University of Piraeus

Department of Digital Systems

Program “Information Systems and Services”

Area of Study: Big Data and Analytics

SKIN LESION ANALYSIS TOWARDS MELANOMA DETECTION FROM DERMOSCOPIC IMAGES USING CONVOLUTIONAL NEURAL NETWORKS

Konstantina Avgeraki

RN:ME1803

Master Thesis

Advisor: Ilias Maglogiannis, Professor and Head of the Dept of Digital Systems

Piraeus 2021

Abstract

Skin cancer is one of the most lethal types of cancer worldwide. There are different types of cancer but the most dangerous one is melanoma. In US alone, 1 out of 5 people will develop skin cancer by the age of 70 and more than 2 people die of skin cancer every hour. Taking these facts under consideration, early detection of skin cancer is crucial, and many lives could be saved. Deep learning technologies are proven to be of great aid in early detection.

In this master thesis, different convolutional neural networks will be presented and tested on HAM10000 dataset of ISIC challenge dataset of 2018, containing 10.000 images of skin lesion from seven different classes. Neural networks will be trained on a portion of the dataset and then tested in new images. The convolutional networks used are Densenet201, Inception V3 and an Ensemble model of the previous two. Lastly, an application is developed which uses by default a convolutional network defined by the user. The user is able to feed an image in the application and a result out of the seven classes predefined will be presented.

Acknowledgements

I would like to thank Prof. Ilias Maglogiannis of the Dept. of Digital Systems of University of Piraeus for his supervision over my Master Thesis, but also for the encouragement and guidance he provided.

I would also like to thank my family and friends for always supporting me in every step and decision I make.

Contents

List of figures

List of tables

Chapter 1: Introduction

1.1 Background of the problem

Skin is the largest organ of the body. Its job is to protect the internal body tissues against extremes of temperature, damaging sunlight, and harmful chemicals, as well as from infections and germs. Skin is also a huge sensor packed with nerves for keeping the brain in touch with the outside world. At the same time, skin allows us free movement, proving itself an amazingly versatile organ. [1]

Skin cancer is the most common form of malignancy that affects human populations all over the world. It is primarily diagnosed visually, beginning with an initial clinical screening by a dermatologist, and potentially followed by dermoscopic analysis, a biopsy, and histopathological examination. For the past years, people would visit their doctors either for their regular routine check or when they would spot something odd on their skin leading them to believe there is an urgency for medical attention. Dermatologists would examine the patient's skin to determine whether the skin changes are likely to be the cause of skin cancer. If there was need for further examination the doctor would order a biopsy, to determine whether the patient suffers from skin cancer and the type of the cancer present.

It is extremely important to identify the cancer in the early stages to achieve a successful treatment. A study conducted by German researchers from the Association of Dermatological Prevention showed a 48% relative reduction in melanoma mortality after carrying out a skin cancer awareness campaign, clinician education and training, and screening of nearly 20% of eligible adults. [2] In order to achieve an early and accurate detection of the cancer, medicine has expanded her boundaries and included Machine Learning Techniques.

Research has shown that software based on deep learning, known as convolutional neural networks, has performed better in identifying cancer than experienced dermatologists. This assumption has been proven in a paper published in the leading cancer research journal Annals of Oncology [3]

There are three main types of skin cancer based on where the cancer begins, Basal cell carcinoma, Squamous cell carcinoma and Melanoma. The first two are known as non-melanoma skin cancers and the last is considered to be the most dangerous skin cancer, as it tends to spread widely. [4]

This thesis is focused on the classification of skin lesion, in order to determine the type of cancer. The American Cancer Society's estimates that in the United States for 2021, there will be 1.9 million new cancer cases diagnosed and 608,570 cancer deaths which makes death by cancer the second most common cause of death in the United States. For melanoma in particular there is an estimation of 5.8% increase in newly reported cases meaning approximately 207.390 new cases will be reported and an increase of 4.8% in deaths, meaning 7.180 people will die of melanoma [5].

Since it is so difficult even for experienced dermatologists to correctly identify skin cancer, in the later years machine learning has come to the aid of this cause. Convolutional neural networks have proven to be extremely helpful and accurate in classifying skin cancer and even proved to perform better than dermatologists.

1.2 Structure of the thesis

This thesis consists of 5 chapters.

In the first chapter an Introduction and background of the skin cancer problem is presented.

In the second chapter basic concepts regarding skin cancer definition, its recognition, and details about how Artificial Intelligence and Machine Learning can aid in its early detection are presented. Also related published papers are also analyzed regarding skin cancer recognition through machine learning techniques.

