Combining deep Learning, handcrafted features, and metadata for the classification of dermoscopy images

by

Dimitrios Tselios

University of Piraeus

Department of Digital Systems

July 2022

Abstract

Malignant melanoma is the worst type of skin cancer and one of the world's fastest-growing malignancies. Early detection and identification of melanoma are critical for a high rate of full cure. Moreover, the importance of patient and dermatologist understanding of early melanoma signs and symptoms constitute a key factor for gaining useful experience concerning the disease. Towards these pillars, automated melanoma diagnosing systems that detect malignant skin lesions at early stages can support dermatologists to obtain objective quantitative markers, to relieve the excessive load of work and facilitate telemedicine capabilities. Deep learning-based approaches' efficiency has soared in recent years, and they now appear to outperform traditional machine learning methods in classification tasks.

In this master's thesis, Convolutional Neural Networks, handcrafted techniques, and metadata are used to extract features from a set of 58,457 dermoscopy skin lesion images. The extracted features from each technique, separately and in combination, are used to train machine learning classifiers towards the creation of a classifier that returns efficient results in terms of accuracy while, simultaneously, exploiting much simpler architectures than the state of the art. A curated ablation study assists in the determination of the base model components for the creation of the final architecture.

The proposed method was tested in the SIIM-ISIC 2020 melanoma classification, and it involves the use of a combination of EfficientNet-B0 features, GLCM, LBP, color moments features, and metadata features to improve model performance by 93,97% AUC score.

Acknowledgments

First of all, I would like to thank Prof. Ilias Maglogiannis of the Department of Digital Systems of the University of Piraeus for supervising and guiding my research. I would also like to thank Nasos Kalipollitis for the support and help it provided me throughout the duration of the thesis.

Moreover, I would also like to thank my family who supported me throughout my efforts by offering me all the necessary means to reach the desired result.

Finally, I would like to thank the University of Piraeus for giving me the opportunity to study in this master's program which was crucial for my personal and professional development

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Skin cancer is regarded by experts as a global health issue, being the most common form of all cancers. Especially, malignant melanoma, is the deadliest case and it causes the most skin cancer deaths. (Skin Cancer Facts & Statistics, 2022) From 2006 until 2016 it was noticed an increase of all type of cancers in both sexes. Melanoma is among the top 3 cancers with the most significant increase in incidence cases along with thyroid cancer and uterine cancer. Malignant melanoma has increased by 39%.The rates of melanoma have been rising rapidly over the past few decades, but this varies due to the shift in population's age structure, to the age-specific incidence rates change and to the population growth. (SEER Cancer Stat Facts: Melanoma of the Skin., 2019). More than 9,500 people are diagnosed with skin cancer in the United States every day. Every hour, more than two people die because of skin cancer. It is considered that in 2022 new melanoma diagnoses will decrease by 4.7%. However, the percentage of melanoma deaths is predicted to grow by 6.5%. Also, approximately 7,650 people are anticipated to die from melanoma where 5,080 are considered to be men and 2,570 women. (Cancer Facts & Figures 2022) In the United States, it is estimated that 197,700 cases of melanoma will be diagnosed in 2022 from which, 97,920 cases will be in situ (non-invasive), restricted to the epidermis. 99,780 cases will be invasive, passing through the epidermis into the dermis. From these cases in the period of thirty days of being biopsied 57,180 will be men and 42,600 be women with Stage I melanoma cancer (SEER Cancer Stat Facts: Melanoma of the Skin., 2019). Patients treated from 30 through 59 days after biopsy have a 5% higher risk while, patients treated later than 119 days after the biopsy have a higher risk of 41% to die. In the United States, the average five-year survival percentage of patients of all melanoma stages is 93%, while people with melanoma diagnosed in early stages is 99%. If the illness spreads to the lymph nodes, the life expectancy drops to 68%, and to 30% if it invades to distant organs. (Skin Cancer Facts & Statistics, 2022)

To limit melanoma cancer from expanding and causing irreversible results, early detection and treatment are of great significance. A broadly used non-invasive technique for skin tumors' diagnosis is dermoscopy. It has been fully established for diagnosing melanocytic tumors, e.g., melanoma, basal cell carcinoma (BCC), and squamous cell carcinoma (SCC) because it allows clinicians to identify them in the early stages (Dermoscopy of Melanoma and Non-melanoma Skin Cancers, 2019). However, a dermatologist can achieve a 65% to 75% accuracy rate with dermoscopy (Nami, Giannini, Burroni, Fimiani, & Rubegni, 2012). As a result, patients are often exposed to biopsies, which are emotionally and financially draining. A study by Elmore showed that by collecting 240 different skin biopsies varying from ordinary moles to advanced melanoma, pathologists disagreed that lesions were not that straightforward, worrying patients. (Elmore, et al., 2017)

Deep learning combined with traditional machine learning techniques gives new examples of how to create learning models for complex data. (O'Mahony, et al., 2020) Automated skin cancer detection is one of the many medical fields that can benefit from this, because it can solve issues related to the lack of medical staff in health care clinics, and in isolated areas, supporting telemedicine as well as the problem of misdiagnosis and recurrences of skin lesions. Using the automated skin lesion detection systems, doctors can receive an immediate answer by capturing a skin lesion image and submitting it through a predictive model, allowing them to diagnose melanoma without lab tests or additional fees even being away from their patients.

In this thesis, a novel approach is proposed that combines deep learning, handcrafted, and metadata features for melanoma classification. We experimented with various lesion segmentation techniques, deep learning, handcrafted and metadata feature extraction, and feature classification. We combined them to form an ablation study of all the possible combinations in an attempt to find the best result. The combination of deep learning, handcrafted features, and metadata improves the accuracy of the base models (deep learning features or learned features) for the binary problem of malignant or benign skin lesion classification problem. The generated results are comparable with the state-of-the-art techniques that employ much more complex neural networks.

1.2 Structure of Thesis

This thesis is divided to the following chapters:

Chapter 1: Introduction refers to the subject and underlines the scope and the aim of the present thesis.

Chapter 2: Background and Related work make a concise introduction to machine learning and take a more in depth look in deep learning methods and algorithms that are used for medical image processing. Moreover, it refers to the existing related work, the basic methods, and the problem of skin lesion classification.

Chapter 3: Methodology analyses in detail the pipeline of the proposed approach.

Chapter 4: Experimental Results contain dataset analysis, the evaluation methods were used, the results of the proposed methods, the performance improvement and the corresponding visualizations.

Chapter 5: Discussion about experimental results, the proposed methods and about the comparison of our method with other research solutions.

Chapter 6: Conclusion and Future Work is a summary of the completed work, and we discuss about the possible future opportunities.

Bibliography contains the list of sources that this research based on.

2 Background and Related Work

This section presents a concise introduction to machine learning and takes a more in-depth look at deep learning methods and algorithms used for medical image processing

2.1 Machine Learning

The earliest thinking and essential work in machine learning were done in the middle of the 20th century by Alan M. Turing. (Turing, 1950) Turing published a paper to consider if the machines could think. Turing proposes the imitation game, a test of machines' ability to interact with humans. More specifically, if an artificial system can interact with a human in natural language conversations. Artur Samuel firstly introduced the term machine learning in 1959 and used this term to describe a self-learning program (Samuel Checkers-playing Program)(Samuel, 1967). Tom M. Mitchell replaced Alan Turing's question and proposed the question if machines can do what we (as thinking entities) can do. He also provided a formal definition of the algorithms studied in the machine learning field: "A computer program is said to learn from experience E for some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." (Mitchell, 1997)

Machine Learning is a part of Artificial Intelligence, and it is the study of algorithms that can improve the use of data. (Mitchell, 1997) Machine learning algorithms build a training model using samples of data to make decisions or make predictions in unseen data.

Traditional programming pertains to any self-constructed program which uses data as input, and the computer program gives the output. During the input step, the program is fed with data by the programmer, who has to set the logic to produce the desired outcome manually. So, programmers have to manually set the rules of their programs via calculations and hardcoding. On the other hand, in machine learning programming, the input data are fed to an algorithm that produces an output. There is no need to program rules and set directions. Instead, the algorithms automatically formulate the logic from the data powerfully.

There are the following four subcategories of machine learning: Supervised Learning, Semi-Supervised Learning, Unsupervised Learning, and Reinforcement Learning.

2.1.1 Supervised Learning

Supervised learning is a subcategory of machine learning and Artificial Intelligence (AI). Supervised learning datasets contain the independent variables or features (X) and the dependent variables or target variables or labels (y). The main target is to produce a mapping function that can predict their outputs with new input data as it is shown in the Equation 1.

$$y = f(X) \qquad (1)$$

The process consists of data samples and an algorithm that learns from the training subset. The algorithm predicts answers and is corrected by a mechanism, loss function that takes into consideration the ground truth. Supervised learning can be further split into classification and regression tasks. In classification problems, the goal is to minimize the error between predicted discrete values and the correct label or the correct class of the data. If the number of labels is two, the problem is called a binary classification. When there are more labels the problem is called multilabel classification problem and when there are more classes, the problem is called a multiclass classification problem. Standard and widely used classification algorithms are Support Vector Machine (SVM), Naïve Bayes, Logistic Regression, Nearest Neighbor, Decision Trees, XGBoost, and Neural Networks.

In regression problems, we predict continuous dependent variables from independent variables. The goal of regression problems is to minimize the error between the predicted and ground truth (target) values. Some standard regression algorithms are Linear Regression, Support Vector Regression (SVR), and Lasso Regression.

2.1.2 Unsupervised Learning

In supervised learning, there are only the input data (X), and there are no output variables. The goal of unsupervised learning is the modeling of data structure or data distribution to take important information about the data. In contrast to supervised learning, there is no correct answer to the data. Algorithms are called to discover interesting patterns or structures, or groups of the unlabeled data. Unsupervised learning algorithms can be further separated into clustering problems, where algorithms cluster data into several groups by similarity, and association rules, which are called to discover rules or relationships that perform into data.

Cluster analysis or clustering is the grouping of data samples, where samples of the same group are more similar than other samples in different groups (clusters). The most common clustering algorithm is the k-means algorithm, which is a centroid-based algorithm. K-means uses the squared Euclidian distance as a similarity measure. The goal is to group the data into K clusters, working iteratively to assign each data point into a group. Other categories of cluster analysis algorithms are density-based, distribution-based, and hierarchical-based algorithms.

Association rules algorithms are trying to discover a possible relationship between the data. The target of the association rules is to associate or predict a relationship of some features with others. The most famous association rule algorithm is the Apriori.

2.1.3 Semi-Supervised Learning

Semi-supervised learning is a combination of the two previous types. In semi-supervised learning, a small sample of the data is labeled, and it is used to determine different classes of the data. Most of the data are unlabeled, and the goal is to find those labeled classes and discover possible additional ones. Semi-supervised learning takes the advangate of supervised and unsupervised learning, and in large-scale high-dimensional problems avoids the problem of collecting a large amount of labeled data. (Zhou & Belkin, 2014)

2.1.4 Reinforcement Learning

Reinforcement learning is a general term for an ensemble of techniques in which a learning system is trying to learn from its environment. The learning models are named agents, and these agents are trained by feedback required to learn their behavior. This problem is also known as a trial-and-error method that starts with totally random trials, and through the actions and feedback, the model finds a solution for a specific problem. The agent decides the best move based on its current situation minimizing the risk function.

2.1.5 Classification Algorithms

In the following section, we will present the essential supervised machine learning algorithms:

2.1.5.1 Artificial Neural Networks

An artificial neural network (ANN) is an information processing model inspired by the biological nervous system of brain. Artificial Neural Networks contain an input layer, one or more hidden layers, and an output layer. The basic type of an ANN is the feedforward neural network, which consists of interconnected neurons, and each neuron receives an ensemble of weighted input. The process follows the summarization of those weighted inputs and, finally, the output generated using an activation function. Figure 1 presents the structure of an ANN where features come either as input from a dataset or another output layer. (Dongare, Kharde, & Kachare, 2008) (Society, 2013).

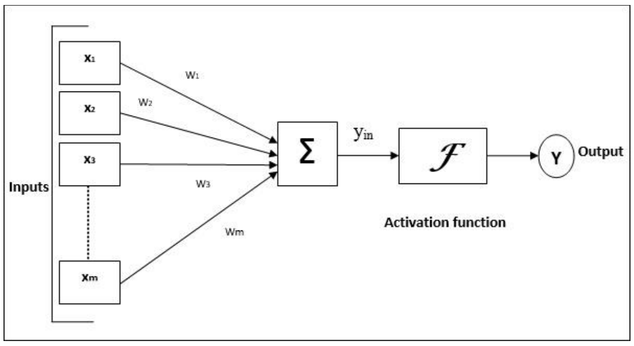


Figure 1: Basic Artificial Neural Network

Figure 1 represents a basic artificial neuron, which contains multiple inputs $[X_1, X_2, X_3, \dots, X_m]$. The neuron input can be calculated as shown in Equations (2) and (3):

$$y \ = \ X_1W_1 + X_2W_2 + X_3W_3 + \cdots + X_mW_m \quad (2)$$

$$y = \sum_{i}^{m} X_i \, W_i \quad (3)$$

Where m is the number of examples and W are the weights.

The activation function or Transfer function computes the relationship between input and output and is divided into two categories:

1. The linear function where the output corresponds to the total weighted output. The linear function does not help with complex data that are usually feed the artificial neural networks. (Myers, 1989)
2. The non-linear functions are the most of activation functions and it is easier to model various data. The non-linear activation functions can be typically divided by their range or curves. The most famous activation functions are Sigmoid, Tanh, Rectified Linear Unit (ReLU), and Leaky ReLU. (Sibi, 2013) (Sharma, 2017) (Agarap, 2018)

A Neural Network consists of multiple connected neurons by layers. In a neural network, as it is mentioned above, there are three-layer types, the input layers are the layers that receive the unweighted input data, the hidden layers and the output layers that receive the output of hidden layers and produce the neuron's output. Figure 2 show the architecture of a neural network with an input layer, a hidden layer and an output layer.

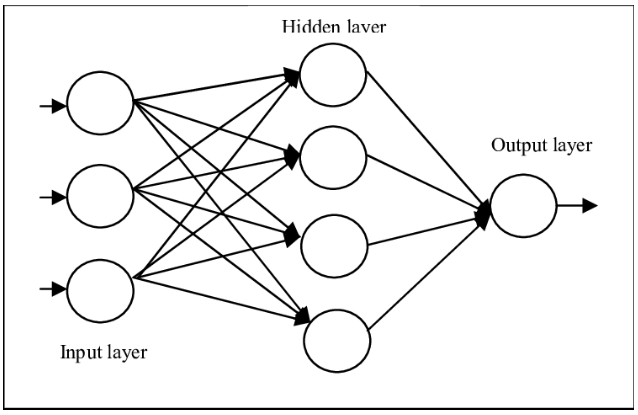


Figure 2: Representation of a neural network (Kataev, et al., 2012)

The neural network learns iteratively for N training cycles (epochs) in the training stage, and it uses a loss function to calculate the prediction error after each epoch in an attempt to minimize it. In other words, the loss function quantifies the difference between the outcome predicted by the machine learning model and the expected outcome. We receive the gradients used to update the training weights from the loss function. The simplest loss function is described in the Equation 5:

$$loss = |truevalue - predicted\ value| \qquad (5)$$

Whereas the most used loss function in regression problems is the mean squared error (MSE) and the mathematical Equation is described in the Equation 6:

$$E = (y_i - \hat{y}_i)^2 \qquad (6)$$

where it is just the squared difference between the expected value (y_i) and the predicted value $(\hat{y}_i)$. The cost function results from the average of all losses outcomes is described in Equation 7:

$$C = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \qquad (7)$$

A Loss functions for classification tasks is the Cross Entropy, which is described by the Equation 8:

$$CrossEntropyLoss = -(y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)) \quad (8)$$

where $\hat{y}_i$ is the predicted value and y_i is the expected value. Other loss functions for classification tasks are Categorical Cross Entropy, Hinge Loss, Cosine Similarity. (Chai, 2014) (Ho, 2019) (Zhang, 2018) (Janocha, 2017) (Xia P. Z., 2015)

2.1.5.2 Random Forest

Decision Tree (DT) is a well-known and influential supervised learning algorithm, which has affected a large range of machine learning, involving both classification and regression problems. (Rokach, 2015) The basic structure of a decision tree contains nodes that represent a feature, links (branches), which represent a rule (decision), and leaf which represents the outcomes. So, a decision tree is a prediction model that represents decision-making rules, using input features to predict a target value accurately. In classification decision trees, target variables are discrete values, and in regression decision trees, target variables are continuous. There are some metrics that define the cost of each split. Two of them are the Gini impurity and the Information Gain. The Gini impurity metric is used in Classification and Regression Tree (CART), and calculates the probability of new, random data being incorrectly classified if it were given a random class label. In case we have C classes and p_i is the probability of picking a data point with class i, the Gini Impurity is described in Equation 9:

$$\mathrm{G} = \sum_{i=1}^{C} p_i * (1 - p_i) \quad (9)$$

The split which produces minimum Gini impurity will be selected.

The Information Gain metric calculates how much entropy we removed. Entropy is defined as in Equation 10:

$$E(S) = -1p_1 logp_1 - 2p_2 logp_2 - \cdots - np_n logp_n = -\sum p_i \, logp_i \quad (10)$$

The Information Gain is defined in Equation 11:

$$IG(T, a) = E(T) - E(T|a) \quad (11)$$

The parent entropy E(T) minus the children's weighted Entropy for each split according to a characteristic E(T|a). The feature that maximizes the difference is the one in which the separation will take place.

Random Forest Classifier contains multiple decision trees as an ensemble. Each decision tree feeds a class prediction, and the class with the most votes will become the model prediction class.

2.1.5.4 XGBoost

Extreme Gradient Boosting (XGBoost) is a scalable distributed gradient boosting decision tree algorithm. XGBoost uses parallel tree boosting which has been proven to provide robustness for classification and regression problems. (Chen & Guestrin, 2016)

A Gradient Boosting Decision Tree (GBDT) is an ensemble machine learning algorithm contiguous to Random Forest. Ensemble learning algorithms employ numerous algorithms to produce a powerful model. Both Random Forest and GBDT are based on decision tree models, but Random Forest uses a bagging technique where the final prediction is based on the majority class. On the other hand, the Gradient Boosting algorithm uses several weak models to improve a single weak model to produce a robust model, to produce a robust model.

Gradient Boosting is an extension of boosting algorithm where the process of producing weak models is based on Gradient Descent. GBDT is an iterative process to train an ensemble of decision trees. Each iteration uses the error of residuals of the previous model to fit the next one. The final prediction model results from a weighted sum of all tree predictions. (Chen & Guestrin, 2016)

XGBoost is an advanced implementation of Gradient Boosting Decision Trees and some regularization factors. (Chen & Guestrin, 2016) XGBoost is the most efficient scalable tree boosting method with applications in different cases such as motion detection, customer behavior analysis, stock sales prediction, and image recognition.

2.2 Deep learning

Deep learning is a subcategory of machine learning and artificial intelligence. In deep learning algorithms, there are artificial neural networks with many hidden layers inspired by human brains' structure, and they learn using data, weights, and biases. The 'deep' refers to the two or more hidden layers. Deep learning algorithms are applied to computer vision, object detection, speech recognition, and bioinformatics.

Deep learning techniques are prevalent because of their ability to process unstructured data as input, such as text or images, extract features automatically, and predict the outputs. An essential characteristic of Deep Neural Networks (DNN) is the so-called feature hierarchy. Each hidden level of neurons trains on a specific type of feature, such as the hands, if we use position and hand-detecting images. The deeper a level is, the more complex the characteristics on which it is trained are. For example, suppose we want to classify some animal photos like cats, dogs, and lions. In that case, deep learning algorithms can detect low level important features, like edges, nodes, then in a higher level it uses those features to detect eyes, noses, mouths, ears and in the highest level it combines these features to semantically important representations for making predictions. This situation is called forward propagation. Moreover, through the back propagation process and gradient descent algorithm, it calculates the prediction errors to update weights and bias, increasing the accuracy of predictions.

However, there are more complicated deep learning models with different structures, and they are specified to solve specific problems. Two of them are the Convolutional Neural Networks (CNN) and the Recurrent Neural Network (RNN). The first are used primarily on computer vision and image classification problems. They can extract features and patterns from images. The second type, which is a sequential model, it is used mainly in speech recognition, time series analysis, and generally in sequential data points.

2.2.1 Convolutional Neural Networks (CNN)

Convolutional neural networks belong to the category of deep neural networks and were first used successfully in the 1998 by LeCun. (LeCun, 1998) This type of neural network is advantageous and is widely used for various functions related to computer vision such as object detection and image classification. In these cases, convolutional neural networks are more popular than classical artificial neural networks. A convolutional neural network is trained by taking images as input and then having three types of layers, convolutional layers, pooling layers and fully connected layers like the one in the Figure 3, where we want to predict whether an image belongs to a particular category or not.

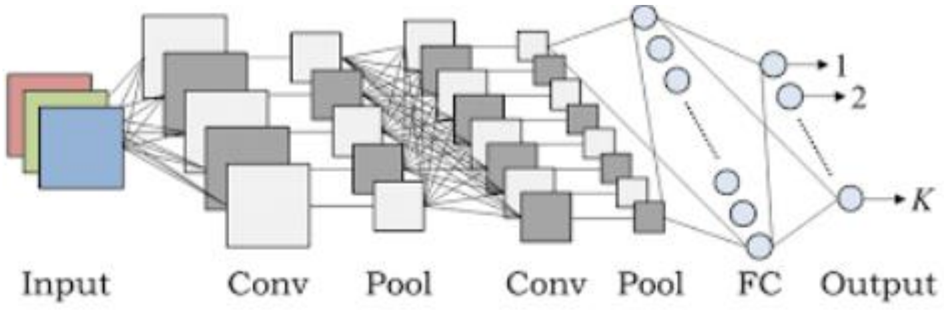


Figure 3: A convolution neural network architecture (Hidaka, 2017)

Each level of CNN usually contains two separate sections. The first is the portion of the feature map present in each CNN and results from the convolution of the previous level with a weighted kernel. The input image relates to the weights of the feature map W_k. The result then goes through a non-linear activation function, generating the final output y_k as it is shown in the Equation 12 where f is the non-linear function, the W_k are the weights over the input x .

$$y_k = f(W_k * x) \quad (12)$$

One of the most popular activation functions is ReLU, which allows the extraction of non-linear features from the convolutional layers. Mathematically it is expressed in the Equation 13 using the max() function over the set of 0 and the input x. The ReLU is linear for values greater than zero, meaning it has a lot of the desirable properties of a linear activation function when training a neural network using backpropagation. Also, it is a nonlinear function as negative values are always output as zero.

$$f(x) = \max(x, 0) \quad (13)$$

Pooling layers take the activation layers' outputs and decrease the dimensionality of the features by combining related features into a single output. Average pooling is a typical method where input values from a small image segment are averaged and specified as the input for the following level. However, a second most used technique, the max pooling, is in which the maximum value of the input set is fed to the next level. In Figure 4 it is described precisely the difference between the max pooling and the average pooling technique.

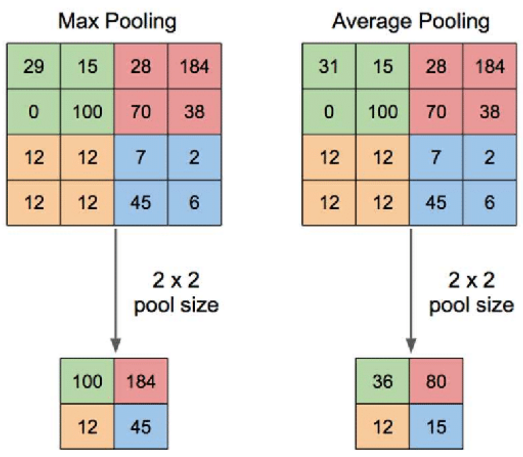


Figure 4: Illustration of Max Pooling and Average Pooling (Yani, 2019)

Convolutional and pooling layers are stacked several times to enhance feature recognition. Next, a set of fully connected neurons, called Fully Connected Layers, which are connected to all feature maps of the previous layer, are accountable for generating high-level inferences. During this stage, the neurons define the detected features and classify them.

