.

[29] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, "Conditional random fields as recurrent neural networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1529–1537.

[30] G. Ghiasi and C. C. Fowlkes, "Laplacian pyramid reconstruction and refinement for semantic segmentation," in *European Conference on Computer Vision*. Springer, 2016, pp. 519–534.

[31] G. Lin, C. Shen, A. Van Den Hengel, and I. Reid, "Exploring context with deep structured models for semantic segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 6, pp. 1352–1366, 2018.

[32] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille, "Attention to scale: Scale-aware semantic image segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3640–3649.

[33] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[34] R. C. Gonzalez, R. E. Woods *et al.*, "Digital Image Processing," 2002.

[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[36] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 437–478.

[37] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[38] N. C. Codella, D. Gutman, M. E. Celebi, B. Helba, M. A. Marchetti, S. W. Dusza, A. Kalloo, K. Liopyris, N. Mishra, H. Kittler *et al.*, "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)," in *Biomedical Imaging (ISBI 2018), 2018 IEEE 15th International Symposium on*. IEEE, 2018, pp. 168–172.

[39] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.