In the third chapter the dataset is analyzed through graphic representations and the pre-process performed in the dataset is documented. A detailed analysis of the convolutional neural networks that will be used in the experiments is performed.

In the fourth chapter the experiments and the results are presented. The networks used are InceptionV3, Resnet201 and Ensemble model combining the previous two. Also, an application developed is explained. where the user can insert an image of one of the seventh classes in which the neural network was performed and the application outputs the result of its prediction.

In the fifth chapter a comparison is made regarding the published papers presented in the second chapter and the results accomplished of this thesis. Lastly, some recommendations for future work are mentioned.

Chapter 2: Literature review

2.1. Skin cancer

2.1.1. Definition

Skin cancer is by far the most common type of cancer. Skin cancer is the abnormal growth of cells in the epidermis which leads to mutation and skin cells grow so rapidly that form tumors. The most common cause of skin cancer is the sun and the Ultraviolent rays the skin is exposed to. Cancer would more often appear in areas that are exposed to the sun such as scalp, face, lips, ears, neck, chest, arm, and hands and on the legs in women. [6]

Although skin cancer is as mentioned above one of the most common types of cancer, the silver lighting is that if detected in the early stages before it has become a full-grown skin cancer it is possible to be fully treated.

Types of skin cancer:

Basal cell carcinoma (BCC)

Basal cell carcinoma is the abnormal, uncontrolled growth of cells that arise from the skin's basal cells in epidermis. If not treated early this type of cancer can be extremely dangerous and even fatal on some occasions.

Squamous cell carcinoma (SCC)

Squamous cell carcinoma (SCC) is an uncontrolled growth of abnormal cells arising from the squamous cells in epidermis. It is the second most common type of cancer and is extremely dangerous as it is quite common to metastasize if not treated early enough.

Melanoma

Melanoma is the most dangerous type of skin cancer. It is produced from the pigment-producing cells that are commonly known as melanocytes which are the cells that produce melanin the pigment that gives your skin its color. It is curable if detected and treated early.

Merkel cell carcinoma (MCC)

Merkel cell carcinoma is associated with a virus called Merkel cell polyomavirus. Although it is a rarer type of cancer than the other three, it can prove extremely dangerous if not detected early as it can metastasize and infect other parts of the body.

There are many more types of skin cancer than the ones mentioned above, others are benign and others malignant which are life threatening. Early detection is one of the most effective ways of successful and full treatment, so it is crucial to discover the best ways to achieve that.

2.2. Dermoscopy

2.2.1. Definition

When having to deal with skin cancer, dermatologists often encounter many difficulties in correctly identifying whether a skin lesion is malignant and needs of immediate attention or whether it is benign. During naked eye examinations of the skin, much of the light transmitted toward the skin is reflected off the stratum corneum and precludes the observer from visualizing the light reflected from the deeper layers of the skin. So naked eye examination is mostly efficient when having to deal with morphological features on the surface layer of the skin. [7]

Dermoscopy, also known as epiluminescence microscopy, or skin surface microscopy, is a non-invasive, in-vivo technique, which has traditionally found use in the evaluation and differentiation of suspicious melanocytic lesions from dysplastic lesions and melanomas, as well as keratinocyte skin cancers such as basal cell carcinoma (BCC) and squamous cell carcinoma (SCC) [8]

Dermoscopy is the study of skin lesions with the aid of a dermatoscope. A usual dermatoscope consists of a magnifier, a light source that is nonpolarized, a transparent plate and a liquid medium between the dermatoscope and the skin. Most modern dermatoscopes do not need the liquid to overcome the reflections on the skin, but instead use polarized light. [9]

With this modern technique it is also possible to store images and videos of the skin lesion for future use. This way it is possible to compare a skin lesion with any lesion to reach to a conclusion whether it is malignant or benign, or even compare the skin lesion of the same patient over time to identify the changes. Usually if a skin lesion remains the same it is not life threatening and when it changes it may be subject to removal.

2.3. Basic Technologies

2.3.1. Artificial Intelligence

Artificial Intelligence, most commonly known as AI, refers to artificially making a computer intelligent, in a way to be able to think, learn, plan, and take actions as a human would do. To achieve that, one should programmatically code the machine. [10]

The explosion of “Big Data” as well as the extreme progress of Internet has made the possibilities of the usage of AI endless. AI can be used in several sectors and industries, such as Health Care, Banking, Public Safety, Manufacturing, Retail and many more.