A normalization or loss layer is inserted to calculate the expected and predicted values difference. Commonly used functions are the softmax, which calculates the success of a class among all exclusive classes, the sigmoid function, which calculates K independent probabilities K of different classes in a range [0,1], and the Euclidean loss, which is used for the regression of the actual value labels. The CNN training is like the ANN training and uses the backpropagation algorithm.

Pre-processing Layers

Before a series of input images are imported to the input layer, multiple preprocessing steps are needed, including resizing and normalization processes. However, CNNs require much fewer preprocessing steps than other neural networks. Otherwise, a simple pre-processing layer is required to eliminate insignificant differences.

Convolutional Layers

A convolutional layer compresses the input by collecting important features from it and producing feature maps to the different feature detectors. In the first convolutional layer, the neurons filter simple features such as edges. While each convolutional kernel can extract features throughout the input layer, the neurons are responsible for different parts of the input layer to produce the feature maps of the same size as the receptive field. This method of sharing convolutional weights is derived from Convolutional Neural Networks and refers to a stack of feature maps. The depth of the stack is determined by the number of neuron types. Each neuron type shares the same weight and bias vector and generates a stack slice. The input size, the kernel size, the depth of the map stack, the zero-padding, and the stride are parameters that determine each convolutional layer.

Activation

The neural networks estimate continuous functions in Euclidean space, using a non-linear activation function. LeCun introduced a sigmoid function to suppress the output of a pooling layer. Later, Jarrett et al. proposed Rectified Linear Units (ReLUs) into CNNs to boost their efficiency. Shortly after, Xavier Glorot et al. underlined that the exceptional efficiency of ReLUs is due to their complex non-linearity, non-differentiability at zero, and sparse features. Finally, ReLUs are generally accepted for the activation of convolutional outputs. ReLU is given by the Equation 14:

$$F(x) = max(0, x) \quad (14)$$

Pooling Layers

Pooling includes various operations such as general pooling, overlapping pooling, etc. It generally acts as a bridge between many convolutional layers. The common pooling methods in CNNs are max-pooling and average-pooling. CNNs benefit greatly from pooling. Pooling seeks to reduce overfitting and computations by focusing local data via a pooling window, hence lowering data dimensionality. In addition, pooling causes invariance in translation, rotation, and scaling, since a few offsets or scaling no longer make a difference after pooling.

Classification Layers

The top layer of a network is the classification layer, which gathers the final convolved feature and outputs a column vector. Each row refers to a class. More technically, the output vector's components each reflect a probability prediction for each class. While convolutions combined with pooling map the raw image data into a feature space, the classification layer influences the projection of the pattern space, providing an obvious exhibition of classification. Most often, CNNs result in a fully connected layer. The fully connected classification layer is a legacy of the idea of 'feature extraction and classification' in Artificial Intelligence. Fully connected inputs are performed to combine and reweight all higher-order features to achieve the spatial transformation.

Dropout and Batch Normalization

During training, regularization techniques like dropout and batch normalization can be used to assist the model in adjusting to new, previously unseen samples and preventing overfitting. (Chang, 2015) (Salakhutdinov, 2014) During the training phase, dropout refers to a regularization approach that zeros out the activation levels of randomly chosen neurons. Not depending on prior units, this stops units from co-adapting and drives them to learn more robust features. Consequently, the network becomes less sensitive to individual factors and has a more vital ability to generalize. Batch normalization includes removing the batch mean from each activation and dividing the batch standard deviation. The neural networks' stability and performance is improving using this method.

Fine Tuning and Transfer Learning

Deep learning can handle a wide range of problems, but it needs a lot of computational power and data. Compared to other fields, data availability in medical imaging is frequently limited. Because of the domain's sensitivity, annotation necessitates several expert judgments on the same data, both costly and time-intensive. Although several deep convolutional neural network approaches have produced significant results in medical imaging, training deep models with a small set of labeled data remains challenging. As a result, overfitting, which appears when a function is strongly fitted to a limited amount of data points, is usually a concern in training deep models with limited examples due to the big number of parameters. Many transfer learning-based approaches in medical imaging classification tasks have been developed to address these constraints.

In transfer learning, a pre-trained model for a task is reused as the starting point for a new task. Transfer learning aims to transfer information gathered from many labeled data to new situations. (S. J. Pan and Q. Yang, 2010) People have natural means of transferring knowledge across tasks by applying it from prior learning experiences to new ones. Many publicly accessible deep models have been pre-trained on the ImageNet dataset (Deng, 2009), containing over 1.2 million pictures. They have produced state-of-the-art results in a variety of medical imaging classification issues. The use of a pre-trained model for feature extraction is one of the most common transfer learning strategies. The pre-trained network acts as a feature extractor, feeding a new classifier, the output layer. Fine-tuning is another transfer learning strategy in which the pre-trained model's weights serve as an initialization scheme for the new task (Shin, 2016). When we apply fine-tuning in a pre-trained network with a new dataset, there are several options, like training the entire initialized network or freezing part of the parameters and training the remainder, generally by updating only the network's final layers. The usage of pre-trained models is limitless, and several studies have been undertaken to take advantage of the benefits of transfer learning in medical imaging.

2.2.2 Deep learning features

3.4.1.1VGG19 features

VGG19 is a variant of the VGG model, which includes 19 layers. More in depth, there are 16 convolution layers, 3 fully connected layers, 5 max Pooling layers, and 1 softmax layer. (Shaha, 2018) The input size of the network is fixed, and it receives RGB images of size 224x224, meaning that it receives a matrix of shape (224,224,3). The pre-processing process deducts the mean RGB value from each pixel and calculates it over the training set. It utilizes kernels of size 3x3 with a stride size of 1 pixel covering the image as a whole. Also, spatial padding maintains the spatial resolution of the image. The max-pooling executes a 2x2 pixel windows with side 2. Moreover, introduces non-linearity to classify the model better and improve computational time using Rectified linear unit (ReLu). Finally, it implements three fully connected layers. The first two fully connected layers have size of 4096. The last fully connected layer has size of 1000, and the final layer is a softmax function. The Figure 5 represents the VGG19 architecture with the convolutional, the max-pooling, the fully connected and the softmax layer.

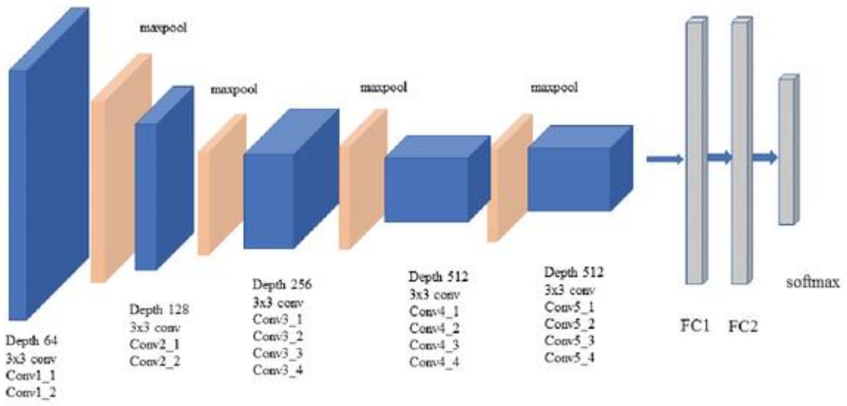


Figure 5: Vgg19 model architecture

When loading the model, the final dense layers were excluded. The feature extraction phase of the model is from the input layer to the last max-pooling layer (designated by 7 x 7 x 512). After defining the model, we loaded the input resized images of size 224x224. We have access to all of the model's layers, and each layer has a layer.name property. Each convolutional layer has two sets of weights. The first is the filters block, and the second is the bias values block. The layer.get weights() method provides access to them. These weights can be retrieved, and their form can then be summarized in Figure 6.

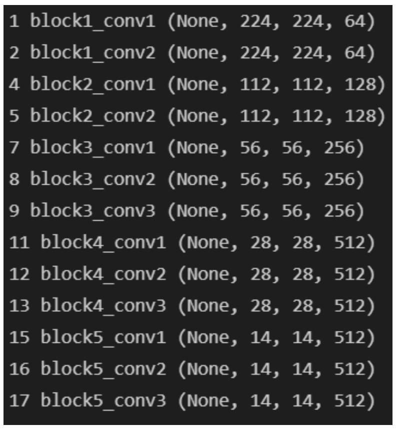

```
1 block1_conv1 (None, 224, 224, 64)
2 block1_conv2 (None, 224, 224, 64)
4 block2_conv1 (None, 112, 112, 128)
5 block2_conv2 (None, 112, 112, 128)
7 block3_conv1 (None, 56, 56, 256)
8 block3_conv2 (None, 56, 56, 256)
9 block3_conv3 (None, 56, 56, 256)
11 block4_conv1 (None, 28, 28, 512)
12 block4_conv2 (None, 28, 28, 512)
13 block4_conv3 (None, 28, 28, 512)
15 block5_conv1 (None, 14, 14, 512)
16 block5_conv2 (None, 14, 14, 512)
17 block5_conv3 (None, 14, 14, 512)
```

Figure 6: VGG19 convolutional blocks

All convolutional layers make use of 3x3 filters. The depth of a filter in a convolutional neural network must match the depth of the input for the filter to operate (e.g., the number of channels). The Figure 7 shows the first six filters from the VGG19 model's first hidden convolutional layer for the three channels. Each filter is plotted as a new row of subplots, and each filter channel or depth is plotted as a new column.

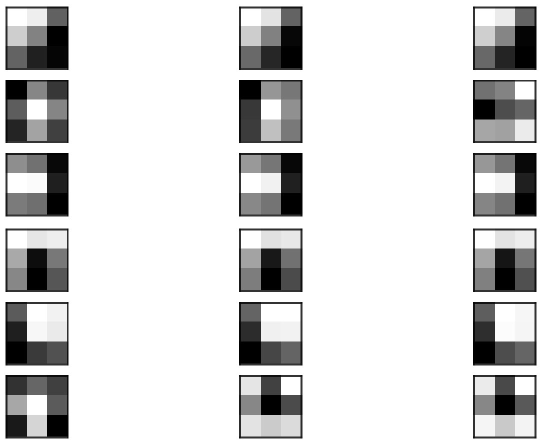


Figure 7: First 6 Filters out of 64 Filters in first hidden layer of VGG19 Model

It can be observed that in certain situations (the first row), the filter is the same across all channels, whereas, in others, the filters are different (the last row). Small or inhibitory weights are shown by dark squares, whereas bright squares indicate high weights. Also, the first row of filters detects a gradient from light in the top left squares to dark squares in the bottom right.

3.4.1.2 DenseNet 121

DenseNets, or densely connected convolutional networks, enhances the depth of deep convolutional networks. DenseNets reuse features instead of gaining power from highly deep designs and they require fewer parameters than regular CNNs since no duplicate feature maps must be learned. Traditional feedforward neural networks connect the output of the layer to the next level after applying a combination of functions. Typically, convolution operations or pooling layers, batch normalization, and an activation function are three components of the network.

The formula for this is the Equation 15:

$$x_l = H_l\left(x_{(l-1)}\right) \quad (15)$$

DenseNets, instead of adding the level output feature maps to the incoming feature maps, it merges them. Thus, the Equation 16 becomes:

$$x_l = \ H_l([x_0, x_1, x_2, \dots, x_{l-1}) \quad (16)$$

The feature map dimensions remain constant inside a block, but the number of filters varies. Transition Layers are the layers between them that handle down-sampling using batch normalization, 1x1 convolution, and 2x2 pooling layers. Because we concatenate feature maps, the channel dimension grows with each layer. We can generalize for the n-th layer if we force H l to create k feature maps every time using the Equation 17:

$$k_l = k_0 + k * (l - 1) \quad (17)$$

The hyperparameter k represents the growth rate. The growth rate handles the amount of data that will be added by each layer to the network. Each level has prior accessibility to the feature maps and therefore, to the collective knowledge. The Figure 8 represents the DenseNet121 model architecture.

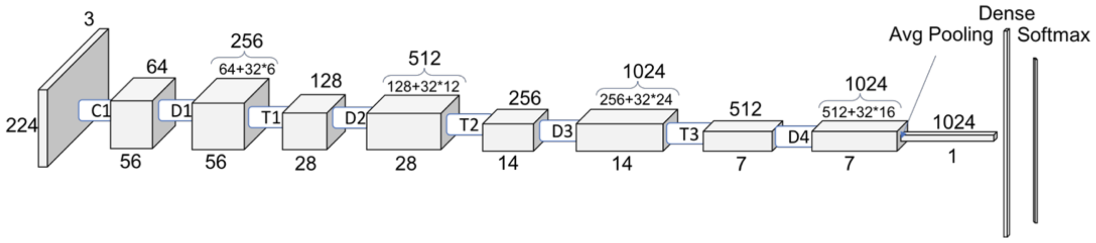


Figure 8: DenseNet121 model architecture

It is worth noting how the initial number of the addition used to generate the feature maps of each new volume corresponds to the preceding volume's feature maps dimension. This explains the bypass connection because it indicates that we are adding new information to the prior volume, which is being reused (concatenate means increasing dimension, not adding values).

Moving 2 levels deeper the functions inside every block can be observed.

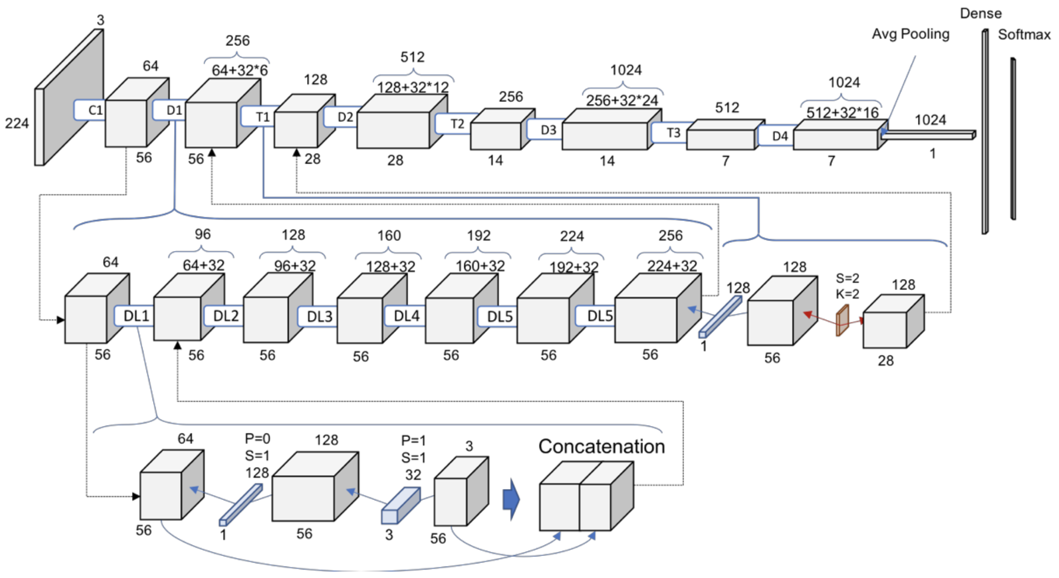


Figure 9: DenseNet121 architecture in 2 deeper levels

As it is shown in Figure 9, each layer adds 32 new feature maps to the preceding volume. This is why, after six levels, we get from 64 to 256. Transition Block also functions as a 1x1 convolution with 128 filters. A 2x2 pooling with a stride of 2 was performed, resulting in the volume and number of feature maps being divided in half. This pattern observation can be used to generate additional assertions. This behavior of adding 32 times the number of layers is achieved at the new deeper level representing the first Dense Layer within the first Dense Block. A 1x1 convolution is applied with 128 filters to reduce the size of the feature maps. Then, a 3x3 convolution uses the 32 feature maps of growth rate. In addition, to add new information to the network's existing knowledge, the input volume and the results of the two processes are merged. (Huang, Liu, van der Maaten, & Weinberger, 2016)

3.4.1.3 EfficientNet

EfficientNet is a scaling CNN architecture that uses a compound coefficient to equivalently scale its dimensions of depth, width, and resolution. (Tan & Le, 2019) The EfficientNet scaling approach, unlike standard practice, consistently scales network width, depth, and resolution with a set of predetermined scaling coefficients. To utilize 2^{ν} additional processing resources, the network depth is expanded by α^{ν}, the width by β^{ν}, and the image size by γ^{ν}. Where, α, β,γ are constant coefficients obtained by a small grid search, the little initial model. EfficientNet adjusts network width, depth, and resolution consistently and rationally using a compound coefficient.

Compound Scaling

Convolutional Neural Networks (CNNs) are typically built with a range of capabilities and then grown-up for higher accuracy as additional resources become available. The modification of 3 main dimensions: depth, width, and resolution, is called scaling a convolutional neural network. The width is representing the filter number in each convolutional layer, whereas the depth is representing the number of convolutional layers. Finally, the resolution is just the supplied image's dimensions.

Scaling up the depth of the network by adding additional convolutional layers helps the network to acquire complicated features. However, the deeper the network, it is suffering more from diminishing gradients and it is harder to train. While batch normalization and skip connections or inter node connections at various levels help solve this problem, true accuracy benefits from rapid network depth saturation. Increasing the width of the networks, on the other hand, allows the layers to learn more fine-grained information. However, just like depth scaling, escalating the width avoids the network to learn complex features, which is leading to better accuracy. Finally, a higher input resolution gives the network more information about the picture and helps it extract finer patterns but only yields small accuracy increases.

Scaling up the model in a combination of width, depth, and resolution improves its prediction ability. This is because as the spatial resolution of an input picture increases, the number of convolutional layers should rise as well, allowing the receptive field to encompass the full image with more pixels. Increasing a network's dimensions arbitrarily does not guarantee better outcomes and balancing all three dimensions is challenging. In reality, to meet resource limits, the method frequently needs several tries to scale up the dimensions correctly. The Figure 10 represents the number of FLOPS of EfficientNets on the ImageNet dataset.

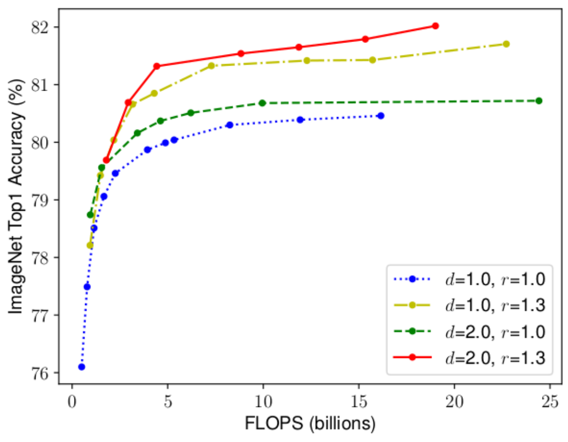


Figure 10: Number of FLOPs of EfficientNets on ImageNet dataset

Scaling network width for different baseline networks means that combining three dimensions should help network scaling for increased accuracy. This is supported by empirical data in the figure above, which shows the accuracy of the networks as a function of increasing width for varied depth and resolution settings. The results show that scaling only one dimension (width) causes accuracy improvement to stop soon. When this is combined with an increase in the number of layers (depth) or input resolution, the model's predictive performance improves.

The idea behind EfficientNet is, to begin with, a high-quality model and scale all three dimensions uniformly using a compound coefficient according to Equation 18:

$$depth = \alpha^{\varphi}, width = \beta^{\varphi}, resolution = \gamma^{\varphi}$$
$$s.t\ \alpha * \beta^2 * \gamma^2 \approx 2, \qquad (18)$$
$$\alpha \leq 1, \beta \leq 1, \gamma \leq 1$$

where the values of α, β, γ are determined using a grid search technique. The ϕ is a user-defined parameter that determines the network's increased computational power. The component that indicates that $\alpha * \beta^2 * \gamma^2$ is close to 2 binds them. The Floating Point Operations Per Second or FLOPS of a convolution process are proportional to $d * w^2 * r$, so that when the network depth is doubled, the FLOPS will double as well. Even yet, doubling network width or resolution will result in a fourfold increase in FLOPS. The computational expenses are controlled by CNN's convolutional processes. The FLOPS will rise by $(\alpha * \beta^2 * \gamma^2)^{\varphi}$ if the network is scaled using the preceding equation. For a new ϕ since $\alpha * \beta^2 * \gamma^2 \approx 2$, the total FLOPs will increase by 2^{φ}. Where ϕ regulates the number of available resources to scale the model, and α,β,γ are grid-searched constants that suggest the network's dimension distributed resources. (Karki, Kulkarni, & Stranieri, 2021) (Tan & Le, 2019)

The Figure 11 represents how the structure of a model changes by scaling width, depth, resolution separately and with compound scaling.

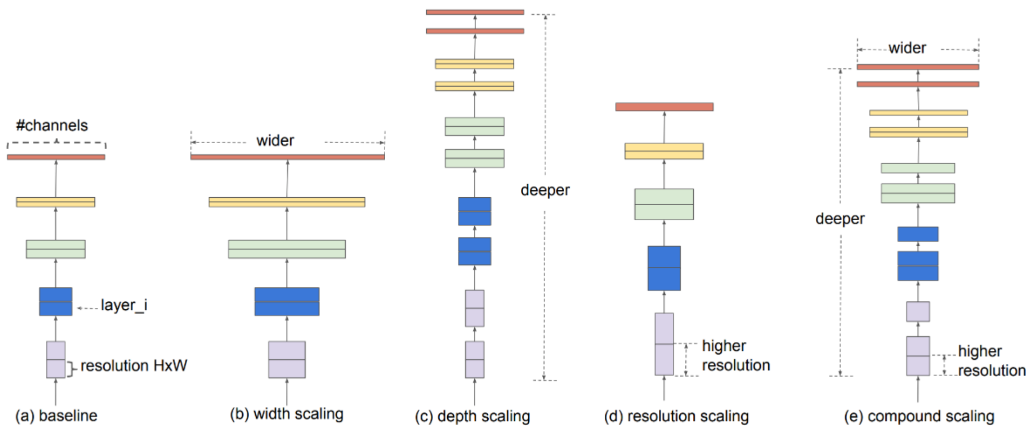
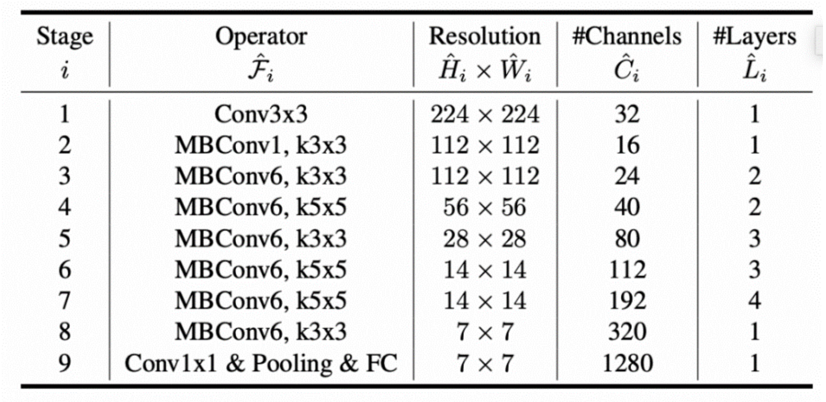


Figure 11: Model Scaling, (a) is a baseline network example, (b)-(d) are convolutional scaling thet only increases one dimention of network width, depth or resolution, (e) is the compound scaling method that uniformly scales all three dimentions with fixed ratio

EfficientNet-B0

When ϕ is set to 1 and twice as many resources are available, a, b, c are the results of a tiny grid search. For f=1, the values $\alpha = 1.2, \beta = 1.1, \gamma = 1.15$ of maintaining the $\alpha * \beta^2 * \gamma^2 \approx 2$. This is the EfficientNet-B0 model. (Sandler, Howard, Zhu, Zhmoginov, & Chen, 2018) Unlike classical convolution, which utilizes a 2-D depth filter to directly convolve the input in-depth, depth-wise separable convolution only applies each filter channel to one input channel. It separates the filter and image into three channels and applies the appropriate filter to each channel. It then applies a pointwise convolution on the output to combine it. (Ali, Shaikh, Khan, & Laghari, 2022) The Figure 12 represents the EfficientNet-B0 model architecture.



Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

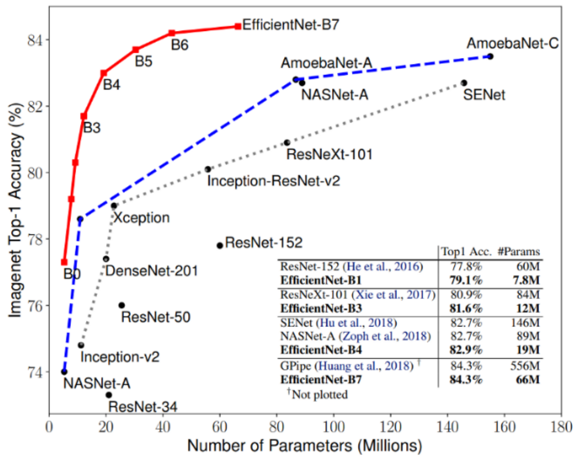
Figure 12: EfficientNet-B0 model architecture

Depthwise convolutions are usually combined with another step called Depthwise Separable Convolution. This is divided into two sections: 1. filtering (all previous processes) and 2. combining (combining the three-color channels to generate an arbitrary number of channels).

EfficientNetB1-B7

The EfficientNets B1-B7 was created using the compound scaling approach. A short grid search was used to identify the α, β, γ, by beginning with EfficientNet-B0 as the baseline model and setting ϕ = 1, assuming twice as many resources are available. When these parameters are specified, they determine the compound coefficient ϕ that increases to scale the base model and generate EfficientNet-B1 -B7. The number at the end of the names indicates the value of the compound coefficient value.

The Figure 13 represents the performance of the EfficientNets in comparison to other CNN architectures on the ImageNet and CIFAR-100 datasets. The largest EfficientNet model, EfficientNet B7, achieved state-of-the-art performance. On ImageNet, it achieved an accuracy of 84.4 percent top-1 and 97.3 percent top-5. Moreover, the EfficientNet model was 8.4 times smaller and 6.1 times quicker than the previous best model.



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

†Not plotted

Figure 13: EfficientNet size and performance on the ImageNet dataset in comparison with other CNN architectures

2.3 Image Segmentation

This section describes all segmentation methods and experiments applied to conclude the proposed segmentation method.

Chan-Vesse Active Contours

The Chan-Vese model is a powerful method, able to produce efficient results where simpler, less sophisticated segmentation methods fail (threshold or gradient-based methods). (Chan & Vese, 2001) The Chan-Vese method optimally adapts a two-phase piecewise-constant model to an image. The segmentation border is indirectly represented by a level set function and allows segmentation to easily handle topological changes. (Getreuer, 2012) The Chan-Vese model is based on the Mumford-Shah function to segment and it is widely used in medical imaging, especially for segmentation of the brain and heart (D. Mumford & J. Shah, 1989) Chan-Vese model uses an additional term that penalizes the fenced area and an additional simplification where has only 2 values as it is described in the Equation 19,

$$u(x) = \begin{cases} c1, where\ x\ is\ inside\ C, \\ c2, where\ x\ is\ outside\ C, \end{cases} \qquad (19)$$

where C is the boundary of a closed set and c1, c2 are the values of u, inside and outside C.

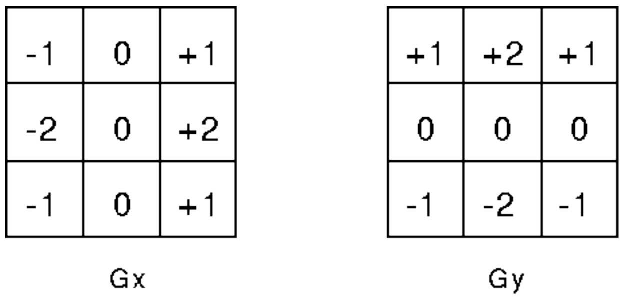
The Chan-Vese technique (Getreuer, 2012) aims to identify the u of this type that best predicts f.

$$arg_{u,C} min \mu Length(C) + \nu(Area(inside(C))) + \lambda 1 \int_{inside(C)} |f(x) - c1|^2 dx + \lambda 2 \int_{outside(C)} |f(x) - c2|^2 dx + \lambda 2 \quad (20)$$

The Chan-Vesse active contour segmentation algorithm is applied to the whole train images to generate masks for each example to detect the Region of Interest (ROI), using the scikit-image library (Van der Walt S. S.-I., 2014) with number of iterations equal to 100, threshold equal to 0,79, and smoothing level equal to 1.

Sobel Operator

The Sobel Operator identifies edges by detecting sudden variations in pixel intensity. A 2-D spatial gradient measurement on a given picture is executed by the Sobel operator, where it emphasizes areas that correspond to edges. The Sobel operator generally operates to detect the estimated absolute gradient magnitude at each point in a given grayscale image. The Sobel operator contains a pair of 3×3 convolution kernels. As it is shown in Figure 14, the one kernel is the second rotated by 90°.



-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

Figure 14: Sobel convolution kernels, (Sobel)

The kernels correspond to edges running vertically and horizontally. The kernels can produce separate calculations of the gradient component by using an input image in horizontal and vertical orientation. To identify the absolute gradient magnitude and the gradient's orientation we can combine them. The gradient magnitude is given by the Equation 21:

$$|G| = \sqrt{{G_x}^2 + {G_y}^2} \qquad (21)$$

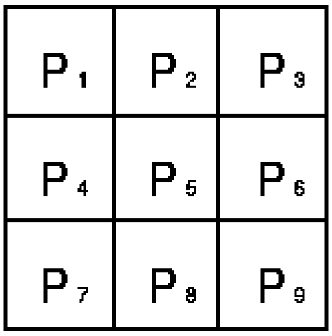
An approximate magnitude is computed by using the Equation 22,

$$|G| = |G_x| + |G_y| \quad (22)$$

The angle of orientation of the edge (relative to the pixel grid) is giving rise to the spatial gradient by the Equation 23:

$$\theta = \arctan(G_x/G_y) \quad (23)$$

In this case, orientation 0 determines that it runs in clockwise direction on the image meaning that the maximum contrast runs from black to white, while other angles differentiate and run to the opposite direction. The Figure 15 represents the two elements of the gradient are easily calculated and added with a single pass over the input image using the pseudo-convolution operator.



P_1	P_2	P_3
P_4	P_5	P_6
P_7	P_8	P_9

Figure 15: Sobel pseudo-convolution detector

Using this kernel, the approximate magnitude is given by the Equation 24:

$$|G| = |(P_1 + 2 * P_2 + P_3) - (P_7 + 2 * P_8 + P_9)| + |(P_3 + 2 * P_8 + P_9) - (P_1 + 2 * P_4 + P_7)| \quad (24)$$

The Figure 16 shows the way the Sobel Operator recognizes edges characterized by sudden variations in pixel intensity.

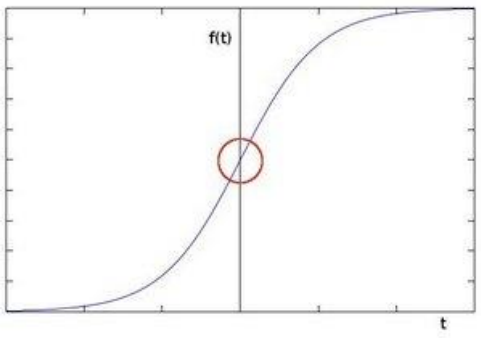


Figure 16:: Sobel: An edge is shown by the "jump" in intensity (opencv)

When we plot the first derivative of the intensity function, the increase in intensity becomes much more apparent. The Figure 17 shows that edges can be noticed in locations where the gradient is higher than a particular threshold value. Furthermore, a quick change in the derivative will result in a change in pixel intensity.

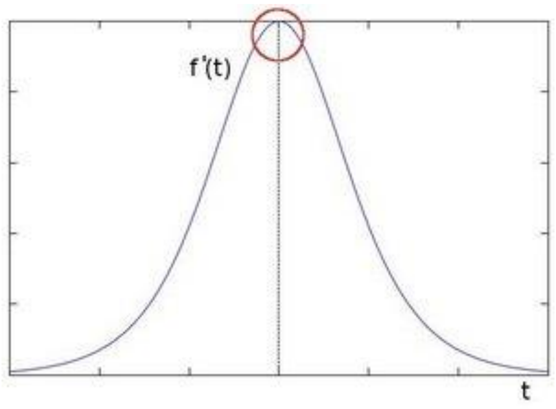


Figure 17: The edge "jump" can be seen more easily if we take the first derivative (actually, here appears as a maximum)

K-means

The k-means algorithm is an unsupervised learning algorithm employed to create a binary segmentation of each example. k-means splits data into several k groups (clusters), where k is the number of clusters with minimum error. The k-means algorithm consists of two separate phases. The first phase calculates the k centroid, and in the second phase, each point is assigned to the cluster closest to that data point. There are various ways of calculating the distance to the nearest centroid. The Euclidean distance is one of the most commonly utilized approach. When an iteration is done, a new centroid for each data point is computed. The centroid point of each cluster is the position at which the total of the distances from all of the objects in that cluster is the shortest. As a result, k-means is an iterative method that minimizes the sum of the distances between each object and its cluster centroid over all clusters. (Dhanachandra, Manglem, & Chanu, 2015) The algorithm for k-means clustering initializes some clusters k and center. Then, taking into account the Equation 25, it calculates the Euclidean distance d between the center and each pixel of an image.

$$d = ||p\ (x, y) - c_k)|| \quad (25)$$

Moving on, it commits all the pixels to the closer center based on distance d. Finally, pixels have been committed are recalculated to the new position of the center based on the Equation 26.

$$c_k = \frac{1}{k} \sum_{y \in c_k} \sum_{x \in c_k} p(x, y) \quad (26)$$

This process is repeated until it meets the allowance or the given error value. K-means has clustered a flattened vector of each grayscale image in 2 clusters.

U-net

A state-of-the-art way to create a segmentation model is to train a U-net deep neural network in an appropriate dataset to segment new lesions using the training weights. The UNet architecture was presented by Ronneberger et al. (Ronneberger, Fischer, & Brox, 2015) for biomedical image segmentation. Although a typical CNN concentrates to image classification tasks, we are also interested in localizing specific structures in biological applications. UNet is committed to resolving this issue. It can localize and recognize boundaries since it performs classification on each pixel, resulting in the same input and output size. The presented architecture in Figure 18 has a "U" shape. It has a symmetrical shape and it includes two main parts. The left or the contracting path, and the right part or expansive path, which is composed by transposed 2d convolutional layers. (Ronneberger, Fischer, & Brox, 2015)

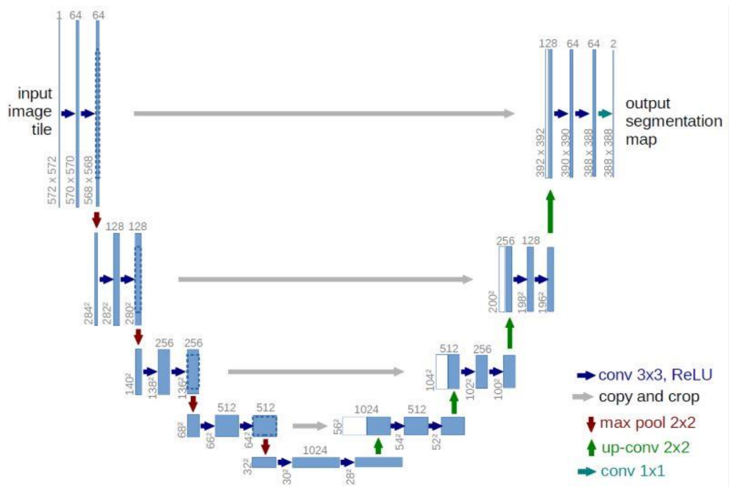


Figure 18: U-Net architecture

Contracting Path – Encoder Blocks

The contracting path or encoder operates as the feature extractor of the input image through the encoder blocks. In each process, there are two convolutional layers. The amount of channels increases the depth of the image. The red arrow in the Figure 14 represents the max pooling which reduces the image dimensions from 572x572 to 568x568 in the first block. It follows a ReLU activation function. This operation is repeated 3 more times to reach the bottommost, where the image size is 28x28x1024.

Expansive Path – Decoder Blocks

In the expansive path or decoder blocks the picture size increases into its original shape, via transposed convolution. The expansive path produces a segmentation mask. The decoder uses a 2x2 transpose convolution which is merged with the correlating skip connection feature map from the expansive path. Skip connections supply features that are lost because of the network's depth. Then, there are two convolutions, each one is followed by a ReLU activation function. The last convolutional layer delivers the output via 1x1 convolution with sigmoid activation function, which returns a segmentation mask. UNets can achieve image segmentation predicting an image mask pixel by pixel. To produce good predictions based on limited amount of examples, data augmentation is recomended. (Ronneberger, Fischer, & Brox, 2015)

Color space segmentation

Color space is an abstract mathematical model, where there is a specific description of how the components are set to be defined. In addition, it describes the manner human color vision can be modeled. There are various color spaces such as RGB, HSV, Lab, Luv, and YCbCr. RGB images can be converted to another color space using transformation functions. The RGB color space is the primary form of image representation, but some applications may find it more convenient to find other color spaces.

The HSV (Hue, Saturation, Value) color space is related to the RGB color space. Hue is the primary color humans perceive. Saturation is the quantity of white light together with the hue. Value is the brightness or the intensity. To simplify, Hue indicates to hue, Saturation to shade, and value to tone. The angle dimension represents Hue(H), starting with Primary Red at 0°, moving through Primary Green at 120°, Primary Blue at 240°, and ending with Red at 360°. The distance from the center axis of the HSV cylinder corresponds to the Saturation (S). A saturation value moving towards the outer edge means that the color value has reached the maximum value for the color defined by the hue. The central vertical axis of the HSV color space is the Value (V). It ranges from black at the bottom with lightness or value of 0 to white at the top with a lightness or value of 1. (Hema & Kannan, 2019) From the different channels of HSV we can identify the lesion and background because of their difference in intensity.

The CIE L*a*b color space has a channel for Luminance (Lightness) and two more color channels the a and the b, the chromaticity layers. The first one, the a* layer represents the fall of the color along the red-green axis, and respectively the b* layer represents the fall along the blue-yellow axis. Negative values of a* represent green and positive values represent magenta; negative values of b* represent blue and positive values represent yellow. It is significant to mention the property of this color space which gives us the ability to communicate different colors on different devices (device independent). (Bora, Kumar Gupta, & Khan, 2008)

CIE Luv color space is designed to be uniform. It combines the L, U, V components which have the range of [0,100], [-134,220], [-140,122] respectively. The CIE Luv color space ignores the part about the luminance scale and it uses a logarithmic scale covering a more extensive range of values. Moreover, it describes colors of white light range in xyz space, which makes it not to be device independent. (Umbaugh, Moss, & Stoecker, 1992)

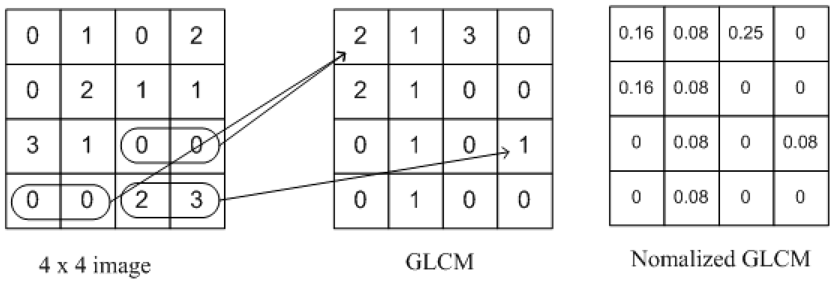
The YCbCr color space represents a luminance component (Y) and two chrominance components (Cb and Cr). The Y component represents the light intensity in this color space. The intensity of the blue and red components corresponded to the green component is represented by the Cb and Cr components, respectively. The YCbCr color space mimics human vision, utilizing the efficiency of the human eye. The light intensity affects the human eye more than the hue changes. Therefore, the human eyes comprehend more luminance information rather than other components. The YCbCr color space takes advantage of this property to reach an efficient representation of images. (Rahmat, Chairunnisa, Gunawan, & Sitompul, 2016)

The color spaces were used as segmentation method taking the advantage of the color intensity transformation. The target was to transform the color intensity of each example, to describe better the texture features. Those different color spaces (HSV, L*a*b, Luv, YCrCb) smooth out low color intensity differences, and instead they focus on high color intensity changes, which in this case are the skin and the lesions.

2.4 Handcrafted Features

Gray-level co-occurrence matrix (GLCM) Features

The Gray-level co-occurrence matrix (GLCM) is a method in statistical image analysis used to estimate image properties in the context of second-order statistics. The GLCM examines the relationship between two neighboring pixels as a second-order texture in an offset. GLCM is the 2-D matrix of joint probabilities $p_{d,\theta}(i,j)$ between pairs of pixels which are separated by a distance d in a certain direction θ defined 14 statistical features from the co-occurrence matrix of grey levels for texture classification. (Benčo & Hudec, 2007) If pairs of highly correlated pixels exist, the inputs are collected along the diagonal of the GLCM matrix. For each distance d and direction θ, the computations are taking considerable time. The distance and the number of alignments is confined to a restricted number of sets. A smaller number of intensity level can decrease the computational time. The classification of fine textures requires the use of small values for d while the classification of coarse textures requires the use of large values for d. Once the matrices are computed, each one must be condensed to a few numbers to classify the texture. (Wang & He, Texture classification using texture spectrum, 1990) A pair of pixels includes reference and neighbor pixels. The specific spatial relationship between the reference and neighbor pixels are defined before the GLCM calculation. For example, we can specify that the neighbor pixel is 1 pixel to the right of the current pixel, or 3 pixels above it, or 2 pixels diagonally (one of NE, NW, SE, SW) from the reference pixel. When a spatial relation is defined, a GLCM of size (range of intensities x range of intensities) is shaped, all initialized to 0. An 8-bit single-channel image has a GLCM of 256x256. The image is then crossed and increases the corresponding matrix cell for each intensity pair to the specified spatial relationship. In the Figure 19 we observe how GLCM calculates the value of GLCM(0,0). It takes the value 2, because there were two horizontal instances contain the values 0 and 0. Also, the calculation of GLCM(2,3) is 1 because one there was one horizontal instance which contain the values 2 and 3.



0	1	0	2
0	2	1	1
3	1	0	0
0	0	2	3

2	1	3	0
2	1	0	0
0	1	0	1
0	1	0	0

0.16	0.08	0.25	0
0.16	0.08	0	0
0	0.08	0	0.08
0	0.08	0	0

Figure 19: Texture descriptor with GLCM

So, GLCM[i,j] entries contain the number of times this intensity pair appears in the image with the defined spatial relationship. The matrix can be symmetric by adding it to its transpose and normalizing it. Each cell expresses the probability of occurrence of that intensity pair in the image. Once the GLCM is computed, the texture properties can be defined from the matrix to represent the textures in the image.

The Energy/Angular Second Moment provides information on image homogeneity; it has low When the probabilities of the grey level pairs are relatively similar the energy has low values, and if not it has high values. Energy is computed as in Equation 27:

$$\sum_{i=0}^{G-1} \sum_{i=0}^{G-1} P(i,j|d,\theta)^2 \quad (27)$$

The entropy measures the disorder of the GLCM and it is computed as in Equation 28:

$$-\sum_{i=0}^{G-1} \sum_{i=0}^{G-1} P(i,j|d,\theta) log_2\big(P(i,j|d,\theta)\big) \quad (28)$$

The correlation measures the grey level linear dependence between pixels at the specified positions. When the values are uniformly distributed in the GLCM, correlation has high values. The local homogeneity (also called inverse difference moment) is high when the same pairs of pixels are found (e.g., in the case of a spatial periodicity). It is computed as in Equation 29:

$$\sum_{i=0}^{G-1} \sum_{i=0}^{G-1} \frac{P(i,j|d,\theta)}{i+(i-j)^2} \quad (29)$$

The Inertia or contrast quantifies local variations present in the image and it is computed as in Equation 30:

$$\sum_{i=0}^{G-1} \sum_{i=0}^{G-1} P(i,j|d,\theta) \quad (30)$$

The Figure 20 shows the pixels 1 and 5 are 0° (horizontal) nearest neighbors to the center pixel; pixels 2 and 6 are 135° nearest neighbors; pixels 3 and 7 are 90° nearest neighbors, and pixels 4 and 8 are 45° nearest neighbors to the center pixel.

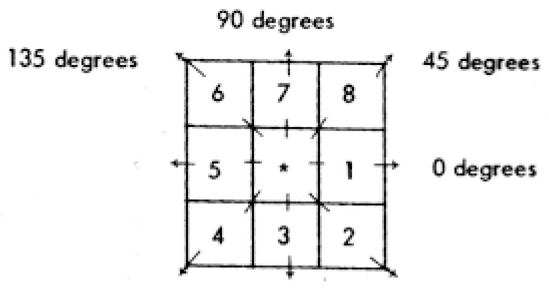


Figure 20: 3x3 window definition and spatial relationship for calculating Haralick texture measures.

Local Binary Pattern (LBP) Features

Local Binary Patterns (LBP) are based on the Texture Spectrum model from 1990. (Wang, Texture Unit, Texture Spectrum, and Texture Analysis, 1990) LBP is an effective texture method which labels the pixels of an image by thresholding pixels around and retrieving the result as a binary number as it is described in the Figure 21. (Xia, et al., 2018)

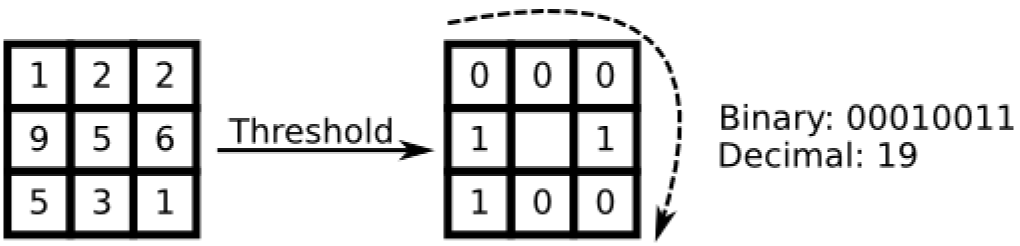


Figure 21: The LBP feature extraction process

In its simplest form, the LBP feature vector is created in the following way: It divides the examined window into cells (e.g., 16x16 pixels for each cell). Then, for each pixel in a cell, it compares it to its 8 neighbor pixels. It follows the pixels along a circle, from left to right or right to left). The neighbor pixels considered can be changed by varying the circle's radius around the pixel, and the quantization of the angular space P. If the center pixel's value is greater than the neighbor's value, it writes "0". Otherwise, it writes "1". This process gives an 8-digit binary number. After all, it calculates the histogram over the cell of the frequency of each number occurring. This histogram can be seen as a 256-dimensional feature vector. The histograms are normalized and concatenated with all cells. This gives a feature vector for the entire window. The feature vectors can be used by machine learning algorithms to classify images. The Figure 22 describes several circumstances of LBP and shows how the radius and the quantization value of the angular space affect the circle along the pixels.