The main reason that AI was originally used, was to automate tasks since it has the ability to process huge amount of data and make faster and more accurate calculations than humans. AI has the ability to process data, learn from them and identify patterns in order to make predictions or take some actions. [11]

Two categories exist for AI, Narrow AI and Strong AI, also known as artificial general intelligence (AGI). Narrow AI is designed to perform a specific task, while Strong AI is designed to mimic the human brain and perform more complex tasks. [11]

Although AI can achieve great performances in a very quick manner, it also has limitations. The main limitation is that it can perform only as good as the data, that it used to learn, are. If the data are insufficient or corrupted, then the results will not be as good as expected. Apart from the limitations, there are also concerns about the usage of AI, that people have been arguing about. For example, a lack of transparency and interpretability in decision-making or issues of data quality and potential bias. Also, safety and security implications, as well as considerations regarding accountability and its potentially disruptive impacts on social and economic structures. As AI is being used more and more in our everyday life, the are some main factors that need to be addressed [12]:

- Socio-economic impact: The fact that machines can now perform tasks mimicking human behavior, and even perform better and faster than humans, has opened new opportunities. This could help in achieving a more effective medical care system or improve productivity and help make industries and services safer.

- Transparency, bias, and accountability: As mentioned above the result of the performance of AI can be as good as the data it uses. So, there is the possibility that AI can make errors or discriminate against some specific individuals due to its training data. Moreover, since it is not always very clear how the decision is made, it is not easy to solve problems of bias and ensure accountability.

- New uses for data: Algorithms used by AI in decision making, use a large amount of data, commonly known as “Big Data”. It is proven that these algorithms perform better when more data are used, which has caused the need for more and more data. Many believe that because of that, there is the risk of oversharing information and privacy will not be respected.

- Security and safety: Advancements in AI have created safety challenges, since the behavior of AI can be unpredictable, or it can be used for malicious purposes.

- Ethics: AI can also make unethical decisions due to the training data, which arises the issue of building in ethical considerations into AI systems and algorithms.

- New ecosystems: AI has provided new means of interacting with network, like voice recognition, which could create new concerns about how open and accessible the Internet has become.

2.3.2. Machine Learning

Machine Learning is a subset of Artificial Intelligence, which is the study of computer algorithms, in order to learn and improve automatically, without the need to be programmed for specific tasks. This way the machine acts similarly to a human, recognizes patterns and reaches to a conclusion based on the data it has processed.

Machine learning needs three components: dataset, features, and algorithms.

Dataset: For a machine learning system to be trained, it needs a very large sample of collections called dataset. These samples can be numbers, images, sound, videos etc. The better the dataset is the better the system will perform. Therefore, it is difficult and needs a lot of effort and time to create a good dataset.

Features: Different features are the solution for the system to make a decision. The system can identify which of the features are mostly correlated in order to conclude to an accurate result or identify the labels that distinguish right and wrong solution.

Algorithms: There are different algorithms to be used in a specific problem. According to which algorithm is used every time, the accuracy could be better or worse. Also, the speed can differ. So, to achieve the best performance, it is essential for the right algorithms to be selected.

There are so many different machine learning algorithms to choose from, but one should keep in mind that there is no algorithm that performs equally well on all tasks (No Free Lunch Theorem). Based on the specific task in question or the nature of the data that will be processed the right algorithm could differ.

Therefore, algorithms can be categorized in groups depending on their learning style or similarity. [13]

Learning style algorithms

Supervised Learning

Supervised Learning uses labeled data to perform classification or regression. It is called supervised, because throughout the process the procedure is being monitored and corrected by the programmer until it achieves the desired results.

Algorithm examples:

- Naive Bayes
- Support Vector Machine
- Decision Tree
- K-Nearest Neighbors
- Logistic Regression
- Linear and Polynomial regressions

Unsupervised Learning

In unsupervised learning, the program is not provided with any features to search for patterns. Instead, unsupervised learning is mostly used for clustering when trying to divide data into groups by similarity. It is also able to recognize patterns that humans would have missed, as it can identify some patterns to provide some insights, even though the programmer does not know exactly what they are trying to find.

Algorithm examples:

- K-means clustering
- DBSCAN
- Mean-Shift
- Singular Value Decomposition (SVD)
- Principal Component Analysis (PCA)
- Latent Dirichlet allocation (LDA)
- Latent Semantic Analysis, FP-growth

Semi-supervised Learning

Semi-supervised learning is a combination of the above two. It uses both labeled and unlabeled data, and although the desired outcome is known to the programmer, the system must detect patterns to make the desired prediction on its own.

Reinforcement Learning

Reinforcement learning uses trial and does not require constant supervision. The system is able to learn in dynamic, noisy environments, even in real world.

Algorithm examples:

- Q-Learning,
- Genetic algorithm,
- SARSA,
- DQN,
- A3C

Similarity Algorithms

Regression Algorithm

Regression algorithm is modelling relationship between an independent and dependent variable, to predict the values for the dependent variable. To do so, it uses the error generated by the model, to refine the model iteratively. [14]

Algorithm examples:

- Ordinary Least Squares Regression (OLSR)
- Linear Regression
- Logistic Regression
- Stepwise Regression
- Multivariate Adaptive Regression Splines (MARS)
- Locally Estimated Scatterplot Smoothing (LOESS)
- Jackknife Regression

Instance-based Algorithms

Instance based learning model does not use the training data, but instead only an instance or examples of the training data that are deemed important or critical to the model. The results are compared through similarity measures to new data, in order to find the best match and make a prediction. For this reason, instance-based methods are also called winner-take-all methods and memory-based learning. This method does not make any computation unless it is required to perform a test and uses only relative data. [15]

Algorithm examples:

- k-Nearest Neighbor (kNN)
- Learning Vector Quantization (LVQ)
- Self-Organizing Map (SOM)
- Locally Weighted Learning (LWL)
- Support Vector Machines (SVM)

Regularization Algorithms

Regularization is an extension to other models that is adding a penalty to the objective function, based on its complexity.

Algorithm examples:

- Ridge Regression
- Least Absolute Shrinkage and Selection Operator (LASSO)
- Elastic Net
- Least-Angle Regression (LARS)

Decision Tree Algorithms

Decision Tree creates a model that predicts the value of a target variable based on several input variables. [16]. Decisions are a representation of a tree structure. Decision Tress are mostly used for regression and classification problems. [15]

Algorithm examples:

- Random forest
- Conditional Decision Trees
- Classification and Regression Tree (CART)
- 5 and C5.0
- Iterative Dichotomizer 3 (ID3)
- Gradient boosting machines (GBM)
- Chi-Squared Automatic Interaction Detection (CHAID)
- Decision stump
- Multivariate adaptive regression splines (MARS)

Bayesian Algorithms

Bayesian methods are those that explicitly apply Bayes' Theorem for problems such as classification and regression. It provides a principled way for calculating a conditional probability, meaning it calculated the probability of an event occurring, given the probability of another event already occurred. [14]

Bayes's theorem is stated mathematically as the following equation:[3]

$$\boxed{P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}}$$

where A and B are events and { P(B)≠0 }

- P(A | B) is a conditional probability: the likelihood of event A occurring given that B is true.
- P(B | A) is also a conditional probability: the likelihood of event B occurring given that A is true.
- P(A) and P(B) are the probabilities of observing A and B respectively; they are known as the marginal probability. [16]

Algorithm examples:

- Naïve Bayes
- Gaussian Naïve Bayes
- Multinomial Naïve Bayes
- Averaged one-dependence estimators (AODE)
- Bayesian belief network (BBN)
- Hidden Markov Models
- Conditional Random fields (CRFs)

Clustering Algorithms

Clustering methods organize the data into groups based on their similarity. The most common methods used are centroid based and hierarchical methods. On centroid-based methods, the centroid is a point at the center around of which data with maximum similarity form a cluster. Hierarchical methods build a hierarchy of clusters. [14]

Algorithm examples:

- K-means
- Single-Linkage clustering
- K-medians
- Hierarchical Clustering
- Fuzzy Clustering
- DBSCAN
- Expectation maximization (EM)
- Gaussian mixture models (GMM)
- DBSCAN
- OPTICS algorithm
- Non-negative Matrix Factorization
- Latent Dirichlet allocation (LDA)

Association Rule Learning Algorithms

Association rule learning is a rule-based machine learning method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using some measures of interestingness [17]. These rules can discover hidden relations in large multidimensional datasets. [14]

Algorithm examples:

- The Apriori algorithm
- The Eclat algorithm
- FP-Growth

Artificial Neural Network Algorithms

Artificial Neural Networks are models that are basically inspired by biological neural networks and do not need specific programmed rules to perform tasks. An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. They are mostly used in regression or classification problems but can be implemented in various different problems. [14]

Algorithm examples:

- Learning vector quantization (LVQ)
- Self-organizing maps (SOM)
- Hopfield network

- Perceptron
- Backpropagation
- Radial Basis Function Network (RBFN)
- Autoencoders
- Boltzmann Machines
- Spiking Neural Networks
- Multilayer Perceptrons (MLP)
- Stochastic Gradient Descent

Deep Learning Algorithms

Deep Learning methods are a modern update to Artificial Neural Networks that exploit abundant cheap computation.