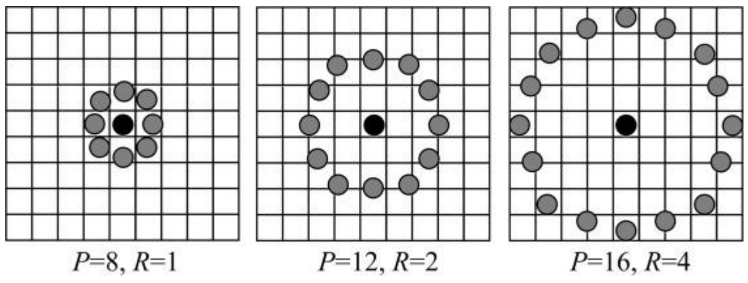


Figure 22: Example of multiscale LBP

Given a referenced pixel P (s, t) and its surrounding pixels Pk, the $LBP_{r,n}(s,t)$ is calculated by the Equation 31:

$$LBP_{r,n}(s,t) = \sum_{k=0}^{n-1} \mu(P(s,t) - P_k(s,t)) * 2^k,$$

$$\mu\,(m) = \begin{cases} 1, & m \geq 0 \\ 0, & m < 0 \end{cases} \qquad (31)$$

Where r is the radius of a circle, and n is the number of surrounding pixels in the circle with radius r.

Color Moments

Color moments are metrics that may be used to identify images based on their color characteristics. The computation of these moments yields a color similarity measurement between images. The assumption behind color moments is that the distribution of color in a picture can be read as a probability distribution. Several moments describe probability distributions (e.g., mean and variance) differentiated by normal distributions. As a result, if a picture's color follows a certain probability distribution, the moments of that distribution can be used as features to identify that image based on color. Color moments are scaling and rotation invariant because they describe the total information of each example.

Mean: The average color value for each channel of an image.

$$E_i = \sum_{j=1}^{N} \frac{1}{N} p_{i,j} \qquad (32)$$

Standard Deviation: The square root of the variance of the distribution of each channel.

$$\sigma_i = \sqrt{\left(\frac{1}{N}\sum_{j=1}^{N}(p_{i,j} - E_i)^2\right)} \qquad (33)$$

Skewness: A measure of the degree of asymmetry in the distribution of each channel.

$$s_i = \sqrt[3]{\left(\frac{1}{N}\sum_{j=1}^{N}(p_{i,j} - E_i)^3\right)} \qquad (34)$$

Kurtosis: Kurtosis is similar to skewness; it provides information about the shape of the color distribution of each channel. More specifically, kurtosis measures how extreme the tails are compared to the normal distribution.

$$\mathrm{Kurt}[x] = E\left[\left(\frac{X - \mu}{\sigma}\right)^4\right] \qquad (35)$$

Color constancy is a detection mechanism of color independent of the light source and it comprises low-level statistics-based methods like grey-world (GW), max-RGB, and grey-edge (GE). (van de Weijer J. , et al., 2007) This task is accomplished by performing two separate steps: estimating the light source's color in RGB coordinates and transforming the image. (Barata, Marques, & Celebi, 2014) Two of the most used simple techniques for estimating the color of the light are the grey-world and max-RGB algorithms. For a color image I, each component of the illuminant is estimated using the Minkowski norm (Thompson, 1996) as follows:

$$\left(\frac{\int (I_c(x))^p dx}{\int dx}\right)^{1/p} = ke_c, c \in \{R, G, B\}, \qquad (36)$$

Where I_c represents the c^{th}color component of an image I. The x = (x, y) is the position of each pixel, k is a normalization constant that ensures that $e = [e_R, e_G, e_B]^T$ has unit length, using the Euclidean norm. P is the degree of the norm. After estimating e, the next step is to transform the image I. A simple way to model this transformation is the von Kris diagonal model. (Feng Xiao*a, 2013)

$$\begin{pmatrix} I_R^t \\ I_G^t \\ I_B^t \end{pmatrix} = \begin{pmatrix} d_R & 0 & 0 \\ 0 & d_G & 0 \\ 0 & 0 & d_B \end{pmatrix} \begin{pmatrix} I_R^u \\ I_G^u \\ I_B^u \end{pmatrix} \qquad (37)$$

where $[I_R^u, I_G^u, I_B^u]^T$ is the pixel value acquired under an unknown light source, and $[I_R^t, I_G^t, I_B^t]^T$ refers to the pixel value transformation as it would appear under the normal light source, considered to be the perfect white light. The matrix coefficients $\{d_R, d_G, d_B\}$ are the mapping parameters, which are related to the estimated illuminant e as describes the Equation 38:

$$d_c = \frac{1}{e_c}, c \in \{R, G, B\} \quad (38)$$

2.5 Deep learning for Medical Images

Deep learning is able to deal with several problems, but it needs a huge amount of data and high computational power. In the medical imaging field, data are limited because of sensitivity issues and difficulties in annotation task (the lack of experts makes the problem expensive and time consuming). Because deep learning, even if medical imaging data are limited, achieved very promising results, it captured the researchers' attention, and until today, it remains a very challenging task which is developing rapidly. Due to the limited access to the data, overfitting is usually an error that occurs. To overcome this problem, transfer learning methods are applied. As it is already mentioned in previous subsection, transfer learning is a method to transfer knowledge which was gained in a large amount of data and it is ready to be used in new conditions. For this reason, transfer learning methods are very popular in medical imaging field such as pre-trained networks as feature extractors, fine-tuning or part of a pre-trained architecture combined with a new deep learning architecture and the weights to be frozen or updated during the training of a new model. Another technique that is popular in medical imaging to prevent overfitting error is data augmentation. (Inoue, 2018) The key factor of this method is the image generation using an original image to achieve data oversampling. This occurs by shifting, zooming, rotating, flipping the dataset's images to increase its diversity. When we apply data augmentation in medical imaging it is important to notice the output of the model and whether data augmentation affect the results or not. For instance, in X-Ray images, heart and aorta are positioned to the right-hand side of the image. By applying horizontal flip, the image can be mistaken for a rare medical condition named situs inversus, (mirrored organs syndrome) (Sirazitdinov, Kholiavchenko, Kuleev, & Ibragimov, 2019) In general, a random combination of data augmentation geometrical transformation does not always affect positively the results. Imbalance class distribution is an another challenging issue we face in medical imaging where the amount of the available example of each class differs a lot. The class with most examples is named majority class and the class the less examples is named minority class. When we train a model on an imbalance dataset, the model becomes biased to the class with the most examples (majority class). Thus, even if the model's accuracy could be high, the model is not able to predict accurately the minority's class examples. In medical imaging problems, different metrics are used as evaluation methods to solve this issue, such as Sensitivity (Recall/True Positive Rate), Specificity (True Negative Rate), Recall, Precision, F1-Score.

Multiple deep learning research approaches have shown positive results in medical imaging field using X-Ray, CT, MRI, Histopathology image data. Recent research mentions that the increase of labeled X-Ray data has grown up the interest to expand more the deep learning techniques. (Baltruschat, Nickisch, Grass, Knopp, & Saalbach, 2019) Baltruschat et al investigate several ResNets (ResNet-38-50-121) network architectures where they experiment in approaches with fine-tuning or without and in the construction of an X-Ray deep learning architecture dedicated to X-Ray images from scratch. The ResNet-50-large-VP achieved Sensitivity of 87,8%, Specificity of 85,9% and AUC of 94,35%. (Baltruschat, Nickisch, Grass, Knopp, & Saalbach, 2019)

Ismael et al. introduced a deep learning approach for brain cancer using MRI scan images diagnosing 3 brain tumor types, the Meningiomas, the Gliomas, and the Pituitary tumors. In this approach a ResNet-50 network architecture achieving accuracy of 97%, precision of 98%, recall or 97%, and F1-Score of 97%. (Ismael, Mohammed, & Hefny, 2020)

An another research using histopathological images and deep learning makes multi-classification for breast cancer. (Han, et al., 2017) This paper introduces the DCNN deep learning architecture which is dedicated to solve the multi-classification problem of breast cancer and achieves the best results of average accuracy of 93,2%, in comparison with several popular CNN architectures such as AlexNet (Krizhevsky, Sutskever, & Hinton, 2012) which achieved 83% accuracy in the binary classification breast cancer task. (Spanhol, Oliveira, Petitjean, & Heutte, 2020)

2.6 Machine Learning in Dermoscopy

Dermoscopy is a non-invasive technique of skin imaging and is a critical method of diagnosing melanoma. The ABCDE rule of dermoscopy or the 7-point checklist was designed for clinicians to assist them to clarify the differences between benign and malignant skin lesions. Asymmetry, Border, Color, Diameter and Evolving (ABCDE) are characteristics of skin lesions that dermatologists examine to diagnose skin lesions. (Beaumont, 2021)

- Asymmetry – Melanoma is often asymmetrical, which means the shape isn't uniform. Non-cancerous moles are typically uniform and symmetrical in shape.
- Border – Melanoma often has borders that aren't well defined or are irregular in shape, whereas non-cancerous moles usually have smooth, well-defined borders.
- Color – Melanoma lesions are often more than one color or shade. Moles that are benign are typically one color.
- Diameter – Melanoma growths are normally larger than 6mm in diameter, which is about the diameter of a standard pencil.
- Evolution – Melanoma will often change characteristics, such as size, shape or color. Unlike most benign moles, melanoma tends to change over time.

Expert clinicians search for visual features to clarify and to diagnose a skin lesion in most dermoscopy methods. However, in case of inexperienced clinician, diagnosis remains a very challenging task. Also, misdiagnosis of a skin lesion is a reason of many skin cancer deaths. (Urbancek, Fedorcova, Tomkova, & R. Sutka, 2015) Regarding the ABCDE rules include checking for asymmetric shape, irregular border, unusual coloring (color distribution), diameter, and the evolution of the fault in units of time, is the most critical rule for diagnosis. (Abbasi, et al., 2004) (Petra Fedorcova & Roman Sutka, 2015) However, this rule cannot be a general rule for recognizing malignant skin lesions as there may be cases with an asymmetric shape, irregular border, or unusual coloration that is a benign skin lesion. At the same time, there may be an incident with a symmetrical shape, usual coloration, and a standard malignant border. The melanoma detected in its early stages, which has not spread beyond the small point of the lesion, is almost always curable. The diagnosing errors occur and the lack of experienced medical staff underline the need of Computed Aided Diagnosis (CAD) systems. (M. E. Celebi Y. A., 2017) Due to the great evolution of image capturing devices, the image quality has improved, attracting the research interest to the automated classification of dermoscopy images. As a result, machine learning and computer vision techniques are being used to develop CAD systems that can detect melanoma and assist dermatologists in early diagnosis to minimize diagnostic errors. As a real-world medical imaging problem, melanoma classification encounters the problem of imbalanced data, where the ratio of malignant cases in contrast with benign and dysplastic instances is much lower. Several approaches try to tackle this problem targeting to data balancing. Over-Sampling and Under-Sampling are two basic techniques that they try to balance the dataset and they are used to the melanoma imbalance dataset problem. (Rastgoo, et al., 2016) Some approaches are using a hybrid method to overcome the imbalance problem applying a random over-sampling method which is followed by data augmentation. (Sayed, Soliman, & Hassanien, 2021) However, a different approach tries to oversample the minority class by synthesizing new examples using the Synthetic Minority Over-sampling Technique (SMOTE) and under-sampling techniques using the Edited Nearest Neighbors by removing samples close to a decision boundary. (Bezerra, et al., 2018)

The first approaches of melanoma classification made use of three basic processes. The preprocessing and skin lesion segmentation, the feature extraction, and the feature classification. The preprocessing step of the image data refers to image resizing, contrast enhancement, noise reduction and hair removal. (A.J.Vyavahare & Mahajan, 2013) (Delibasis

& Maglogiannis, 2015) The next step is the segmentation of skin lesions identifying the region of interest (ROIs) excluding the non-lesion area. The literature for medical image segmentation covers several techniques which can be implemented individually or in combination. Some of those methods are active contours, clustering, histogram thresholding, edge detection. (Bakheet, 2017) (Abhilash, 2021) (Majtner, Yildirim-Yayilgan, & Hardeberg, 2016) The feature extraction process, is the step where features are collected based on multiple characteristics such as border irregularity (Q. Abbas, 2011) (R. Erol, 2017) asymmetry (W. Stoecker, 1992), color (S. Seidenari, 2005) (Zornberg, 2014) and texture. (H. Iyatomi, 2008) (M. E. Celebi H. A., 2007) Finally, those features are used as inputs to feed a machine learning classifier such as support vector machine (Ghalib, 2017), bayesian classifiers (L. Li, 2014), decision trees (Ghalib, 2017) and k-nearest neighbors (Ghalib, 2017) (L. Li, 2014) (Celebi, 2016) Riaz et al. (Riaz, Hassan, & Javed, 2014) described a method for classifying dermoscopy pictures using a mix of texture and color parameters. The texture of their skin is made up of a variety of Local Binary Patterns (LBP). The authors employed typical HSV histograms to extract color features.

Deep learning approaches such as deep convolutional neural networks and transfer learning have shown that they achieve better results. The feature extraction step is automated and depends on the model's architecture to detect and extract descriptive features to train properly. (A. Esteva, 2017) (J. Kawahara, 2016) (Shen, 2018) (A. Rezvantalab, 2018) However, several approaches are trying to combine those Deep learning with traditional machine leaning approaches to improve the models' efficiency and accuracy. Mollersen et al. demonstrated using divergence-based color characteristics in 2015, where the divergence between the distribution of pixel color values in a lesion picture and the pixel color values of either a benign or malignant model is used to create these characteristics. (Møllersen, Hardeberg, & Godtliebsen, 2015) Moreover, texture-based features such as generalized co-occurrence matrices, gradient histograms, Local Binary Patterns, and RSurf features have been used combined with the AlexNet network (Krizhevsky A. I., 2012) to extract features which used to train a Support Vector Machine. (Majtner, Yildirim-Yayilgan, & Hardeberg, 2016) Codella et al. (Codella, et al., 2015) use a mix of deep learning, sparse coding, and support vector machine (SVM) learning techniques for melanoma classification. On a dataset of 2624 clinical instances of melanoma, atypical nevi, and benign lesions, their technique had a classification accuracy of 93.1%. Also, the research of (Xie, et al., 2017), began by identifying 57 descriptive features, including 50 color and texture characteristics and seven novel lesion border features. The mentioned features are designed to be insensitive to the imperfection of the lesions. The Principal Component Analysis (PCA) method (Jolliffe, 2005) was used to reduce feature dimensionality and improve classification performance by removing less important or noisy data. A collection of neural networks that operate together to increase net diversity. The model was created by mixing BP neural networks with fuzzy neural networks. The exported features were input for Random Forest, KNN, Gentle Adaboost, and SVM classifiers achieving an accuracy of 94,17%, sensitivity of 95%, and specificity of 93,75%. Hagerty et al, (Hagerty, et al., 2019) proved that the inclusion of classical handcrafted image processing features and the Deep learning features (ResNet-50) resulted in a better performing model, increasing the AUC from 0.83 without the HC features, to 0.94. Regarding

Kaggle's ISIC2020 challenge leader score, the strategy of a series of different analyses is beneficial, as the top scores improve this methodology. The best solution earned a ROC-AUC score of 94.90%, using 18 networks. (Ha, Liu, & Liu, 2020) The EfficientNet-B3 - B7 models, trained at a different resolution, and the SE-ResNext101 and ResNest101 were used. That means that networks trained with a range of input sources can produce a diverse collection of results but also means, making computation both costly and time-intensive.

3 Methodology

This section describes the suggested methodology for solving the problem of melanoma classification.

3.1 Proposed Methodology

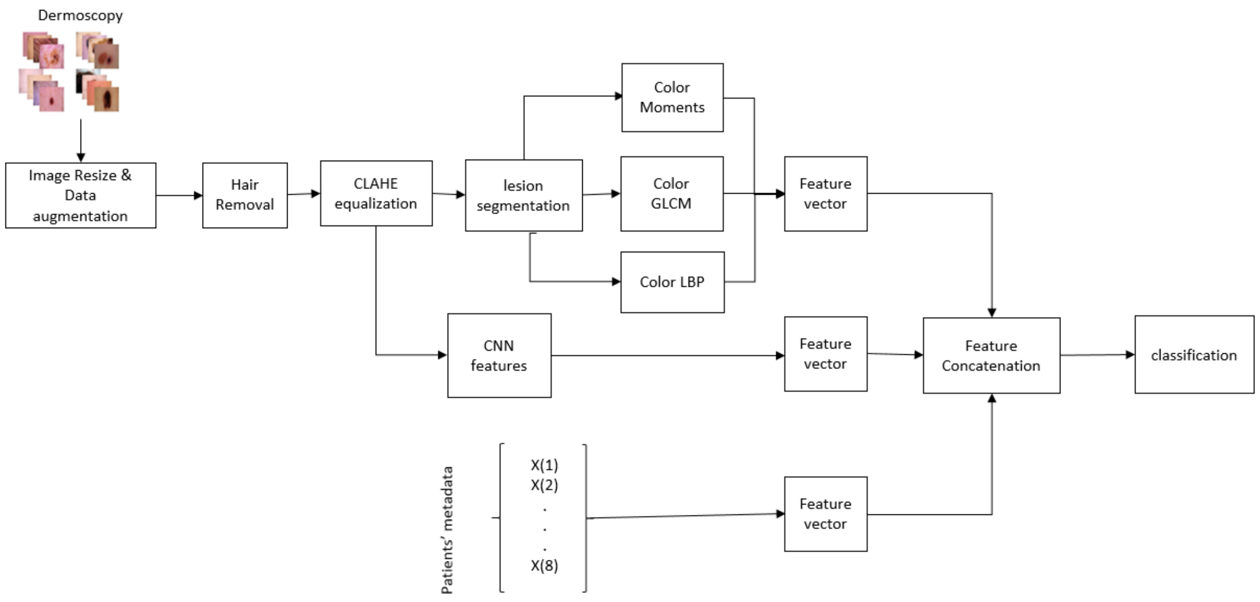


Figure 23: Proposed architecture

In our approach as is it described in the Figure 23, we combine CNN features with handcrafted features and metadata features for melanoma classification. It is divided into 8 steps. The Image Resize and Data Augmentation, the hair removal, the CLAHE equalization, the lesion segmentation, the handcrafted feature extraction, the CNN feature extraction, the metadata features, and the features classification.

For pre-processing and handcrafted feature extraction phases, we embraced the recommended methodologies and the existing code resource available in the GitHub repository "Multi-Color-Space-Features-for-Dermatoscopy-Classification" (Saha, 2020). Especially, in the initial stages of pre-processing, the repository guided our approach to apply hair removal and Contrast Limited Adaptive Histogram Equalization (CLAHE) on our dataset. In the context of image segmentation, we adopted the color space techniques recommended by Saha. Leveraging the established color space conversions, we navigated the complexities of dermoscopic images, preserving vital color information while enhancing luminance contrast. Subsequently, in the feature extraction phase, we integrated Gray-Level Co-occurrence Matrix (GLCM), Local Binary Pattern (LBP) and color moments techniques into our feature extraction pipeline. These techniques, enabled us to derive meaningful features from dermoscopic images, facilitating a more comprehensive analysis.

Keeping all these in mind for the pre-processing and the handcrafted methods, we further investigated with various lesion segmentation techniques, deep learning (CNNs) and metadata feature extraction to obtain a feature classification model. It is expected that by combining all these techniques will achieve higher accuracy of our model.

3.2 Image Resize and Data Augmentation

The first step of our approach is the image resize and data augmentation. We resized the skin lesion images to 224x224x3 dimensions, which are the input shape of the CNN architectures that employed. Also, an oversampling strategy is utilized to augment the minority class (melanoma) by shifting, rotating, zooming and flipping the skin lesion examples to deal with the imbalanced dataset. Although ISIC spent many years creating this dataset containing tens of thousands of annotated dermoscopic pictures, it is still considered a small to medium dataset for training deep architectures with millions of parameters. The data augmentation approach was utilized to increase the variety and the size of the train set without actually gathering additional data. Since each model has been built to perform better in different input-sized images, the whole images are resized according to the default input size of each model. The default input size for the CNN models we were used is 224x224, and the resize is implemented based on those dimensions. The resizing task was applied to the training, validation, and test set. The Table 1 describes the augmentation process after applying different values for various augmentation circumstances, such as a random rotation between 0 and 45 degrees to the entire train dataset. Also a width and height shift of 0.2, shear and zoom with a range of 0.2. Finally, the horizontal flip is used for image horizontal flipping.

Table 1 Data augmentation process

Random rotation	Between 0-45
Width shift range	0.2
Height shift range	0.2
Shear range	0.2
Zoom range	0.2
Horizontal flip	True

3.3 Hair Removal and CLAHE equalization

The next step of the pre-processing techniques is the hair removal and the CLAHE equalization methods. We explore essential pre-processing techniques, including hair removal and Contrast Limited Adaptive Histogram Equalization (CLAHE), as mentioned by (Saha, 2020). These techniques have a significant role in enhancing the quality of dermoscopic images for subsequent analysis.

For the hair removal task, to address the issue of dark hair occlusions in dermoscopic images, we first transformed the RGB image channels into the LUV color space using the cv2.cvtColor

OpenCV library (Bradski, 2000). A spherical structuring element (SE) is created using cv2.getStructuringElement. The radius of this structure is defined by the SE_radius parameter. A morphological closing is applied to the L channel of LUV image using cv2.morphologyEx in order to remove the dark regions, i.e. hair. Then the difference between the two, morphological closing and the L channel image, isolates the presence of the hair, creating a hair mask. Then, by applying hysteresis threshold, onto the hair mask, we further remove any remaining noise, and the presence of hair becomes more significant. This technique provides the intensity levels of hair pixel, and sets a threshold interval below which is assumed any low intensive pixel is not significant, i.e. not hair. The high and low threshold values were set to default. Then, this hair mask is subtracted from the L channel of LUV image, producing the final LUV image without hair. Finally, the CIE Luv image is converted to RGB.

In the next step we applied Contrast Limited Adaptive Histogram Equalization (CLAHE) to improve the contrast of the final images. CLAHE operates by defining the ClipLimit which sets the threshold for contrast limiting and tileGridSize which sets the number of tiles in the row and column. The values we used were the default ones as mentioned by (Saha, 2020).

Table 2 Notation table

Parameters	Description	Value
SE_Radius	Determines the radius of the spherical structuring element (SE) used in the morphological closing operation.	13
Threshold	Controls the threshold value used during hysteresis thresholding when generating the binary mask	20
ClipLimit	Sets the threshold for contrast limiting	0,9
tileGridSize	sets the number of tiles in the row and column.	1

Then, our approach divided into 2 different pipelines. The first pipeline follows the handcrafted features extraction, where we experiment with several image segmentation methods to define the region of interest (lesion) and then to extract features creating a feature vector. In the second pipeline, we use CNN models to extract a second feature vector.

3.4 Image Segmentation

The image segmentation task is the previous step before we extract handcrafted features. In this approach, we experimented with various segmentation methods in order to define which method supports in the most proper way to create segmentation masks and to define the lesions. It is important to distinguish between the techniques that were specifically used for lesion segmentation and those that were used for simple filtering and global feature extraction.

Lesion Segmentation Techniques

The primary goal of lesion segmentation is to accurately identify and isolate the region of the image that contains the skin lesion. The following techniques were used for lesion segmentation:

1. **K-Means Clustering**: K-means clustering was used to partition the pixels in the image into different clusters based on their intensity values. This technique is effective in separating the lesion from the surrounding skin.
2. **U-Net**: U-Net is a deep learning-based approach that was used for semantic segmentation of the skin lesions. It is particularly effective for biomedical image segmentation.
3. **Chan-Vese Algorithm**: The Chan-Vese algorithm is an active contour model used to detect objects in an image. In this study, it was used to detect the contours of the skin lesions.

Filtering and Global Feature Extraction Techniques

In addition to lesion segmentation, other techniques were used for simple filtering and global feature extraction:

1. **Sobel Operator**: The Sobel operator was used for edge detection. However, it is important to note that the way the Sobel operator was applied used did not constitute lesion segmentation. Instead, it was used for filtering and enhancing the edges in the image.
2. **Color Spaces**: Different color spaces were explored to analyze the color characteristics of the images. This was used for global feature extraction rather than segmentation.

The first image segmentation method is the Chan-Vesse active contour segmentation algorithm. The active contour estimates the pixel energy and it separates the pixels generating a parametric contour with similar characteristics. The active contour model defines the lesion segmentation by minimizing the energy function. The energy function minimizes the energy that acquired by threshold function which defines whether a given lesion is inside or outside the contour. The active contour is applied to the whole train images to generate masks for each example to segment the lesions using the scikit-image library (Van der Walt S. S.-I., 2014). The parameters of the chan-vesse active contour algorithm we set are the number of iterations equal to 100, threshold equal to 0,79, and smoothing level equal to 1.

The second method we applied is the sobel operator. The sobel operator identifies edges by detecting sudden variations in pixel intensity. A 2-D spatial gradient measurement on the lesion images is executed by the sobel operator which emphasizes areas that correspond to edges. The sobel operator estimates the absolute magnitude at each point of each given greyscale skin lesion examples. The sobel output of this process in an array of floats which describes the sobel edge map.

The third image segmentation technique is the k-means. We use the k-means clustering algorithm to group color pixels with similar color intensive values. We fixed the number of clusters as 2 in order to split each lesion image into background and foreground based on similar color intensities. In this way, we create a binary segmentation lesion mask. The algorithm clusters the values of each pixel into one of the 2 groups based on the sum of squared distances of samples.

The fourth image segmentation technique was the training of a UNet giving as input skin lesion examples and their masks to predict in new skin lesion examples their segmentation masks. The goal is to create automated predictions of lesion segmentation boundaries. The ISIC 2016 challenge contains 900 dermoscopic train images, paired with the expert manual tracing of the lesion boundaries in the form of a binary mask. Both the training number of examples and their lesion mask is 900. The test set contains 379 images and 379 masks of the same format as the training set. (I.S.I.C, 2016) The Figure 24 shows a skin lesion input example and its ground truth mask.

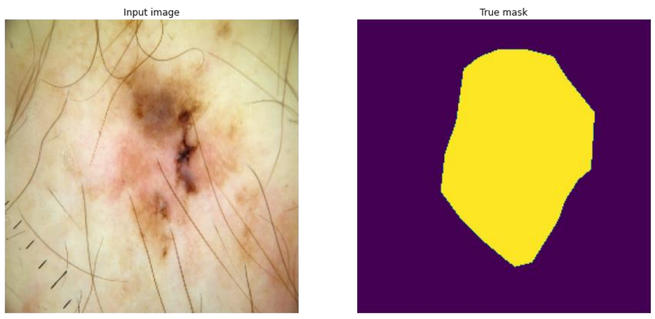


Figure 24: Original lesion(left), True lesion mask (right)

The U-net training pipeline contains the pre-processing step, the training step, and the evaluation step. The Figure 25 represents the UNet architecture we used to predict lesion segmentation masks.

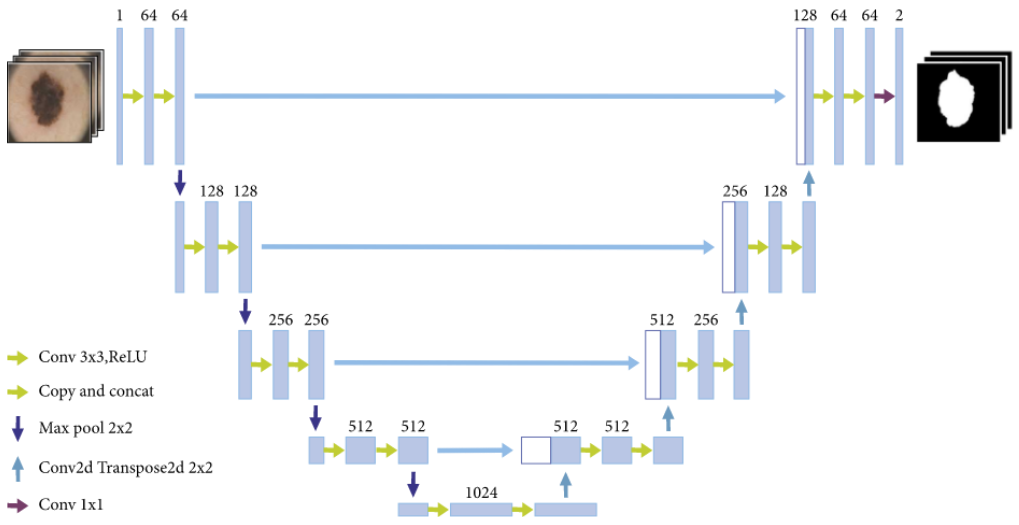


Figure 25: The U-Net architecture for melanoma segmentation

In the pre-processing phase, each example, both original and mask, was normalized by dividing each pixel by 255. Additionally, data augmentation was implemented using the random left and right flip in a round of 0.5.

In the training phase we created image batches of with size 16, and these batches were randomly shuffled. The UNet model has total parameters equal to 6,504,227, trainable parameters equal to 4,660,323, and non-trainable parameters equal to 1,843,904. The model was compiled with Adam optimizer, which is a stochastic gradient descent algorithm, that computes individual adaptive learning rates for different parameters, by estimating the first- and second-order moments of the gradients. (Kingma & Ba, 2015). Also, the training epochs are ten and the steps per epoch is equal to 42. The training and evaluation completed with loss: 0.1679, accuracy: 0.9328, validation loss: 0.2049 and validation accuracy: 0.9260.

The last technique we experimented is the color space. The color spaces were used as a method for image filtering, taking the advantage of the color intensity transformation, extracting global features. We transform each lesion training example into color spaces, to describe better the texture features. Those different color spaces are the HSV, the CIE L*a*b, the CIE Luv, and the YCrCb, which smooth out low color intensity differences, and instead they focus on high color intensity changes, which in this case are the skin and the lesions.

3.5 Feature Extraction

The next step is the feature extraction which refers to converting raw data into numerical features that can be processed while retaining the information in the original dataset. This can produce better results than applying machine learning directly to the raw data. Feature extraction performed handcrafted (manually) and automatically using the CNN models.

3.5.1 Handcrafted Features

Handcrafted feature extraction follows the image segmentation task, and it requires to identify and to describe the features relevant to a particular problem and to implement a method for extracting those features. Utilizing the insights available by (Saha, 2020), we implement three handcrafted feature extraction methods, the GLCM, the LBP and the color moments.

GLCM Features

The GLCM (GLCM) features combine GLCM features using 3 channels of color spaces such as RGB, HSV, Luv, Lab, and YCrCb. Each image is divided into three color channels to account for color information, with each channel represented by 256 gray levels. GLCMs (GLCM) are computed using each of the channels individually by calculating the

$$G_{x,\theta}, x \in \{r, g, b\}, \theta \in \{0°, 45°, 90°, 135°\}$$

$$G_{x,\theta}, x \in \{h, s, v\}, \theta \in \{0°, 45°, 90°, 135°\}$$

$$G_{x,\theta}, x \in \{l, a, b\}, \theta \in \{0°, 45°, 90°, 135°\}$$

$$G_{x,\theta}, x \in \{l, u, v\}, \theta \in \{0°, 45°, 90°, 135°\}$$

$$G_{x,\theta}, x \in \{Y, Cb, Cr\}, \theta \in \{0°, 45°, 90°, 135°\}$$

Moreover, it is computed the texture properties of each color channel as bellow:

$$Contrast_{x,\theta} = \left(C_{x,0} + C_{x,45} + C_{x,90} + C_{x,135}\right)/4$$

$$Energy_{x,\theta} = (E_{x,0} + E_{x,45} + E_{x,90} + E_{x,135})/4$$

$$Disimmilarity_{x,\theta} = (D_{x,0} + D_{x,45} + D_{x,90} + D_{x,135})/4$$

$$Corelation_{x,\theta} = (C_{x,0} + C_{x,45} + C_{x,90} + C_{x,135})/4$$

$$Homogenety_{x,\theta} = (H_{x,0} + H_{x,45} + H_{x,90} + H_{x,135})/4$$

Where x ∈[{r,g,b},{h,s,v},{l,a,b},{l,u,v},{Y,Cb,Cr}].

For the implementation of GLCM we used the scikit-image python library (Van der Walt S.-I. , 2014) .

In the case of the sobel operator's images, we have 1 channel in each example. We calculate the GLCM by computing the texture properties Contrast, Energy, Dissimilarity, Correlation, Homogeneity for the 1 channel of each training example.

The parameters to produce the greyscale co-occurrence matrix for each channel are: channel=3, bit_debth=8, distances=1, angles = 0, 45, 90, 135 degrees, levels = 2^8. We also selected to compute the following texture properties for each channel from the matrix to represent the textures in the image:

Contrast.mean(), Dissimilarity.mean(), corellation.mean(), homogeneity.mean()

The features which produced using the segmented lesions by active contours, k-means and UNet techniques are 4 for each R, G, B channel and the final vector has size of 12 features. The features which produced using the segmented lesions of sobel operator technique are 4 for the 1 channel and the final feature vector has size of 4 features. The features which produced using the segmented lesions of color spaces are 4 for each channel of each color space. We concatenated the GLCM_RGB, GLCM_HSV, GLCM_LAB, GLCM_YCrCb, GLCM_LUV. The final vector includes a set of 60 features.

LBP Features

The LBP (CLBP) features combine LBP features using 3 channels of color spaces such as RGB, HSV, Luv, Lab, and YCrCb. To retrieve color information, each image is divided into three color channels. To implement LBP, we used the scikit-image python library. (Van der Walt S.-I. , 2014) For each of the 3 channels, we set the number of circularly symmetric neighbors P = 8, a radius R = 2. Therefore, we included in the calculations of each window 8*2 number of points/pixels. Also, we set the method of determining the pattern as 'uniform', which improved the rotation invariance with uniform patterns and finer quantization of the annular space, the grayscale, and rotation invariant. We produced 10 features as the bins of the histogram created. We calculated the LBP for each channel as:

- LBP_RGB = lbp_R, lbp_G, lbp_B,
- LBP_HSV = lbp_H, lbp_S, lbp_V,
- LBP_YCrCb = lbp_Y, lbp_Cr, lbp_Cb,
- LBP_LUV = lbp_L, lbp_u, lbp_v,

- LBP_LAB = lbp_L, lbp_a, lbp_b,

The features which produced using the segmented lesions by active contours, k-means and UNet techniques are 10 for each R, G, B channel. The final vector has size 30 features. The features which produced using the segmented lesions of sobel operator technique are 10 for the 1 channel and the final feature vector has size of 10. The features which produced using the segmented lesions of color spaces are 10 for each channel of each color space. We calculated the LBP_RGB, LBP_HSV, LBP_LAB, LBP_YCrCb, LBP_LUV, and a final vector of 150 features was created.

The GLCM and the LBP features were normalized between 0 and 1 using the Equation 40:

$$Z = \frac{(X - \mu)}{\sigma} \quad (40)$$

Color Moments

Color moments include both shape and color information. if an image's color follows a certain probability distribution, the moments of that distribution can be used as features to identify that image based on color. As a result, we used color moments to describe probability distributions such as mean, standard deviation, skewness, kurtosis. We transform the segmented images which contain 3 channels into different color spaces (RGB, HSV, YCrCb, CIE Lab, CIE Luv), color constancy, and max color constancy for each color space, to extract color moments as features. We estimate the light source of an input image as proposed in (van de Weijer & Gevers, 2005). We constructed a function that uses the correct image function of Mina Sami (Mina, 2018) , which corrects image colors by performing diagonal transformation according to the given estimated illumination of the image and the grey edge function, which depending on its parameters, the estimation is equal to grey-world, max-RGB, general grey-world, shades-of-grey, or grey-edge algorithm.

To extract the color space color moments, we transform the RGB images into color spaces (HSV, YCrCb, CIE Lab, CIE Luv). Also we calculate the mean, the standard deviation, the skewness and the kurtosis of each example for each channel of the RGB, HSV, YCrCb, CIE Lab, CIE Luv color spaces. The extracted features from the color space color moments contain 60 features which include information about color moments of color spaces.

To transform images into Grey-World, first, it was used the Gaussian filters of scikit-image for pre-processing of input image setting sigma = 0. Then we set the Minkowski normalization = 1. The function returns the illuminant color estimation for the original RGB image (gwRGB). We calculate the color moments (mean, standard deviation, skewness, kurtosis) of the grey-world Color Constancy (gw) of each color space, such as gwRGB, gwHSV, gwYCrCb, gwLab, gwLuv. The extracted features contain 60 features which have information about the color moments information for each individual color channel of color spaces.

Finally, in the max-RGB Color Constancy we replace every pixel with its strongest RGB channel. Concentrating on less complicated color constancy based on low-level visual features, such as max-RGB. Max-RGB is a simple and quick color constancy algorithm that predicts the color of the light source based on the maximum response of the various color channels. (Land & McCann, 1971) To transform images into max-RGB, first, we use the Gaussian filters of scikit-image setting sigma = 0. Then we set the Minkowski normalization = -1. The function returns the illuminant color estimation for the original RGB image (max-RGB). We calculated the max Color Constancy of each color space, including max-RGB, max-HSV, max-YCrCb, max-Lab, and max-Luv. We extracted 60 features which contain information about color moments of including max-RGB, max-HSV, max-YCrCb, max-Lab, and max-Luv.

The color moment features finally generated a feature vector of size 180 for each example. The feature vector contains the calculations of each color moment (mean, std, skew, kurt) for each channel of each color space, grey-level color constancy color space, and max color constancy color space. There are 45 different channels, 4 color moments, and we receive a feature vector with a size of 180. The color moment features of 1 channel images, contain the mean, the standard deviation, the skewness and the kurtosis of this channel. The number of color moment features is 4 for each lesion example. The color moment features are normalized between 0 and 1 using the Equation 40.

3.5.2 Deep learning features

The deep learning features is the second feature extraction method of the pipeline for melanoma classification. Although data augmentation process oversampled the dataset by shifting, rotating, zooming and flipping, it is still not desirably large for training models with millions of parameters. To come up with the limited training skin lesion examples the transfer learning method was applied, using the pre-trained weights on the ImageNet dataset. The target of this is to use the knowledge which has already been gained about several objects, to the imported models and to use those weights to allow the model to specialize into skin lesion problem. We employed three CNN models as feature extractors, the VGG19, the DenseNet121 and the EfficientNet-B0. The models using transfer learning initialized the pre-trained weights on the ImageNet without fine-tuning the layers.

The VGG19 model includes 16 convolutional layers, 5 max-pooling layers and 3 fully connected layers. The input image shape of the model is 224x224x3 as the shape of the training set is after image resize process. Moreover, the non-linear ReLU activation function is employed to improve the computational time and the predictions. Also, we excluded the fully connected layers and we keep the pre-trained model with an output of size 4096 features, which is the result of the last pooling layer after the convolutional layers.

The DenseNet121 model requires fewer parameters than regular CNNs and it reuses features instead of gaining power from highly deep designs. The input size of the model's architecture is 224x224x3, the same size of the resized training examples. To extract features from DenseNet121, we excluded the fully connected layers and we keep the pre-trained model with an output of size 1024 features, which is the result of the last pooling layer after the convolutional layers.

The EfficientNet-B0 model is the base model of the EfficientNets where ϕ is set to 1 and twice as many resources are available, a, b, c are the results of a tiny grid search. For f=1, the values $\alpha = 1.2, \beta = 1.1, \gamma = 1.15$ of maintaining the $\alpha * \beta^2 * \gamma^2 \approx 2$.The input shape of the model's architecture is 224x224x3 as the resized training examples. As we did with the VGG19 and the DenseNet121 we excluded the fully connected layers and we keep the pre-trained model with an output of size 1280 features, which is the result of the last pooling layer after the convolutional layers.

The Deep Learning feature vectors are used both as base models to train individually the machine learning classifiers, and in combination with handcrafted features and metadata.

3.5.3 Patients' Metadata Features

The patients' metadata is the third feature vector we constructed. The metadata contain information about the diagnosis and site of the lesion, the approximate age and gender of the patient, and an anonymized patient identification number, which allows lesions from the same patient to be mapped together. The train metadata was merged creating a feature vector which includes categorical variables, such as the 'patient sex' and 'anatom site general'. Those variables were converted into binary vectors. An additional feature that was added is the age which was normalized into 0-1. The metadata feature vector after the transformations was applied contains features about patient sex, torso, lower extremities, upper extremities, head/neck, oral/genital, palms/soles, none, and age.

3.6 Image Classification

We produced three feature vectors using several feature extraction methods: CNN features, handcrafted features, and metadata features. We use the CNN feature vector as input to train the Random Forest and the XGBoost machine learning classifiers. Also, we combine the CNN feature vector with the handcrafted feature vector by concatenating those feature vectors. The new feature vector is used as input to train the same machine learning classifiers. Finally, in the last approach, we combine the CNN, the handcrafted and the metadata feature vectors. We concatenate those feature vectors and the new feature vector is used as input to train the same machine learning classifiers. The supervised machine learning classifiers detect valuable correlations between extracted feature data and predict whether or not a new patient's lesion image is melanoma. We linked each feature vector generated for each lesion to a binary label indicating whether or not the patient matching this lesion is melanoma. Two classification algorithms, a Random Forest (RF) and XGBoost, were trained and tested. We selected these classifiers due to their efficiency and robustness in handling complex data. The number of melanoma images after the data oversampling is 16.491, and the number of non-melanoma is 32.542. We split the dataset into train and validation sets where the ratio is 70-30, respectively. For the code implementation, we used the python Scikit-Learn library. (Pedregosa, 2011)

To train the Random Forest algorithm we applied Grid Search of Scikit learn library, using different parameters to the RF algorithm to find the best combination of parameter values. We tested the following values to the RF classifier:

Number of estimators: [start=100, stop=2000, num=50)]. The number of trees we want to build before taking the maximum voting or averages of predictions.

Criterion: {"Gini", "Entropy"}, the function to measure the quality of a split. Supported criteria are "gini" for the Gini impurity and "entropy" for the information gain.

Max features: ['auto', 'sqrt'] Auto: This will simply take all the features which make sense in every tree. Here we simply do not put any restrictions on the individual tree. Sqrt: This option will take the square root of the total number of features in the individual run. For instance, if the total number of variables is 100, we can only take 10 of them in the individual tree." log2" is another similar type of option for max_features.

Max depth: [start=5, stop=100, num=10)], the maximum depth of the tree.

Min samples per leaf: [1, 2, 5, 10]. if we have built a decision tree before, we can appreciate the importance of minimum sample leaf size. Leaf is the end node of a decision tree. A smaller leaf makes the model more prone to capturing noise in train data.

Min samples per split: [2, 5, 10, 20], The minimum number of samples required to split an internal node.

Class Weight: [{'balanced'}, {0:0.5,1:1}, {0:1, 1:4}, {0:1, 1:5}, {0:1, 1:6}, {0:1, 1:7}]

To train the XGBoost algorithm we applied Grid Search of Scikit learn library, where we are searching for the best combination of the parameter values of the algorithm.We tested the following parameter values:

gamma: [0,0.1,0.2,0.4,0.8,1.6,3.2,6.4,12.8,25.6,51.2,102.4, 200], Minimum loss reduction required to make a further partition on a leaf node of the tree. The larger gamma is, the more conservative the algorithm will be.

Learning rate': [0.01, 0.03, 0.06, 0.1, 0.15, 0.2, 0.25, 0.300000012, 0.4, 0.5, 0.6, 0.7], Step size shrinkage used in update to prevents overfitting. After each boosting step, we can directly get the weights of new features, and eta shrinks the feature weights to make the boosting process more conservative.

Max depth': [2,3,4,5,6,7,8,9,10,11,12,13,14], Maximum depth of a tree. Increasing this value will make the model more complex and more likely to overfit.

Number of estimators: [50,80,100,115,130,150, 200, 400, 550, 700, 750, 1000, 1500, 2000]

Reg alpha: [0,0.1,0.2,0.4,0.8,1.6,3.2,6.4,12.8,25.6,51.2,102.4,200], L1 regularization term on weights. Increasing this value will make model more conservative. Normalised to number of training examples

Reg lambda: [0,0.1,0.2,0.4,0.8,1,3.2,6.4,12.8,25.6,51.2,102.4,200], L2 regularization term on weights. Increasing this value will make model more conservative. Normalised to number of training examples.

3.7 Python Code Analysis

In this section, we describe the methodology used to create feature vectors and perform classification in the context of lesion analysis. The overall approach consists of three pipelines, each focusing on different aspects of feature extraction and classification. We provide a detailed explanation of each pipeline and the corresponding Python code implementation.

Handcrafted Feature Extraction Pipeline

The first pipeline focuses on extracting handcrafted features from various image processing techniques and color spaces. The goal is to obtain both global features and features specifically from the lesion region. The following steps were performed:

1.Lesion Segmentation using Active Contours:

- Active contours, specifically the technique of snake-based segmentation, were applied to segment the lesion region in the images.

- The resulting segmented lesion masks were used as input to extract features.

- Feature extraction techniques such as Gray-Level Co-occurrence Matrix (GLCM), Local Binary Patterns (LBP), and Color Moments were applied to the lesion region.

2.Lesion Segmentation using UNet:

- The UNet architecture was employed to segment the lesion region in the images.

- The output of the UNet segmentation was used to extract features.

- Similar to the previous step, GLCM, LBP, and Color Moments were computed on the segmented lesion region.

3.Lesion Segmentation using K-Means:

- K-Means clustering was performed on the image pixels to separate the lesion region from the background.

- Features were extracted using GLCM, LBP, and Color Moments from the segmented lesion region obtained from K-Means clustering.

4.Image Filtering with Sobel and Color Spaces:

- The Sobel operator was applied to the images to obtain edge features.

- Additionally, features were extracted from different color spaces, such as RGB, HSV, and LAB, to capture color information.

- Global features were extracted from the entire image using GLCM, LBP, and Color Moments.

The Python code for the handcrafted feature extraction pipeline involves implementing the respective techniques, such as active contours, UNet, K-Means, Sobel filtering, and color space conversions. Additionally, libraries for GLCM, LBP, and Color Moments can be utilized for feature extraction. The code will iterate over the dataset, perform the necessary segmentation and feature extraction operations, and store the resulting feature vectors.

Convolutional Neural Network (CNN) Feature Extraction Pipeline

The second pipeline focuses on extracting features using pre-trained CNN models. Three popular CNN architectures, namely VGG 19, DenseNet121, and EfficientNet-B0, were employed for feature extraction. The following steps were performed:

1. VGG 19 Feature Extraction:

- The pre-trained VGG 19 model was utilized to extract features from the input images.

- The output of the model, typically the fully connected layers or the global average pooling layer, was considered as the feature vector.

2.DenseNet121 Feature Extraction:

- The pre-trained DenseNet121 model was employed to extract features from the input images.

- Similar to the previous step, the output of the model served as the feature vector.

3.EfficientNet-B0 Feature Extraction:

- The pre-trained EfficientNet-B0 model was used to extract features from the input images.

- Once again, the output of the model was considered as the feature vector.

The Python code for the CNN feature extraction pipeline involves loading the pre-trained models, preprocessing the input images according to the respective model's requirements, passing the images through the models, and extracting the features from the desired layers. The code will iterate over the dataset, extract features using each CNN model, and store the resulting feature vectors.

Metadata Collection Pipeline

The third pipeline focuses on collecting metadata information associated with the images. This information can include patient demographics, image acquisition details, and any other relevant information. The metadata is represented as a feature vector.

The Python code for the metadata collection pipeline involves accessing the metadata associated with each image, extracting the required information, and converting it into a feature vector representation.

Classification

The classification step involves combining the extracted feature vectors from the different pipelines to perform lesion analysis. The following combinations were considered:

1. Base Model with CNN Features:

- The feature vectors extracted solely from the CNN models (VGG 19, DenseNet121, EfficientNet-B0) were combined.

2. CNN Features with Handcrafted Features from Lesion Segmentation Techniques:

- The feature vectors obtained from the CNN models were combined with the handcrafted features extracted from the lesion regions segmented using active contours, UNet, and K-Means.