They are concerned with building much larger and more complex neural networks and, as commented on above, many methods are concerned with very large datasets of labelled analog data, such as image, text. audio, and video.

Algorithm examples:

- Convolutional Neural Network (CNN)
- Recurrent Neural Networks (RNNs)
- Long Short-Term Memory Networks (LSTMs)
- Stacked Auto-Encoders
- Deep Boltzmann Machine (DBM)
- Deep Belief Networks (DBN)

Dimensionality Reduction Algorithms

Like clustering methods, dimensionality reduction seeks and exploits the inherent structure in the data, but in this case in an unsupervised manner or order to summarize or describe data using less information.

This can be useful to visualize dimensional data or to simplify data which can then be used in a supervised learning method. Many of these methods can be adapted for use in classification and regression. [15]

Algorithm examples:

- Principal Component Analysis (PCA)
- Principal Component Regression (PCR)
- Partial Least Squares Regression (PLSR)

- Sammon Mapping
- Multidimensional Scaling (MDS)
- Projection Pursuit
- Linear Discriminant Analysis (LDA)
- Mixture Discriminant Analysis (MDA)
- Quadratic Discriminant Analysis (QDA)
- Flexible Discriminant Analysis (FDA)

Ensemble Algorithms

Ensemble methods are models composed of multiple weaker models that are independently trained and whose predictions are combined in some way to make the overall prediction.

Ensemble methods usually produces more accurate solutions than a single model. Much effort is put into what types of weak learners to combine and the ways in which to combine them. This is a very powerful class of techniques. [15]

Algorithm examples:

- Boosting
- Bootstrapped Aggregation (Bagging)
- AdaBoost
- Weighted Average (Blending)
- Stacked Generalization (Stacking)
- Gradient Boosting Machines (GBM)
- Gradient Boosted Regression Trees (GBRT)
- Random Forest

2.3.3. Convolutional Neural Networks (CNN)

Artificial Intelligence and Machine Learning have experienced an amazing growth, providing the world with more and more capabilities every day in new areas. One of these areas is the Computer Vision.

In Computer Vision, the goal is for machines to perceive the world as humans do and be able to learn, think and make decisions the way humans would only a lot faster. Deep learning schemes have been widely used to solve computer vision problems. The benefit of Deep learning is that it mimics the way human brain processes large scale data by allowing computational models of multiple processing layers to learn and represent data. Deep learning consists of a rich family of methods, like neural networks and a variety of feature learning algorithms. Deep learning methods have become very popular since they have shown evidence of better performance against state-of-the-art techniques. [18]

The fact that there are too many large datasets publicly available has empowered deep learning techniques. One technique that has contributed a lot in Image Recognition is Convolutional Neural Network. Although convolutional Neural Networks are most widely known for image classification, they do apply to other areas as text analysis, or sound recognition when it is represented visually as a spectrogram. [19]

A Convolutional Neural Network is a Deep Learning algorithm, that takes an image as input, process it, assign weights, and biases and classify it into a category. The architecture of Convolutional neural networks is similar to that of the human brain and in particular it is inspired by the visual cortex. The human brain has numerous interconnected neurons. The receptive fields of the different neurons are defined by the size of the region that produces the feature in the input and can capture different patches of the image. [20]

The human brain is able to recognize instantly an image, without any particular effort. This is what is also the goal when using a computer to perform the same task. In a similar way, the computer is able to perform image classification by looking for low level features such as edges and curves, and then building up to more abstract concepts through a series of convolutional layers. This is a general overview of what a CNN does. [21]. In an extremely high-level approach, what CNN does is receive an input (e.g an image) and produce an output (a class or a probability of classes), through a series of activities. In Figure 1, a representation of a CNN used to classify handwritten digits is presented. [22]