3. CNN Features with Handcrafted Features from Global Image Analysis:

- The feature vectors obtained from the CNN models were combined with the handcrafted features extracted from global image analysis techniques using Sobel filtering and different color spaces.

4. CNN Features with Handcrafted Features from Lesion Segmentation Techniques and Metadata:

- The feature vectors obtained from the CNN models were combined with the handcrafted features extracted from the lesion regions segmented using active contours, UNet, and K-Means.

- Additionally, the metadata feature vector collected from the third pipeline was also included in the combination.

The Python code for the classification step involves concatenating the feature vectors obtained from the different pipelines according to the desired combinations. Machine learning algorithms, such as support vector machines (SVM), random forests, or deep neural networks, trained on the combined feature vectors for lesion classification.

By implementing the described pipelines and classification combinations in Python, the complete workflow for lesion analysis can be realized, incorporating both handcrafted and deep learning-based features for improved classification performance.

4 Experiments and Results

This section contains all of the experiments, tests, and results used in this study. Evaluation metrics are included for each of the different methods of segmenting, extracting, and training features. Finally, the various trained model combinations are evaluated and compared.

4.1 Dataset

4.1.1 The ISIC 2020 Archive

The International Skin Imaging Collaboration (ISIC) (ISIC, 2020) collaborates between academics and industry that aims to make it easier to use digital skin imaging to help reduce melanoma lesions. Clinical decision support, and automated diagnosis, digital photographs of skin lesions may be utilized to educate professionals and the general public about melanoma detection and directly help diagnose melanoma. ISIC seeks to achieve its objectives by developing and promoting digital skin imaging standards and collaborating with the dermatological and computer science communities to enhance diagnosis. (Haofu Liao, 2015) (Li & Shen, 2018) (Rezvantalab, Safigholi, & Karimijeshni, 2018) Currently, a lack of standards for dermatologic imaging undermines the quality and usefulness of skin lesion imaging.

The ISIC makes publicly accessible new labeled skin images that supplement prior years' datasets, resulting in a considerable increase in the ISIC archive's total quantity over time, making it the most significant skin lesion image collection. (Rotemberg, et al., 2021) The ISIC 2020 challenge contain dermoscopy images from the Hospital Clínic de Barcelona, Medical University of Vienna, Memorial Sloan Kettering Cancer Center, Melanoma Institute Australia, University of Queensland, and the University of Athens Medical School. Over 2,000 patients contributed to the ISIC2020 dataset, including 32,542 benign and 584 malignant skin lesions. Each image's metadata includes the lesion's diagnosis and location, the patient's estimated age, gender, and an anonymized patient identification number that allows lesions from the same patient to be mapped together. The benign images are subdivided into the following types (nevus, seborrheic keratosis, lichenoid keratosis, solar lentigo, lentigo NOS, cafe-au-lait macule, atypical melanocytic proliferation, and unknown). In contrast, the malignant images are all diagnosed as melanoma. Notably, there are no basal cell or squamous cell carcinoma incidences in the dataset, making the problem a binary classification problem of melanoma detection. There is an ISIC dataset of 10,982 unlabeled photos with their metadata also provided for assessment purposes. The Figure 26 represents some benign and malignant examples of ISIC 2020 dataset.

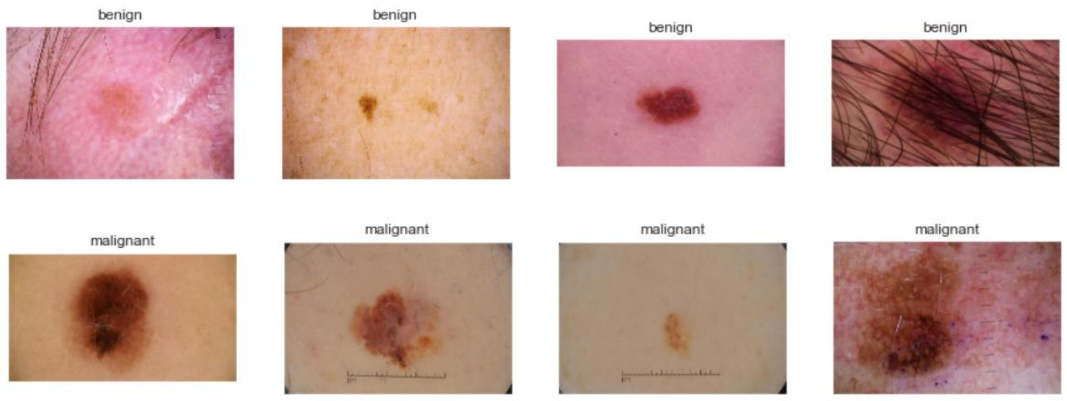


Figure 26: Non melanoma top, melanoma bottom

4.1.2 Discover the dataset

In the ISIC 2020 dataset, 79.2 % of patients had no melanoma, whereas 20.8 % had at least one, with 16 lesions per patient. Only 584 (1.8 %) of the 33,126 dermoscopic pictures were histopathologically confirmed melanoma. Each training image has an average of 12.743.090 pixels, ranging from 307.200 to 24.000.000 pixels. The patient's estimated age at the time of picture collection, gender, general anatomic location of the lesion, patient identification number (patient ID), benign/malignant type, and specific diagnosis (if one was available) were all included in the metadata for each image. The age and gender distribution are represented in the following Figure 27, where it seems that the gender is nearly divided, and most patients are in the 40-60 age group.

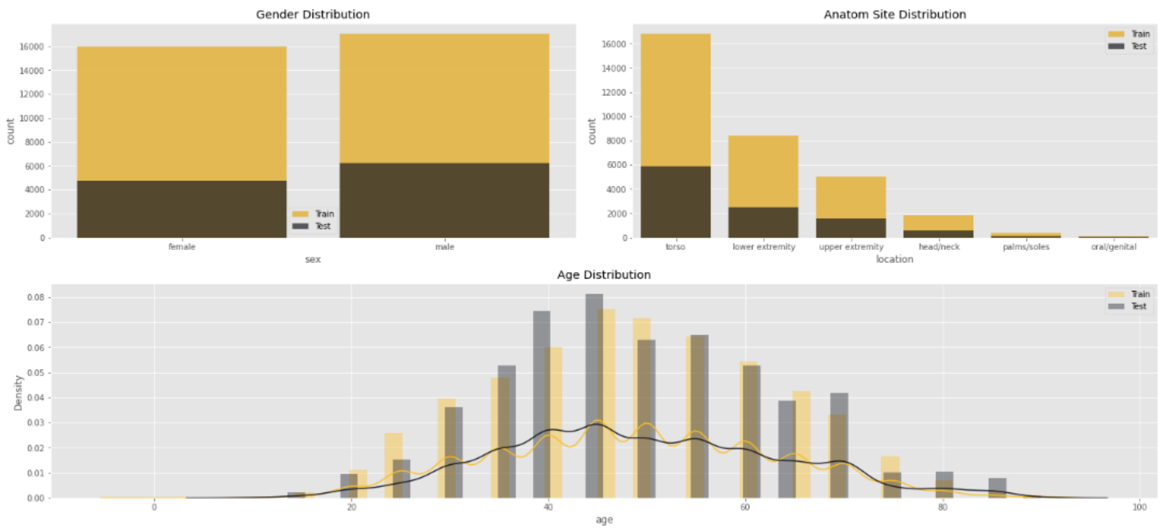


Figure 27: Gender, Anatomy Site and Age distribution of ISIC 2020 dataset

The Figure 28 shows that in the training set, most lesions are located in the torso (52%), the lower extremities have fewer (26%), and the upper extremity and head have even fewer (15% and 6% respectively).

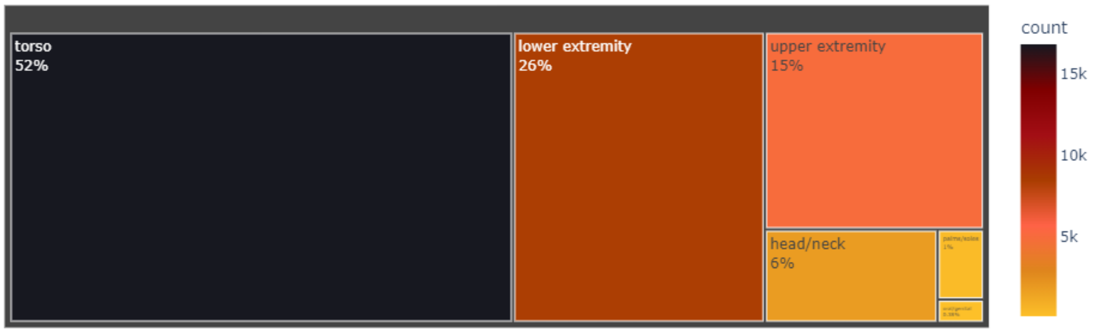


Figure 28: Scans by Anatom Site in Train Data

Even though the body parts with the most lesions are the torso, the lower and the upper extremities, as it is described in Figure 29 the body parts with the most malignant lesions are located in the head, oral/genital, and upper extremity.

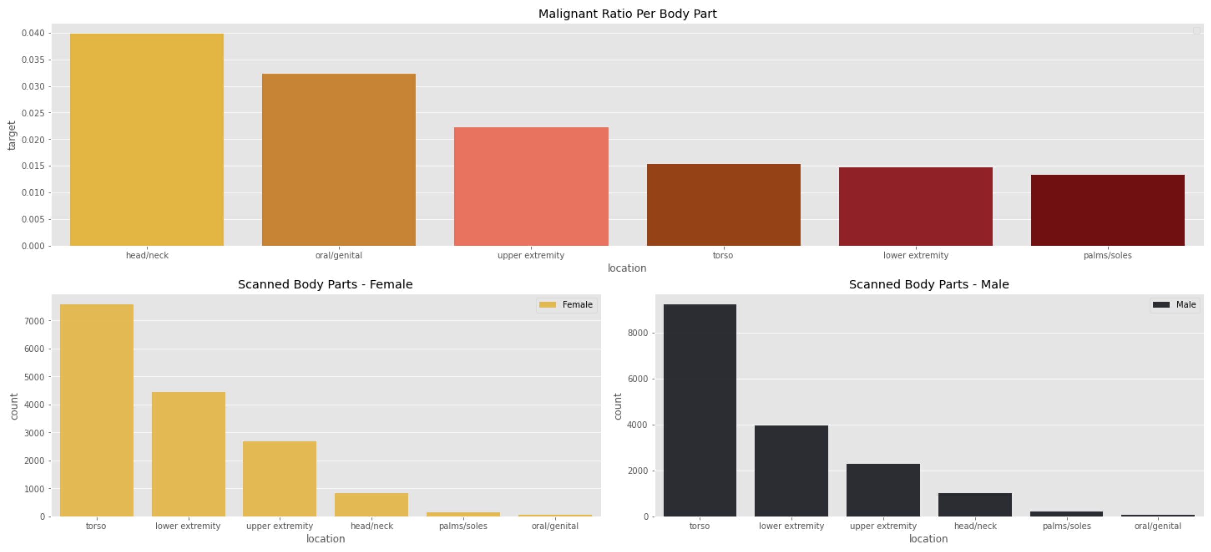


Figure 29: Malignant Ration per Body part and per sex

Additionally, males have a 62 % dominance in malignant images, while the male-to-female ratio is 52 to 48%. In addition, the location of the malignant lesion varies according to the gender of the patient. Although the torso is the most common lesion location in both men and women, it accounts for almost half of all lesions in men and 39 % in women. Female examples of the lower extremities are more prevalent than male lesions. Males account for 18%, while females account for 26% of the set. Finally, Females are more likely than males to have upper extremities malignant lesions (23% - 17 %).

4.1.3 Class Distribution

The term imbalanced dataset means that one class (named the minority group) contains significantly fewer samples than the other (named the majority group). Classification results will be better if the dataset is balanced across classes. In this instance, the classifier will concentrate on learning majority classes while ignoring minority classes partially or fully. Because melanoma is such a small minority group (1.76 % or 18:1000), there is a big ratio of imbalance, which can cause a biased classifier, lowering its ability to predict correct the melanoma class. The following Figure 30 represents the class distribution of the dataset, where the unique malignant class is the melanoma class, while the other classes are benign categories.

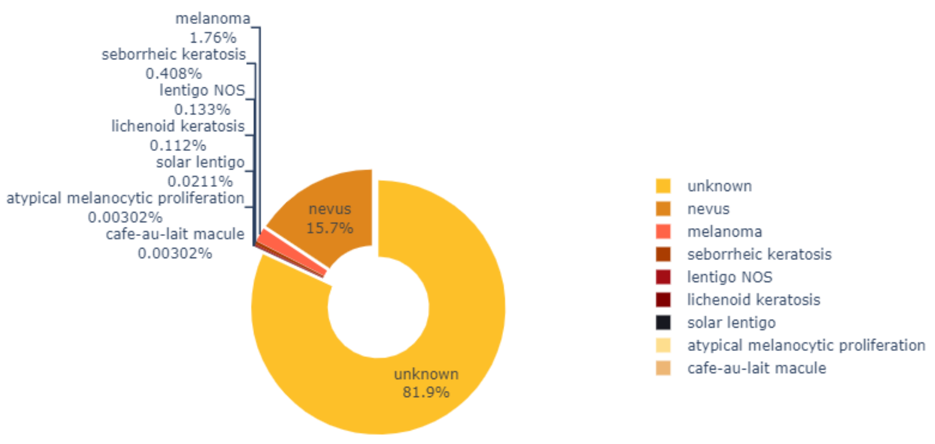


Figure 30: Class distribution of the ISIC dataset

4.2 Evaluation Methods

4.2.1 Confusion Matrix

The confusion matrix is a metric for evaluating the outcomes of classification. We have two classes, the positive and the negative class:

• True Positive: the number of samples that belong to the positive class and were correctly classified as positive (TP)

• Number of samples in the negative class that were correctly recognized as negative - True Negative (TN)

• The number of samples that should have been classified as positive but were incorrectly classified as negative - False Positives (FP)

• The number of samples that should have been classified as negative but were incorrectly classified as positive - False Negative (FN)

The classification accuracy is then determined as follows:

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

Precision is the percentage of positive examples that were predicted correctly:

$$Precision = \frac{TP}{TP + FP}$$

Sensitivity/Recall is the proportion of actual positive cases which are correctly identified:

$$Recall = \frac{TP}{TP + FN}$$

Specificity is the proportion of actual negative cases which are correctly identified:

$$Specificity = \frac{TN}{TN + FP}$$

4.2.2 ROC Curve and Area Under Curve (AUC)

ROC (receiver operating characteristic) A Curve is a common graph for measuring and evaluating a classifier's performance. Given what we have discussed thus far, if we train a classifier to identify a dataset with two classes, we can easily measure the output and compare it to the ground truth to generate the confusion matrix. As a result, we get the original and anticipated classes for each sample, which populate the TP, TN, FP, and FN values. Then, using the mathematical types provided above, we can obtain the classification's true positive and false-positive rates. (M & M.N, 2015) A ROC curve is a two-dimensional graph with the x-axis representing the true positive rate and the y-axis representing the false positive rate. The curve is the classifier's result, while the y=x axis is the result of a random classifier that classifies the dataset with a 50% chance for each class. The greater the distance between the diagonal axis and the higher the curve, the better the categorization. The 'Area Under Curve' is the area specified under the curve used to determine the correctness of the categorization (AUC).

4.3 Base Models Results

We needed a base model to start the comparison and to see if the different experimented techniques had a positive effect on the model. To study and evaluate the impact of each different approach, we first trained three base models that will serve as a basis for comparing different techniques. We extracted features from three pre-trained neural networks, VGG19, Densenet121, and EfficientNetB0, as inputs into two different classifiers, Radom Forrest (RF), and XGBoost, for this purpose. Based on the grid search, the Random Forest trained with number of estimators: 2000, criterion: gini, max features: auto, max depth:20, minimum samples per split:10, class weight={'balanced'}, minimum samples per leaf:5, random state: 40.

The XGBoost based on the grid search we applied, it was trained with gamma: 0.1, learning rate: 0.01, max depth: 10, number of estimators: 2000, L1 regularization parameter alpha: 0.1, L2 regularization parameter lambda: 1. The Table 3 and Table 4 contain the experimental results of the base models of the Random Forest and the XGBoost classifiers respectively.

Table 3 Results of the base CNN models and Random Forest classifier

Feature Extractor (trained with random forest)	Accuracy	Precision	Recall	F1	Specificity	AUC
VGG19	0,8183	0,7954	0,7992	0,7972	0,857	0,7991
DenseNet121	0,722	0,6918	0,7006	0,6961	0,765	0,8195

EfficientNetB0	**0,8314**	**0,8106**	**0,8342**	**0,8222**	**0,826**	**0,8341**

Table 4 Results of the base CNN models and XGBoost classifier

Feature Extractor (trained with XGBoost)	Accuracy	Precision	Recall	F1	Specificity	AUC
VGG19	0,867	0,8632	0,832	0,8471	0,938	0,832
DenseNet121	0,8446	0,8357	0,8067	0,8209	0,921	0,8446
EfficientNetB0	**0,908**	**0,9054**	**0,8854**	**0,8952**	**0,954**	**0,908**

Although the results indicate that some base feature extraction methods could be a good candidate for this melanoma classification problem, melanoma classification is a crucial medical issue, and we want to get the most accurate result is possible. So far, the EfficientNetB0 appears to be the most valuable model, which evaluated in Random Forest classification results with 83,14% accuracy, 81,06% precision, 83,42% recall, 82,22% F1-score, 82,6% specificity and 83,41% AUC. The XGBoost evaluation results are 90,8% accuracy, 90,54% precision, 88,54% recall, 89,52% F1-score, 95,4% specificity and 90,8% AUC.

4.2 Combination of CNN Features and Handcrafted Features

Receiving the base model results, we want to observe if the combination of the vectors that collect the CNN with handcrafted improves the training performance. We introduced a new vector derived from the concatenation of CNN and handcrafted features, which used as the input for machine learning classifiers. The Table 5 and Table 6 show these results for Random Forest and XGBoost classifiers respectively.

Table 5 : Results of the combined CNN and Hardcrafted features and Random Forest classifier

Segmentation Technique	Feature Extractor (trained with Random Forest)	Accuracy	Precision	Recall	F1	Specificity	AUC
Chan Vese	VGG19+HC[1]	0.824	0.801	0.807	0.804	0.859	0.807
	DenseNet121+HC	0,635	0,660	0,677	0,668	0,551	0,672
	EfficientNetB0+HC	0,839	0,818	0,834	0,826	0,852	0,834
Sobel	VGG19+HC	0.840	0.819	0.832	0.825	0.858	0.832
	DenseNet121 +HC	0,840	0,819	0,832	0,825	0,858	0,832
	EfficientNetB0 +HC	0,793	0,775	0,803	0,788	0,774	0,803
K_means	VGG19 +HC	0,823	0,801	0,803	0,802	0,865	0,803
	DenseNet121 +HC	0,708	0,704	0,729	0,716	0,665	0,729
	EfficientNetB0+HC	0,852	0,832	0,855	0,843	0,846	0,855
Unet	VGG19 +HC	0,854	0,834	0,859	0,846	0,845	0,859
	DenseNet121+HC	0,708	0,704	0,729	0,716	0,665	0,729
	EfficientNetB0+HC	0,803	0,785	0,813	0,793	0,784	0,813

[1] HC: GLCM + LBP + Color Moments

Color Spaces	VGG19+HC	0,635	0,660	0,677	0,668	0,551	0,672
	DenseNet121+HC	0,707	0,694	0,716	0,704	0,691	0,716
	EfficientNetB0+HC	**0,839**	**0,818**	**0,834**	**0,826**	**0,852**	**0,843**

Table 6 Results of the combined CNN and Handcrafted features and XGBoost classifier

Segmentation Technique	Feature Extractor (trained with XG Boost)	Accuracy	Precision	Recall	F1	Specificity	AUC
Chan Vese	VGG19 + HC	0,928	0,927	0,909	0,917	0,966	0,909
	DenseNet121 + HC	0,844	0,835	0,806	0,82	0,921	0,806
	EfficientNetB0+HC	0,924	0,923	0,904	0,913	0,964	0,904
Sobel	VGG19 + HC	0,944	0,941	0,933	0,937	0,967	0,933
	DenseNet121 +HC	0,927	0,920	0,916	0,918	0,951	0,916
	EfficientNetB0+HC	0,946	0,943	0,934	0,938	0,97	0,934
K_means	VGG19+HC	0,928	0,927	0,909	0,917	0,966	0,909
	DenseNet121+HC	0,844	0,835	0,807	0,82	0,921	0,807
	EfficientNetB0+HC	0,924	0,923	0,904	0,913	0,964	0,904
Unet	VGG19+HC	0,933	0,934	0,915	0,924	0,971	0,915
	DenseNet121+HC	0,898	0,891	0,879	0,894	0,938	0,879
	EfficientNetB0+HC	0,945	0,946	0,931	0,938	0,975	0,931
Color Spaces	VGG19+HC	0.947	0.943	0.937	0.94	0.966	0.937
	DenseNet121+HC	0,921	0,912	0,910	0,911	0,944	0,910
	EfficientNetB0+HC	**0,937**	**0,931**	**0,927**	**0,929**	**0,958**	**0,927**

The above results show that combining handcrafted features with the CNNs feature vectors helps to improve classification and increases the accuracy of the final models. By comparing the classification results using the evaluation metrics, we can observe that the comparison of the EfficientNetB0 features with color GLCM, color LBP and color moment features produces better results. The best evaluation results of Random Forest are 83,9% accuracy, 81,8% precision, 83,4% recall, 82,6% F1-score, 85,2% specificity and 84,3% AUC. The XGBoost best evaluation results are 93,7% accuracy, 93,1% precision, 92,7% recall, 92,9% F1-score, 95,8% specificity and 92,7% AUC.

4.3 Combination of CNN Features and Handcrafted Features and metadata

In the final experiments we add even more features to the classifier in order to see whether accuracy can be improved even further. We combined three feature vectors, the CNN feature vector, the handcrafted feature vector, and the metadata feature vector. The Table 7 and Table 8 show these results for Random Forest and XGBoost classifiers respectively.

Table 7 Results of the combined CNN and Handcrafted and metadata features and Random Forest classifier

Segmentation Technique	Feature Extractor	Accuracy	Precision	Recall	F1	Specificity	AUC

	(trained with Random Forest)						
Chan Vese	VGG19 + HC+ m[2]	0,803	0,785	0,813	0,798	0,784	0,813
	DenseNet121+HC+m	0,707	0,694	0,716	0,704	0,691	0,716
	EfficientNetB0 + m	0,846	0,825	0,845	0,834	0,839	0,849
Sobel	VGG19 + HC + m	0,793	0,775	0,803	0,788	0,774	0,803
	DenseNet121+HC+m	0,708	0,704	0,729	0,716	0,665	0,729
	EfficientNetB0+HC+m	0,854	0,834	0,859	0,846	0,845	0,859
K_means	VGG19+HC+m	0,803	0,785	0,813	0,798	0,784	0,813
	DenseNet121+HC+m	0,707	0,694	0,716	0,704	0,691	0,716
	EfficientNetB0+HC+m	0,839	0,818	0,834	0,826	0,852	0,843
Unet	VGG19+HC+m	0,829	0,808	0,811	0,809	0,867	0,811
	DenseNet121+HC+m	0,692	0,687	0,710	0,698	0,657	0,710
	EfficientNetB0+HC+m	0,838	0,854	0,867	0,86	0,884	0,867
Color Spaces	VGG19+HC +m	0,842	0,821	0,845	0,832	0,836	0,845
	DenseNet121+HC+m	0,854	0,834	0,857	0,845	0,848	0,857
	EfficientNetB0+HC+m	**0,861**	**0,842**	**0,868**	**0,854**	**0,848**	**0,868**

Table 8 Results of the combined CNN and Handcrafted and metadata features and XGBoost classifier

Segmentation	Feature Extractor (trained with XG Boost)	Accuracy	Precision	Recall	F1	Specificity	AUC
Chan Vese	VGG19 + HC+ m	0,927	0,926	0,908	0,916	0,965	0,908
	DenseNet121+HC+m	0,863	0,853	0,833	0,842	0,924	0,833
	EfficientNetB0 + m	0,927	0,927	0,907	0,916	0,968	0,907
Sobel	VGG19 + HC + m	0,943	0,940	0,932	0,936	0,966	0,932
	DenseNet121+HC+m	0,927	0,921	0,915	0,918	0,952	0,915
	EfficientNetB0+HC+m	0,947	0,945	0,936	0,940	0,971	0,936
K_means	VGG19+HC+m	0,927	0,926	0,908	0,916	0,965	0,908
	DenseNet121+HC+m	0,862	0,852	0,833	0,841	0,922	0,833
	EfficientNetB0+HC+m	0,928	0,928	0,908	0,917	0,968	0,908
Unet	VGG19+HC+m	0,939	0,940	0,922	0,930	0,974	0,922
	DenseNet121+HC+m	0,904	0,899	0,883	0,89	0,947	0,883
	EfficientNetB0+HC+m	0,942	0,944	0,926	0,934	0,975	0,926
Color Features	VGG19+HC +m	0,946	0,941	0,937	0,939	0,965	0,937
	DenseNet121+HC+m	0,923	0,915	0,911	0,913	0,947	0,911
	EfficientNetB0+HC+m	**0,949**	**0,946**	**0,939**	**0,942**	**0,969**	**0,939**

The above results validate the study's approach of combining CNN, handcrafted, and metadata features to improve base model accuracy. Combining the EfficientNetB0 features, the GLCM, LBP and Color Moment features, and the metadata features produced the best results in our study, for both different machine learning classifiers (Random Forest and

[1] HC: GLCM + LBP + Color Moments [2] m: Metadata

XGBoost). The results were evaluated, and the Random Forest achieved accuracy 86,17%, precision 84,22%, recall 86,85%, F1 score 85,4%, specificity 84,8% and AUC 86,85%. The XGBoost classifier evaluated with accuracy 94,94%, precision 94,6%, recall 93,98%, F1 score 94,28%, Specificity 96,9% and AUC 93,97%.

4.4 Comparison with State-of-the-Art approaches

Because several measurements can be used to evaluate the classification, and different metrics can verify each study, it is not easy to draw a clear conclusion about which approach will produce better results. Additionally, the dataset to which each experiment was applied can significantly impact the outcomes. However, the results we verify in our approach are among the best for melanoma classification in the studies presented in the Table 9 there is still room for improvement, especially when it comes to medical issues where even small improvements in accuracy are important. However, these preliminary findings show that by combining deep learning features with handcrafted features and metadata features, we can produce high-performance classifiers for melanoma classification, avoiding the use of complicated deep neural network architectures with millions of parameters and high costs.

Table 9 Comparison with State-of-the-Art approaches

	Dataset Size	Accuracy	Sensitivity	Specificity	AUC
Xie et.al	240	0.9417	0.95	93.75	
Tomas Majner	1.279	0.826	0.533	0.898	0.78
Esfahani et.al	6120	0.81	0.81	0.80	
Codella et.al	2.150	0.931	0.926		
Han et.al	100.000	0.875			0.95
Jason R. Hajerty etal	2.594				0.94
Kaggle leaderboard	58.457				0.9404

4.5 Visualization

The visualization contains results about hair removal, CLAHE equalization, chan vesse active contour segmentation, sobel operator segmentation, k-means segmentation, UNet segmentation and color space segmentation. It also includes visualizations of Grey World transformation on color spaces and max color constancy transformation on color spaces. The Figure 31 shows the steps to remove hair in an image, using CIELuv color space. In the (a) image we show the original RGB lesion image, in the (b) we show the CIE Luv lesion image, and in (c), the hair mask in the L channel using the hysteresis thresholding. Then the next step (d) shows the CIE Luv image after the hair removing and then (e) the returning to the RGB channels. The last image (f) shows the results of CLAHE equalization, which supports the image contrast. The Figures 32,33,34,35,36 represent the segmentation results of the implemented methods. In the Figure 32 we show the segmentation results of the active contours method. However, in the first row we can observe that the segmentation mask covers regions of the lesion, which means that we lose information about this part of the lesion. The Figure 33 shows the results of the sobel operator segmentation technique. In the last raw of images, we show how sensitive to image artifacts is this method, which create some noise to the image. The Figure 34 shows the segmentation results of the k-means. However, in the second row of this figure we observe the weakness of this method where the segmentation mask leaves out along with the lesion a part of the skin which is chromatically close to the lesion. The Figure 35 represents the predicted results of the UNet model, with the segmentation mask and the lesion segmentation of a skin lesion image. Furthermore, the Figure 36 represents the segmentation results using the (a) HSV, (b) CIE Luv, (c) YCrCb and the (d) CIE Lab color spaces. The results show that the color spaces smooth out low color intensity differences, and instead they focus on high color intensity changes, which in this case are the skin and the lesions. The Figures 37 and 38 represent the Grey-Word transformation on color spaces and the max-color constancy transformation on color spaces respectively. More specifically, the Figure 37 contains the (a) original RGB skin lesion image, (b) the grey word RGB, (c) the grey word HSV, (d) the grey word Lab, (e) the grey word YCrCb, and (f) the grey word Luv transformations. The Figure 38 includes the (a) original RGB skin lesion image, (b) the max color constancy RGB, (c) the max color constancy HSV, (d) the max color constancy Lab, (e) the max color constancy YCrCb, and (f) the max color constancy Luv transformations. Those transformations were used to extract color moments in the handcrafted feature extraction process.

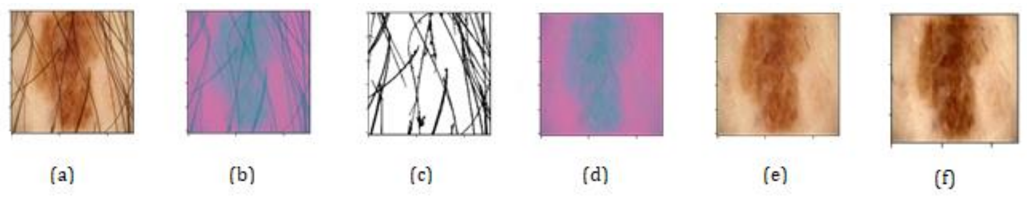


Figure 31: Hair removal using CIE Luv color space and CLAHE equalization: (a) original RGB image, (b) CIE Luv image, (c) hair mask, (d) CIE Luv image after hair removal, (e) RGB image after hair removal (f) CLAHE equalization

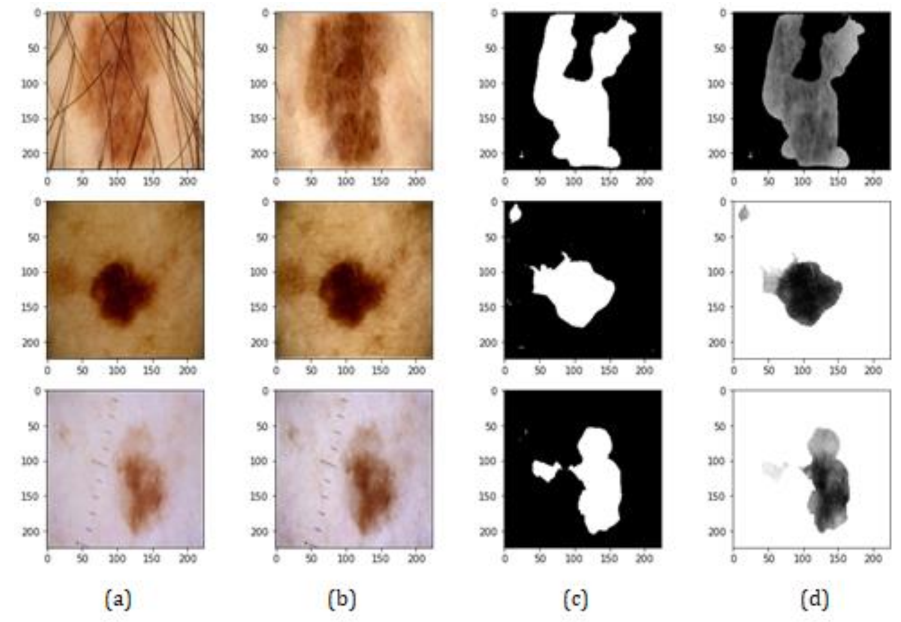


Figure 32: Chan-Vesse Active Contours segmentation process: (a) original RGB image, (b) RGB image after hair removal, (c) segmented mask, (d) lesion segmentation

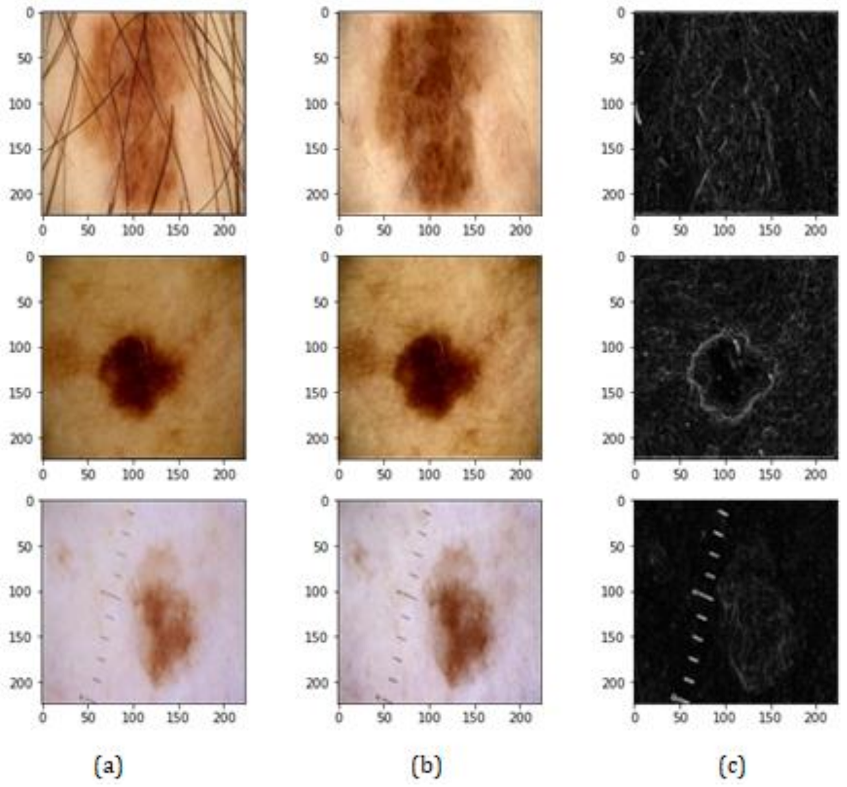


Figure 33: Sobel based segmentation process: (a) original RGB image, (b) RGB image after hair removal, (d) segmented lesion

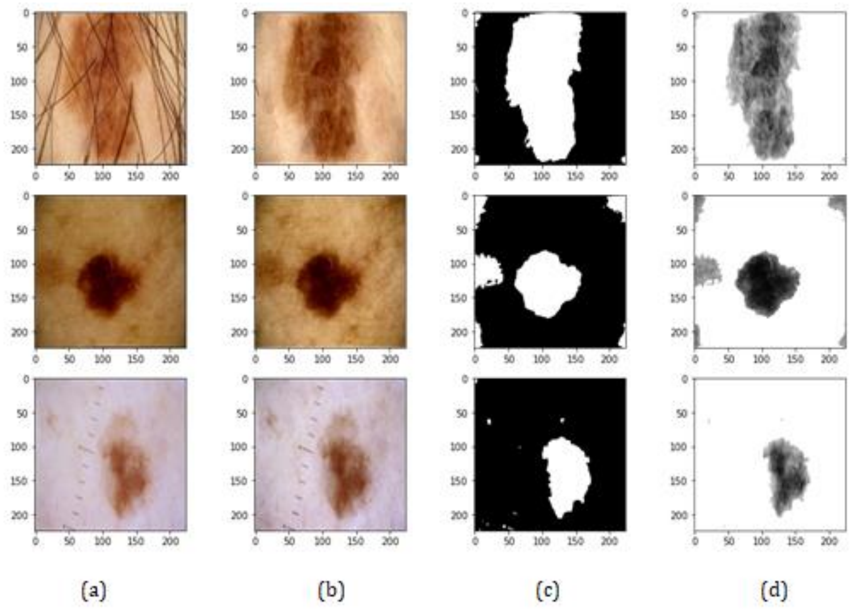


Figure 34: K-means based segmentation process: (a) original RGB image, (b) RGB image after hair removal, (c) segmented mask, (d) segmented lesion

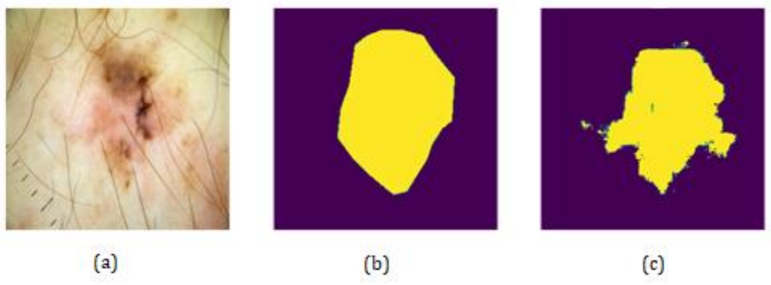


Figure 35: Unet based segmentation process: (a) input RGB image, (b) True mask, (c) predicted mask

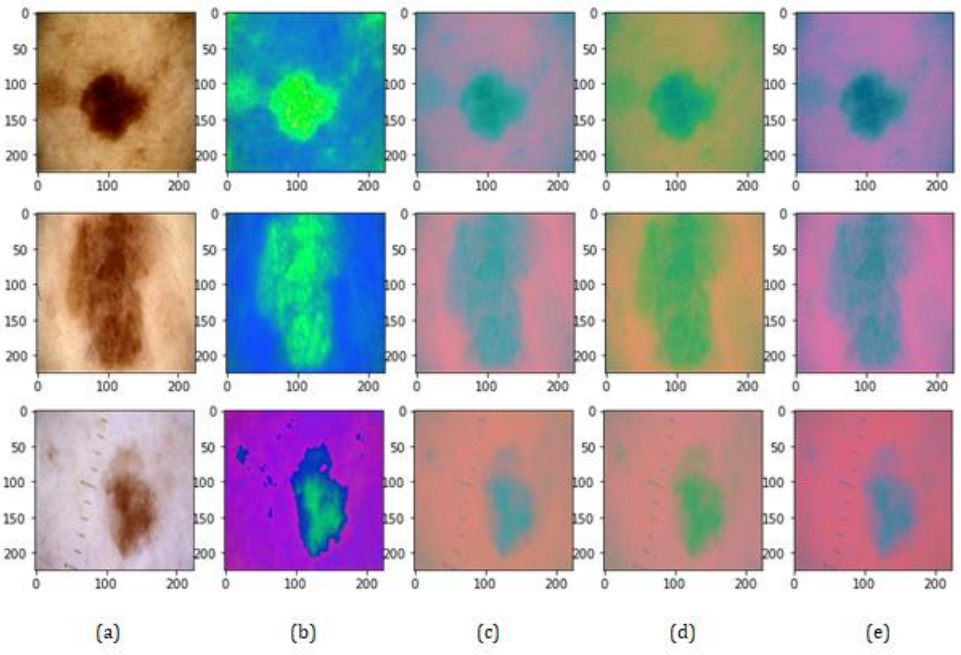


Figure 36: Color space based segmentation process: (a) original RGB image, (b) HSV color space segmentation, (c) CIE Lab color space segmentation, (d) YCrCb color space segmentation, (e) CIE Luv color space segmentation

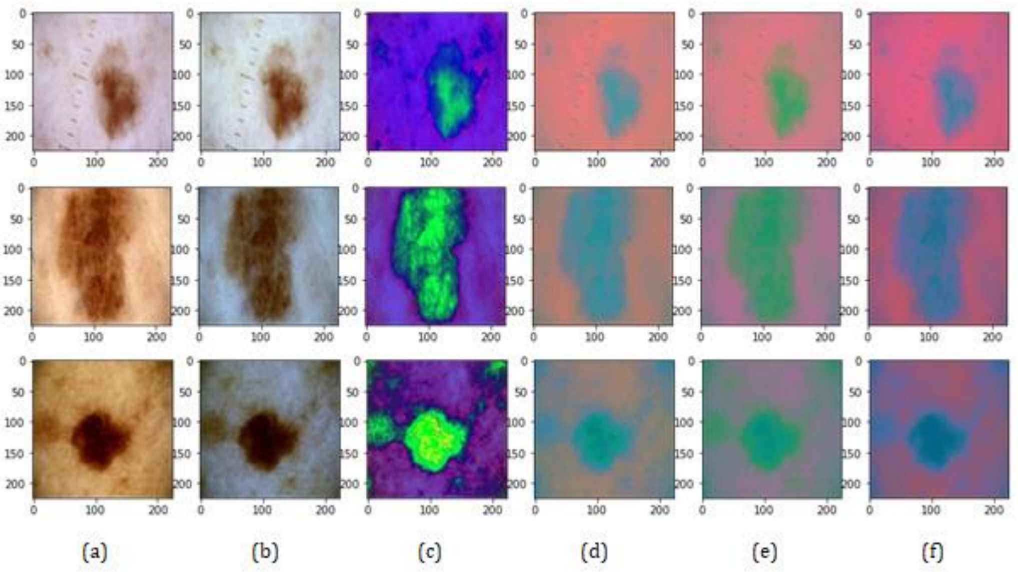


Figure 37 : Grey World Transformation on Color spaces: (a) original RGB image, (b) gwRGB, (c) gwHSV, (d) gwLab, (e) gwYCrCb, (f) gwLuv

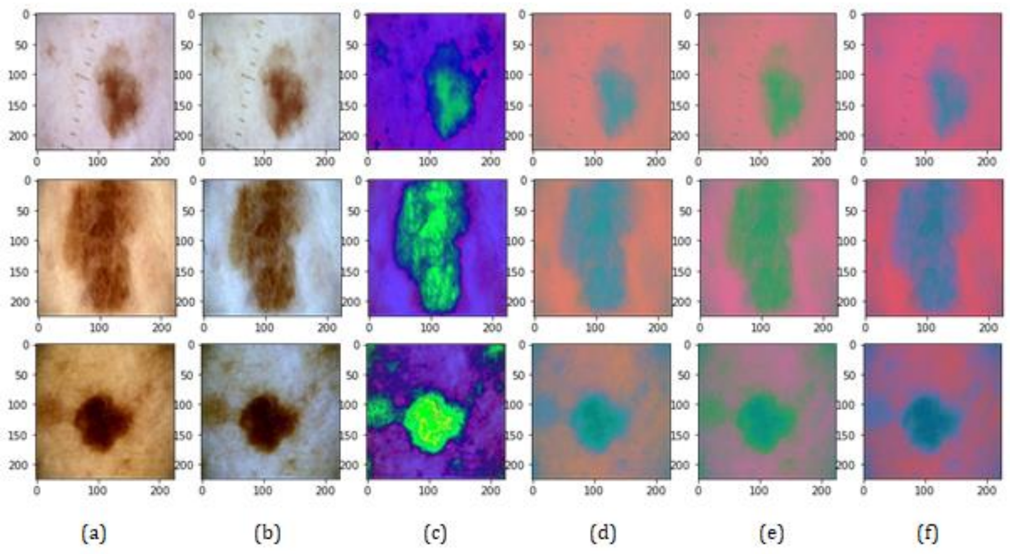


Figure 38: Max Color Constancy image transformation in different color spaces: (a) original RGB image, (b) mxccRGB, (c) mxccHSV, (d) mxccLab, (e) mxccYCrCb, (f) mxccLuv)

5 Discussion

It is crucial to mention that various state-of-the-art neural networks were used to diagnose melanoma for this thesis. However, it should be noted that these neural networks contained pre-trained weights from a different dataset, the Imagenet. This dataset was not created to identify skin lesions but trains, dogs, airplanes, and much more. Gathering data from skin lesions to identify benign or malignant skin lesions is a much more complicated process due to the lack of specific features that make a lesion malignant. This is why dermatologists cannot diagnose with sufficient accuracy whether a skin lesion is a melanoma or not, achieving 59% accuracy, a low percentage for such a significant medical issue. (Kittler, Pehamberger, Wolff, & Binder, 2002) (Marchetti, et al., 2018)

As the discussion progresses, the dataset is our primary source of knowledge. Improved feature extraction techniques can lead to very high learning and accuracy rates. We used a small to medium-sized dataset for training a Machine Learning model. Also, we face the class imbalance of the ratio of 1/60 as the examples of melanoma are only 5106, 60 times less than non-melanoma. These issues have a direct impact on the results. The data augmentation method certainly improves final results and up-sampling and down-sampling techniques. Moreover, the concatenation of feature vectors extracted from Convolutional Neural Networks (CNN) with handcrafted and metadata has shown that they can significantly improve base models that contain only features of CNNs.

More specifically, we saw that the features of the Convolutional Neural Networks were extracted from a pre-trained network, had learned at a level to recognize different categories and then they called upon to collect data from skin lesions. According to the architecture of each deep neural network, we conclude that some neural networks are better candidates to detect specific features achieving better accuracy rates. We used three different CNN architectures, the VGG19, the DenseNet121, and the EfficientNet-B0 and each one formed a base model. Best accuracy is performed by the EfficientNet-B0, where 1280 features were extracted, and they became the input for the Random Forest and XGBoost classifiers.

We also observed the best segmentation technique with the color spaces (HSV, Luv, Lab, YCrCb), and we extracted features from each color channel with color GLCM, color LBP, and Color Moments. Combining these features with the CNN features proved that they significantly improved the accuracy of the models. This combination became input to the XGBoost, improving the performance of the base model and achieving an accuracy of 93.73%, precision 93,14%, recall 92,71%, F1-score 92,92%, sensitivity 89,6%, specificity 95,8%, and AUC 92,71%.

The further extension of the feature vectors consists of patient metadata (age, location, gender). The combination of EfficientNet-B0, color GLCM, color LBP, Color Moments, and metadata features which fed the XGBoost classifier achieving accuracy levels of 94,94%, precision 94,6%, recall 93,98%, F1-score 94,98%, sensitivity 91,1%, specificity 96,9% and AUC 93,97%.

6 Conclusion and Future work

The combination of CNN features, handcrafted features, and metadata features as the input for the XGBoost classifier achieves very high classification performance in detecting melanoma from dermoscopy images, with the best model to compete for top performance, achieving the ROC-AUC rating of 94.04 percent.

These results are promising and suggest that approaches based on combining Deep-Learning, handcrafted, and metadata features have high potential in melanoma recognition, and this model might be a dermoscopy image-based automated melanoma classification system that expert dermatologists could use to support their diagnosed decisions. The findings revealed that using data augmentation to address the problem of class imbalance had a positive effect. Various CNN models were utilized as base models, including VGG19, DenseNet121, and EfficientNet-B0, to research if extending the CNN feature vector with handcrafted and metadata features can increase the performance of the base models. These results are promising and suggest that approaches based on combining Deep learning, handcrafted, and metadata features have high potential in melanoma recognition.

In the future, we will concentrate on using the models EfficientNetB1-B7 as feature extractors, scaling the width, height, and resolution, and using fewer parameters with purpose an even better accuracy. Also, since the dataset size used in this thesis is medium, we will use different datasets to test the accuracy of our proposed model. Finally, the lack of proper explainability of the predictive models still prevents the widespread use of clinical trials. We intend to explore the development of develop an explainability method to visualize the most indicative areas of melanoma. In this way, we will link the visual stimulus to the predictive model output so that the model becomes more transparent and reliable, providing interpretability of the model prediction in a way that is understandable to dermatologists.

Bibliography

A. Esteva, B. K. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature, vol. 542, pp. 115–118, doi: 10.1038/nature21056*.