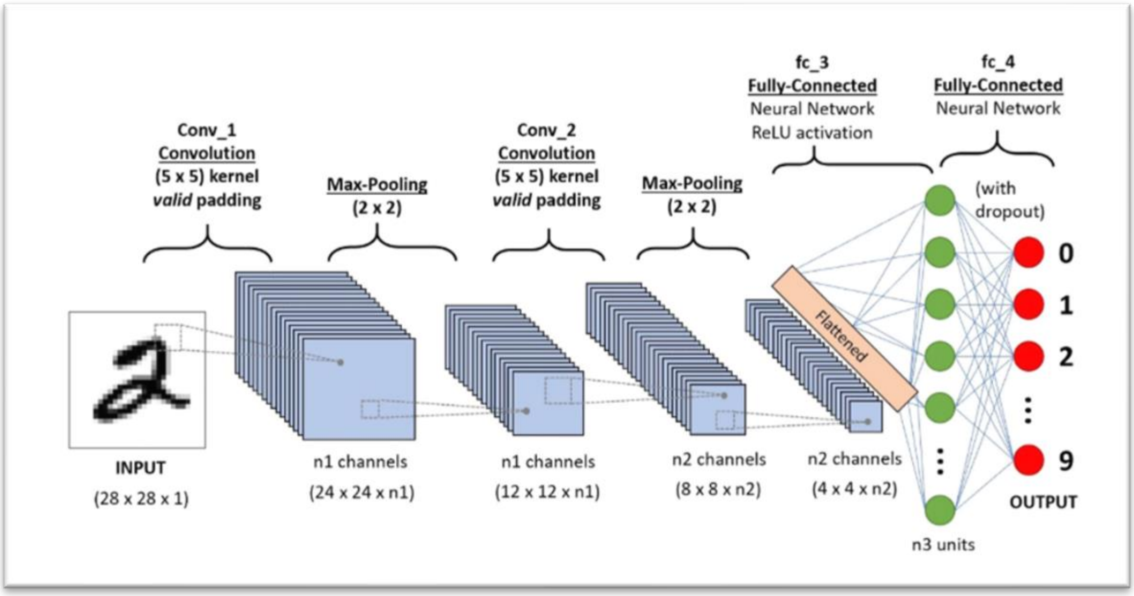

Figure 1. CNN sequence to classify handwritten digits

A CNN consists of convolutional and pooling layers that take place interchangeably optionally ending up to fully connected layers.

Convolutional layers: A convolutional layer is used for feature extraction and consists of filters that have learnable parameters. The height and weight of the filters are smaller than those of the input volume. Convolution is performed between each filter and the input image to create the feature map by sliding across the width and height of the input image and compute dot products between the filter and the input at every spatial position. The output volume is obtained by stacking these feature maps on top of each other across the depth dimension. The height and the weight are smaller than the input which translates into the receptive field for each neuron being small and equal to filter size. [23]

Pooling layers: As mentioned above, convolutional layers create feature maps by recording the exact position of the features. This means that if the preprocess of the input, for example an image, is changed then the feature map created will also differ from the previous one created. To prevent this from happening, Pooling layers are used as a method of downsampling, in order to create a lower resolution version of an input signal by reducing the dimensions(width, height) of the input, but preserving the important features of the image. The new version created is of benefit since it prevents the network from overfitting and also requires less computational cost, so efficiency is improved. The most common practice is a pooling layer to be used after a convolutional layer in a Convolutional neural network. Average and Max pooling are the most widely used functions. [18]

Fully connected layers: Fully connected layers are the last layers of a CNN. The output of the pooling layer of the convolutional layer is flattened and then fed to the fully connected layer. In fully connected layers all inputs of one layer have connections to each activation unit of the next layer, meaning every input neuron is connected to every output neuron. By using these layers, the network can make a more accurate prediction based on the whole image and not only a fraction of it. [18]

Apart from the main Layers of a CNN, there are also other layers that are very commonly used such as Non-linear layers, most of the times using a ReLU activation function which applies an elementwise activation by thresholding at zero.

A ConvNet architecture consists of a repetition of the above layers (Figure 1) in a sequence to best fit the problem in question.

A most certain problem that could arise when training CNNs, due to the need of too many parameters been learned, is overfitting. Solutions to this problem are data augmentation, dropout, and stochastic pooling. A way to accelerate learning process is to use a pretrained model. This way some parameters are already pretrained and someone can select which one he wants to train from beginning. This way, the process is a lot faster.

There are several architectures in the field of Convolutional Networks. The most common are:

LeNet. In 1990´s Yann LeCun introduced the first successful Convolutional Networks, LeNet being one of the most commonly known, used for hand-written digit recognition tasks.