A. Rezvantalab, H. S. (2018). Dermatologist Level Dermoscopy Skin Cancer Classification Using Different Deep Learning Convolutional Neural . *Networks Algorithms," arXiv:1810.10348*.

A.J.Vyavahare, & Mahajan, P. R. (2013). Artefact Removal and Contrast Enhancement for Dermoscopic Images Using Image Processing Techniques. *International Journal of Innovative Research in Electrical, Electronics, Instrumental and Control Engineering, vol. 1, no. 9*.

Abbasi, N., Shaw, H., Rigel, D., Friedman, R., McCarthy, W., Osman, I., . . . Polsky, D. (2004, 12). Early Diagnosis of Cutaneous Melanoma. *JAMA, 292*(22), 2771.

Abhilash, K. K. (2021). *Adaptive Active Contour Segmentation for Melanoma Diagnosis from Dermoscopy Images. In: Dash, S.S., Panigrahi, B.K., Das, S. (eds) Sixth International Conference on Intelligent Computing and Applications . Advances in Intelligent Systems and Computing,.* vol 1369. Springer, Singapore. https://doi.org/10.1007/978-981-16-1335-7_12.

Agarap, A. F. (2018). . Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375.

Ali, K., Shaikh, Z., Khan, A., & Laghari, A. (2022, 12). Multiclass skin cancer classification using EfficientNets – a first step towards preventing skin cancer. *Neuroscience Informatics, 2*(4), 100034.

Bakheet, S. (2017). An SVM Framework for Malignant Melanoma Detection Based on Optimized HOG Feature. *Computation, vol. 5, no. 1, pp. 4, doi: 10.3390/computation5010004*.

Baltruschat, I. M., Nickisch, H., Grass, M., Knopp, T., & Saalbach, A. (2019). Comparison of deep learning approaches for multi-label chest X-ray classification. Scientific reports, 9(1), 1-10.

Barata, C., Marques, J., & Celebi, M. (2014, 10). Improving dermoscopy image analysis using color constancy. *2014 IEEE International Conference on Image Processing (ICIP)*, 3527-3531.

Beaumont. (2021). Skin Cancer | ABCDE Assessment for Melanoma |.

Benčo, M., & Hudec, R. (2007). *Novel Method for Color Textures Features Extraction Based on GLCM.*

Bezerra, L., Maia, A. C., Thiago, P., Santos, C., Lima, N. d., Serra, H. O., . . . Paiva, A. C. (2018). Evaluation of Melanoma Diagnosis using Imbalanced Learning. DOI: https://doi.org/10.5753/sbcas.2018.3680.

Bora, D., Kumar Gupta, A., & Khan, F. (2008). *International Journal of Emerging Technology and Advanced Engineering Comparing the Performance of L*A*B* and HSV Color Spaces with Respect to Color Image Segmentation.* Retrieved from www.ijetae.com

Bordallo, L. M. (2016). *European Association for Signal Processing.* Institute of Electrical and Electronics Engineers Sixth International Conference on Image Processing Theory, Tools and Applications : IPTA 2016, Oulu, Finland, December 2016: IEEE Finland Section.

Bradski, G. (2000). website, available on: https://docs.opencv.org/3.4/d2/d2c/tutorial_sobel_derivatives.html. *The OpenCV Library. Dr. Dobb&#x27;s Journal of Software Tools.*

Brynolfsson, P., Nilsson, D., Torheim, T., Asklund, T., Karlsson, C., Trygg, J., . . . Garpebring, A. (2017, 12). Haralick texture features from apparent diffusion coefficient (ADC) MRI images depend on imaging and pre-processing parameters. *Scientific Reports, 7*(1).

Celebi, N. M. (2016). An Overview of Melanoma Detection in Dermoscopy. *Images Using Image Processing and Machine Learning," arXiv:1601.07843*.

Chai, T. &. (2014).). Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature. Geoscientific model development, 7(3), 1247-1250.

Chan, T., & Vese, L. (2001). *Active Contours Without Edges.*

Chang, J. R. (2015). Batch-normalized maxout network in network. . *arXiv preprint arXiv:1511.02583.*

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System.

Codella, N., Cai, J., Abedini, M., Garnavi, R., Halpern, A., & Smith, J. (2015). Deep Learning, Sparse Coding, and SVM for Melanoma Recognition in Dermoscopy Images. 118-126.

D. Mumford, & J. Shah. (1989). *Optimal approximation by piecewise smooth functions and associated variational problems," Communications on Pure and Applied Mathematics* (Vol. 42).

Delibasis, K., & Maglogiannis, I. (2015). Hair removal on dermoscopy images. *Annu Int Conf IEEE Eng Med Biol Soc, pp. 2960-3. doi: 10.1109/EMBC.2015.7319013*.

Deng, J. D.-J.-F. (2009). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248–255).

Dhanachandra, N., Manglem, K., & Chanu, a. Y. (2015). *Image Segmentation Using K -means Clustering Algorithm and Subtractive Clustering Algorithm* (Vol. 54). Elsevier B.V.

Dongare, A., Kharde, R., & Kachare, A. (2008). *Introduction to Artificial Neural Network.*

Elmore, J., Barnhill, R., Elder, D., Longton, G., Pepe, M., Reisch, L., . . . Piepkorn, M. (2017, 6). Pathologists' diagnosis of invasive melanoma and melanocytic proliferations: Observer accuracy and reproducibility study. *BMJ (Online), 357*.

Feng Xiao*a, J. E. (2013). Preferred Color Spaces for White Balancing . *DOI:10.1117/12.478482*.

Fitzmaurice, C. A.-N.-G.-B. (2018). Cancer Incidence, Mortality, Years of Life Lost, Years Lived With Disability, and Disability-Adjusted Life Years for 29 Cancer Groups, 1990 to 2016, a systematic analysis for the global burden of Disease Study blobal burden of disease cancer collaboration. *JAMA Oncology*, 1553-1568.

Getreuer, P. (2012, 8). Chan-Vese Segmentation. *2*, pp. 214-224. Image Processing On Line.

Ghalib, A. V. (2017). Automatic Detection and Classification of Skin Cancer. *International Journal of Intelligent Engineering and Systems , vol. 10, no. 3, pp.444-451, doi: 10.22266/ijies2017.0630.50,*.

H. Iyatomi, H. O. (2008). Computer-Based Classification of Dermoscopy Images of Melanocytic Lesions on Acral Volar Skin," J. Invest. Dermatol. *vol. 128, no. 8, pp. 2049-2054, doi: 10.1038/jid.2008.28*.

Ha, Q., Liu, B., & Liu, F. (2020, 10). Identifying Melanoma Images using EfficientNet Ensemble: Winning Solution to the SIIM-ISIC Melanoma Classification Challenge.

Hagerty, J., Stanley, R., Almubarak, H., Lama, N., Kasmi, R., Guo, P., . . . Stoecker, W. (2019, 7). Deep Learning and Handcrafted Method Fusion: Higher Diagnostic Accuracy for Melanoma Dermoscopy Images. *IEEE Journal of Biomedical and Health Informatics, 23*(4), 1385-1391.

Han, J. (n.d.). *On Multiple Interpretations.*

Han, Z., Wei, B., Zheng, Y., Yin, Y., Li, K., & Li, S. (2017). Breast cancer multi-classification from histopathological images with structured deep learning model. Scientific reports, 7(1), 1-10.

Haofu Liao. (2015). A Deep Learning Approach to Universal Skin Disease Classification.

Hema, D., & Kannan, S. (2019). Interactive Color Image Segmentation using HSV Color Space. *Science and Technology Journal, 7*, Vol. 7 Issue: 1 ISSN: 2321-3388 http://doi.org/10.22232/stj.2019.07.01.05. Retrieved from http://doi.org/10.22232/stj.2019.07.01.05

Hidaka, A. a. (2017). Consecutive Dimensionality Reduction by Canonical Correlation Analysis for Visualization of Convolutional Neural Networks. *roceedings of the ISCIE International Symposium on Stochastic Systems Theory and its Applications*, doi = {10.5687/sss.2017.160.

Ho, Y. &. (2019). The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling. IEEE Access, 8, 4806-4813.

Hoda, M., Bharati Vidyapeeth's Institute of Computer Applications and Management (New Delhi, I., & Institute of Electrical and Electronics Engineers. Delhi Section. (n.d.). *Proceedings of the 8th INDIACom, 2014 International Conference on Computing for Sustainable Global Development (05th - 07th March, 2014) : INDIACom-2014.*

Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. (2016, 8). Densely Connected Convolutional Networks.

Humeau-Heurtier, A. (2019). Texture feature extraction methods: A survey. *IEEE Access, 7*, 8975-9000.

I.S.I.C. (2016). I. S. I. C. (ISIC) Available: https://www.isic-archive.com.

Retrieved from http://www.image-net.org

Inoue, H. (2018). Data augmentation by pairing samples for images classification. arXiv preprint arXiv:1801.02929.

Institute of Electrical and Electronics Engineers. (n.d.). *ICIP 2014 : 2014 IEEE International Conference on Image Processing (ICIP) took place 27-30 October 2014 in Paris, France.*

ISIC. (2020). I. S. I. C. (ISIC). [Online]. Available: https://www.isic-archive.com.

Ismael, S. A., Mohammed, A., & Hefny. (2020). An enhanced deep learning approach for brain cancer MRI images classification using residual networks. Artificial intelligence in medicine, 102, 101779.

J. Kawahara, A. B. (2016). Deep features to classify skin lesions," 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI). *pp. 1397-1400, doi: 10.1109/ISBI.2016.7493528*.

Janocha, K. &. (2017). On loss functions for deep neural networks in classification. arXiv preprint arXiv:1702.05659.

Jolliffe, I. (2005). Principal component analysis." Encyclopedia of statistics in behavioral science .

Karki, S., Kulkarni, P., & Stranieri, A. (2021, 2). Melanoma classification using EfficientNets and Ensemble of models with different input resolution. *2021 Australasian Computer Science Week Multiconference*, 1-5.

Kataev, M. Y., Kataev, S. G., Maksyutov, S., Andreev, A. G., Andreev, A. G., Bazelyuk, S. A., & Lukianov, A. K. (2012). Mathematical algorithms for processing and analysis of near-infrared data from a satellite-borne Fourier transform spectrometer. *p 330-335, DOI:10.1007/s11182-012-9816-3*.

Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. *arxiv.org/abs/1412.6980*.

Kittler, H., Pehamberger, H., Wolff, K., & Binder, M. (2002, 3). Diagnostic accuracy of dermoscopy. *The Lancet Oncology, 3*(3), 159-165.

Krizhevsky, A. I. (2012). Imagenet classification with deep convolutional neural networks.Advances in neural information processing systems 25 .

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, 1097–1105.

Krizhevsky, A., Sutskever, I., & Hinton, G. (n.d.). *ImageNet Classification with Deep Convolutional Neural Networks.* Retrieved from http://code.google.com/p/cuda-convnet/

L. Li, Q. Z. (2014). Automatic diagnosis of melanoma using machine learning methods on a spectroscopic system. *BMC Medical Imaging, vol. 14, no. 36*.

Land, E., & McCann, J. (1971, 1). Lightness and Retinex Theory. *Journal of the Optical Society of America, 61*(1), 1.

LeCun, Y. (1998). Object recognition with gradient-based learning.

Li, Y., & Shen, L. (2018, 2). Skin Lesion Analysis towards Melanoma Detection Using Deep Learning Network. *Sensors, 18*(2), 556.

Linda G. Shapiro, & George C. Stockman. (2001). *Computer Vision* (Vols. ISBN 0-13-030796-3). New Jersey, Prentice-Hall,.

M, H., & M.N, S. (2015, 3). A Review on Evaluation Metrics for Data Classification Evaluations. Academy and Industry Research Collaboration Center (AIRCC).

M. E. Celebi, H. A. (2007). A methodological approach to the classification of dermoscopy images. *Comp. Med. Imag. and Graph., vol. 31, no. 6, pp. 362–373, doi: 10.1016/j.compmedimag.2007.01.003*.

M. E. Celebi, Y. A. (2017). Unsupervised border detection in dermoscopy images," Skin Research and Technology. *vol. 13, no. 4, pp. 454-462, doi: 10.1111/j.1600-0846.2007.00251.x*.

Majtner, T., Yildirim-Yayilgan, S., & Hardeberg, J. (2016). Efficient Melanoma Detection Using Texture-Based RSurf Features. 30-37.

Marchetti, M., Codella, N., Dusza, S., Gutman, D., Helba, B., Kalloo, A., . . . Halpern, A. (2018, 2). Results of the 2016 International Skin Imaging Collaboration International Symposium on Biomedical Imaging challenge: Comparison of the accuracy of computer algorithms to dermatologists for the diagnosis of melanoma from dermoscopic images. *Journal of the American Academy of Dermatology, 78*(2), 270-277.e1.

Mina, S. (2018). Available on: https://ffmpeg.org/doxygen/trunk/vf__colorconstancy_8c_source.html.

Mitchell, T. (1997). *Machine Learning.*

Møllersen, K., Hardeberg, J., & Godtliebsen, F. (2015, 8). Divergence-based colour features for melanoma detection. *Colour and Visual Computing Symposium (CVCS)*, 1-6.

Retrieved from http://arxiv.org/abs/1707.00058

Myers, D. J. (1989). Efficient implementation of piecewise linear activation function for digital VLSI neural networks. Electronics Letters, 25, 1662.

Nami, N., Giannini, E., Burroni, M., Fimiani, M., & Rubegni, P. (2012, 2). Teledermatology: State-of-the-art and future perspectives. *Expert Review of Dermatology, 7*(1), 1-3.

Nikolaos Sapountzoglou, J. L. (2020). Fault diagnosis in low voltage smart distribution grids using gradient boosting trees. ISSN 0378-7796, https://doi.org/10.1016/j.epsr.2020.106254.

Nunnari, F., Bhuvaneshwara, C., Ezema, A., & Sonntag, D. (2020). A Study on the Fusion of Pixels and Patient Metadata in CNN-Based Classification of Skin Lesion Images. 191-208.

O'Mahony, N., Campbell, S., Carvalho, A., Harapanahalli, S., Hernandez, G. V., Krpalkova, L., . . . Walsh, J. (2020). Deep Learning vs. Traditional Computer Vision. DOI: 10.1007/978-3-030-17795-9_10.

Ojala', T., Pietikhenl, M., & Harwood2, D. (n.d.). *Performance Evaluation of Texture Measures with Classification Based on Kullback Discrimination of Distributions.*

Pacheco, A., & Krohling, R. (2019, 9). The impact of patient clinical information on automated skin cancer detection.

Pedregosa, F. V. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research.*, 2825–2830.

Petra Fedorcova, S., & Roman Sutka, J. (2015). Misdiagnosis of Melanoma: A 7 Year Single-Center Analysis. *Journal of Pigmentary Disorders, 02*(09).

Q. Abbas, M. E. (2011). Computer-aided pattern classificationsystem for dermoscopy images," Skin research and technology. *vol. 18, no. 3, pp. 278-289, doi: 10.1111/j.1600-0846.2011*.

R. Erol, M. B. (2017). Texture based skin lesionabruptness quantification to detect malignancy," BMC Bioinformatics, pp. 51–60, . *doi: 10.1186/s12859-017-1892-5*.

Rahmat, R., Chairunnisa, T., Gunawan, D., & Sitompul, O. (2016, 12). Skin color segmentation using multi-color space threshold. *2016 3rd International Conference on Computer and Information Sciences, ICCOINS 2016 - Proceedings*, 391-396.

Rastgoo, M., Lemaitre, G., Massich, J., Morel, O., Marzani, F., Garcia, R., & Meriaudeau, F. (2016). Tackling the Problem of Data Imbalancing for Melanoma Classification. DOI: 10.1007/978-3-030-17795-9_10.

Rena Indriani, O., Atika Sari, C., Jaya Kusuma, E., Hari Rachmawanto, E., & Rosal Ignatius Moses Setiadi, D. (n.d.). *Tomatoes Classification Using K-NN Based on GLCM and HSV Color Space.*

Reza, A. (2004, 8). Realization of the Contrast Limited Adaptive Histogram Equalization (CLAHE) for Real-Time Image Enhancement. *The Journal of VLSI Signal Processing-Systems for Signal, Image, and Video Technology, 38*(1), 35-44.

Rezvantalab, A., Safigholi, H., & Karimijeshni, S. (2018, 10). Dermatologist Level Dermoscopy Skin Cancer Classification Using Different Deep Learning Convolutional Neural Networks Algorithms.

Riaz, F., Hassan, A., & Javed, M. Y. (2014, 8). Detecting melanoma in dermoscopy images using scale adaptive local binary patterns. *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 6758-6761.

Richter, G., & MacFarlane, A. (2005, 3). The impact of metadata on the accuracy of automated patent classification. *World Patent Information, 27*(1), 13-26.

Rokach, L. M. (2015). Data Mining and Knowledge Discovery Handbook. Springer, Boston, https://doi.org/10.1007/0-387-25465-X_9.

Ronneberger, O., Fischer, P., & Brox, T. (2015, 5). U-Net: Convolutional Networks for Biomedical Image Segmentation.

Rotemberg, V., Kurtansky, N., Betz-Stablein, B., Caffery, L., Chousakos, E., Codella, N., . . . Soyer, H. (2021, 12). A patient-centric dataset of images and metadata for identifying melanomas using clinical context. *Scientific Data, 8*(1), 34.

S. J. Pan and Q. Yang. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345-1359, doi: 10.1109/TKDE.2009.191.

S. Seidenari, G. P. (2005). "Colors in atypical nevi: A computer description reproducing clinical assessment," Skin research and technology, . *vol 11, no. 1, pp. 36–41, doi: 10.1111/j.1600-0846.2005.00097.x*.

Saha, A. (2020). *Github*. Retrieved from https://github.com/anindox8/Multi-Color-Space-Features-for-Dermatoscopy-Classification

Salakhutdinov, N. S. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 1929--1958, url= http://jmlr.org/papers/v15/srivastava14a.html.

Sami, M. (n.d.). [Available] [GitHub] https://github.com/MinaSGorgy/Color-Constancy.

Samuel, A. (1967). *Some Studies in Machine Learning Using the Game of Checkers. II-Recent Progress.*

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018, 1). MobileNetV2: Inverted Residuals and Linear Bottlenecks.

Sayed, G. I., Soliman, M. M., & Hassanien, A. E. (2021). A novel melanoma prediction model for imbalanced data using optimized SqueezeNet by bald eagle search optimization. DOI: 10.1016/j.compbiomed.2021.104712.

Shaha, M. &. (2018). Transfer learning for image classification. Second international conference on electronics, communication and aerospace technology (ICECA) (pp. 656-660). IEEE.

Sharma, S. S. (2017). Activation functions in neural networks. towards data science 6.12 (2017): 310-316.

Shen, Y. L. (2018). Skin Lesion Analysis towards Melanoma Detection Using DeepLearning Network," Sensors (Basel), vol. 18, no. 12, pp. 556, doi: 10.3390/s18020556.

Shin, H.-C. a. (2016). Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning. *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1285-1298, doi: 10.1109/TMI.2016.2528162.

Sibi, P. J. (2013). Analysis of different activation functions using back propagation neural networks. Journal of theoretical and applied information technology, 47(3), 1264-1268.

Sirazitdinov, I., Kholiavchenko, M., Kuleev, R., & Ibragimov, B. (2019). Data Augmentation for Chest Pathologies Classification. IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019).

Skin Cancer Facts & Statistics. (2022). *Skin Cancer Foundation*, Available: https://www.skincancer.org/skin-cancer-information/skin-cancer-facts/.

Sobel. (n.d.). Website Available on source: https://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm.

Society, I. C. (2013). IEEE International Conference on Control System, Computing and Engineering. *IEEE International Conference on Control System, Computing and Engineering .* Chapter Malaysia; Institute of Electrical and Electronics Engineers, Proceedings, : ICCSCE 2013 : 29 Nov. - 1 Dec. 2013, Parkroyal Penang Resort, Batu Ferringhi, Penang, Malaysia.

Spanhol, F., Oliveira, L. S., Petitjean, C., & Heutte, L. (2020). Breast cancer histopathological image classification using convolutional neural networks. In International Joint Conference on Neural Networks (2016).

Stanley, R., Stoecker, W., & Moss, R. (n.d.). *A relative color approach to color discrimination for malignant melanoma detection in dermoscopy images.*

Tan, M., & Le, Q. (2019, 5). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.

Thompson, A. C. (1996). Minkowski geometry. Cambridge University Press, .

Turing, A. (1950). *MIND A QUARTERLY REVIEW OF PSYCHOLOGY AND PHILOSOPHY I.-COMPUTING MACHINERY AND INTELLIGENCE.* Retrieved from http://mind.oxfordjournals.org/

Umbaugh, S., Moss, R., & Stoecker, W. (1992, 5). An automatic color segmentation algorithm with application to identification of skin tumor borders. *Computerized Medical Imaging and Graphics, 16*(3), 227-235.

Urbancek, S., Fedorcova, P., Tomkova, J., & R. Sutka. (2015). "Misdiagnosis of Melanoma: A 7 Year Single-Center Analysis," Journal of Pigmentary Disorders. *doi: 10.4172/2376-0427.1000208*.

van de Weijer, J., & Gevers, T. (2005). Color constancy based on the Grey-edge hypothesis. *IEEE International Conference on Image Processing 2005*, II-722.

van de Weijer, J., Gevers, T., Gijsenij, A., van de Weijer, J., Gevers, T., & Gijsenij, A. (2007). Edge-Based Color Constancy. *IEEE Transactions on Image Processing, Institute of Electrical and Electronics Engineers, 16*(9), 100. Retrieved from https://hal.inria.fr/inria-00548686

Van der Walt, S. S.-I. (2014). scikit-image: image processing in Python. PeerJ, 2, e453.

Van der Walt, S.-I. (2014). scikit-image: image processing in Python. PeerJ, 2, e453.

W. Stoecker, W. W. (1992). Automatic detection of asymmetry in skin tumors," Computerized medical imaging and graphics: the official journal of the Computerized Medical Imaging. *vol. 16, no. 3, pp. 191–197, doi: 10.1016/0895-6111(92)90073-I*.

Wang, L. (1990). *Texture Unit, Texture Spectrum, and Texture Analysis.* IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING (Vol. 28, Issue 4).

Wang, L., & He, D.-C. (1990). Texture classification using texture spectrum. In *Pattern Recognition* (Vol. 23, pp. 905-910). Volume 23, Issue 8,ISSN 0031-3203,https://doi.org/10.1016/0031-3203(90)90135-8.

Xia, P. Z. (2015). Learning similarity with cosine similarity ensemble. Information Sciences, 307, 39-52.

Xia, Z., Ma, X., Shen, Z., Sun, X., Xiong, N., & Jeon, B. (2018). Secure image LBP feature extraction in cloud-based smart campus. Institute of Electrical and Electronics Engineers Inc.

Xie, F., Fan, H., Li, Y., Jiang, Z., Meng, R., & Bovik, A. (2017, 3). Melanoma classification on dermoscopy images using a neural network ensemble model. *IEEE Transactions on Medical Imaging, 36*(3), 849-858.

Yani, M. a. (2019). Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry's Nail. *Journal of Physics: Conference Series*, doi = {10.1088/1742-6596/1201/1/012052.

Yap, J., Yolland, W., & Tschandl, P. (2018, 11). Multimodal skin lesion classification using deep learning. *Experimental Dermatology, 27*(11), 1261-1267.

Zhang, Z. &. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. Advances in neural information processing systems, 31.

Zhou, X., & Belkin, M. (2014). Semi-Supervised Learning. 1239-1269. Retrieved from https://linkinghub.elsevier.com/retrieve/pii/B978012396502800022X

Zornberg, M. E. (2014). Automated quantification of clinically significant colors in dermoscopy images and its application to skin lesion classification. *IEEE Systems Journal, vol. 8, no. 3, pp. 980-984, doi: 10.1109/JSYST.2014.2313671*.