AlexNet. It was inspired by the architecture of LeNet, but was differentiated in terms of depth, length, and layers. It was the first neural network that popularized Convolutional Network, proposed by Alex Krizhevsky, Ilya Sutskever and Geoff Hinton. Until then, only a single Convolutional Layer was used followed by a Pooling layer, but AlexNet introduced the method of stacking layers on top of each other. This neural network won the ILSVRC (ImageNet Large Scale Visual Recognition Competition) challenge in 2012 by achieving a top-5 error of 16%, when the second-best model achieved a 26% error.

ZF Net. In 2013 Matthew Zeiler and Rob Fergus won the ILSVRC by introducing the ZFNet. A significant upgrade over AlexNet that reduced the filter size and the stride on the first layer and achieved a better accuracy.

GoogLeNet. In 2014 the winner of the ILSVRC was GoogleNet introduced by Szegedy et al. from Google. The main breakthrough of GoogleNet is the Inception Module, that reduced the number of parameters in the network to only 4M and the use of Global Average Pooling instead of the fully-connected layers at the end of the CNN, which resulted in a deeper architecture and significant decrease in error rate.

VGGNet. The runner-up in ILSVRC 2014 was the network from Karen Simonyan and Andrew Zisserman that became known as the VGGNet. The network achieved 92.7% top-5 test accuracy in ImageNet by using only 3x3 convolutions and 2x2 pooling, replacing this way the large kernels used by the winner of 2012 ILSVRC, Alexnet, and pointing out the importance of the depth of the network in its efficiency. The inventors also made the weights of the network freely available in Caffe. The most important drawback in VGGNet, is that it uses a lot of memory because of its depth and too many parameters (140M). This makes the network really difficult to train, which results in other smaller networks to be preferred instead. [24]

ResNet. Residual Network developed by Kaiming He et al. was the winner of ILSVRC 2015 with 152 layers. The need for ResNet was born because in some cases, adding more layers on top of a network, increases its training error. ResNet introduces the skip connection, that skips layers and connects to the output. This method was the solution to training deep neural networks without the problem of vanishing gradient, so the training and test error rate would not increase. [25]

CNNs have outperformed standard machine learning techniques in fields like image or patterns recognition. The wide range of applications, the easiness in training and the outstanding performance they can achieve, has made them one of the top and still developing trends in the past years.

2.4. Related Work

2.4.1. ISIC Challenge

The International Skin Imaging Collaboration (ISIC) is an international effort to improve melanoma diagnosis, containing over 13,000 dermoscopic images that were gathered from leading clinical centers. The ISIC Challenge is about skin lesion recognition, lesion segmentation, detection of clinical diagnostic patterns, and lesion classification. It has become widely known and more and more registrations have occurred over the past years. It is one of the largest repositories of images for skin lesion which are evaluated of experts to ensure credibility and quality. As mentioned above it is common for dermatologists to misdiagnose a lesion for benign when it is malignant. Through this challenge there has been numerous algorithms developed for skin lesion classification in aid of improving accuracy and reducing time of diagnosis.

2.4.2. Papers in ISIC

Three papers regarding skin lesion analysis towards melanoma detection will be presented. These papers were first published through ISIC challenge 2018

2.4.2.1. Transfer Learning for Automatic Disease Diagnosis with Dermoscopic Images

In this paper the researchers used pre-trained models as ResNet, Xception, Inception-v4, and DenseNet, and fine-tuned them in PyTorch [26]. They split the dataset into training, test, and validation in a ratio of 80%, 10% and 10% respectively. Firstly, the images were normalized and for the calculation of mean and standard deviation HAM10000 dataset was used. Secondly the images were resized to 800x600 and then cropped to 600x450 in a random manner. This aimed to perform augmentation to the dataset.

Since different classes of the dataset did not contain an equal number of images, it was decided to use different weights and more specifically classes that has fewer images would have higher weights. The parameters used were Model fit: linear, model optimizer: SGD with learning rate 0.002 and momentum 0.8 and in some methods also tried Nesterov Momentum whose results were better that standard momentum. The results after fine -tuning for all models used are the below:

Evaluation Metrics	ResNet 18	Inception -v4	Xception	DenseNet 161	DenseNet 161-nestrov	DenseNet 201	DenseNet 201-nestrov	DenseNet 169	DenseNet 169-nestrov	DenseNet 121-nestrov
Accuracy	80.44%	78.75%	75.77%	81.43%	84.11%	82.52%	83.42%	83.32%	85.30%	81.13%
Precision	79.30%	80.40%	83.61%	83.75%	84.59%	84.29%	84.14%	84.53%	84.95%	81.86%
Recall	80.44%	78.85%	75.77%	81.43%	84.11%	82.52%	83.42%	83.32%	85.30%	81.13%
F1	78.91%	78.62%	77.61%	82.08%	84.23%	83.11%	83.63%	83.71%	84.92%	81.17%

Table 1.Summary of model performance with imbalanced weighted loss function

Final conclusion was that Densenet models outperform the rest models, so the final result for ISIC challenge using these trained networks was as below table:

Model	DenseNet 161	DenseNet 201	DenseNet 161-nestrov	DenseNet 201-nestrov	DenseNet 169-nestrov	DenseNet 121-nestrov
Accuracy	87.40%	88.30%	82.60%	82.70%	85.70%	84.70%

Table 2. ISIC 2018 validation dataset accuracy using different version of Densenet models

2.4.2.2. A Dense CNN approach for skin lesion classification

In this paper the researchers used the pre-trained Densenet-121 architecture and modified it in a reduce version of 61 layers which later was fine-tuned on the HAM10000 dataset for ISIC 2018 Challenge [27]. In order for high discriminative features to be learned they used jointly SoftMax, and center loss functions balanced by means of a hyper parameter. Densenet121 was reduced from 4 dense blocks to 3 without modifying the inside structure of each dense block. Also, the third layer was modified to be composed of 22 convolutional layers instead of 48 of original structure. Input images were 224x224 and a 1x1 convolutional layer followed by 2x2 average pooling was used between two dense blocks. The last dense block included a global average pooling before the final output layer. In the fine-tuning phase, layers from the last layer were fully trained from scratch. The dataset was divided into training and validation set with a ratio of 80% and 20% respectively. The number of images was also increased through random transformation as rotation, flipping and affine. Also, a centered and square cropping was made, of amplitude equal to the shorter side of the starting image which resulted in a dimension of 224x224 for the images. SGD was chosen as optimizer with learning rate starting at 0.01, weight decay 0.0001 and momentum 0.9. The maximum number of iterations was 75000, decreasing the learning rate by a factor of 10 at each step of 20000 iterations. Finally, value 0.8 was used for the λ parameter.

Dataset	MEL	NV	BCC	AKIEC	BKL	DF	VASC	Total
Augmented Dataset	1113	6705	514	327	1099	115	142	10015
Augmented Training Dataset	891	5364	412	262	880	92	114	8015
Augmented Test Dataset	222	1341	102	65	219	23	28	2000

Table 3.Dataset split: 80% training; 20% test

Dataset	MEL	NV	BCC	AKIEC	BKL	DF	VASC	Total
Augmented Dataset	50529	87165	46774	45453	51653	43815	43990	369379
Augmented Training Dataset	40095	69732	37492	36418	41360	35052	35226	295375
Augmented Test Dataset	10434	17433	9282	9035	10293	8763	8764	74004

Table 4.Balanced Dataset

The accuracy that the neural network achieved was 89,2% in the validation set.

2.4.2.3. Computer Aided Diagnosis of Skin Lesions from Morphological Features

In this paper an ensemble model was introduced consisting of five neural networks [28]:

- ResNet 50 v1
- Inception v3
- Xception
- DenseNet 201
- InceptionResNet v2

The fully connected layer of each pretrained network was replaced with a randomly initialized (Glorot Uniform) matrix. The fully connected layer was penalized with both an L2 and an L1 regularizer, each weighted at 0.01. A categorical SoftMax cross entropy term weighted at 1.0 with the Adam optimizer were used for optimization and all weights were optimized at all steps of training. The initial learning rate was $10-4$ which was then scaled by a factor of 0.1 when validation accuracy would not improve for 10 steps. Rest parameters of the Adam algorithm were not changed, default values were used. The input images for each model were of size 299×299 for Inception, Xception, and InceptionResNet and 224x224 for Resnet and DenseNet. For the final prediction, the images were resized to 329 × 329 or 254 × 254 and cropped to fit each preprocessing function. For training, each image was scaled to normally distributed random dimensions centered at 30 more than the desired dimensions with a standard deviation of 15. Afterwards, the images were randomly rotated, bilinear interpolation, cropped, flipped, brightness and contrast were adjusted with a max change of 30% in both cases. The transformation was performed at each training step. The altered images, were inserted to each model's preprocessing function before training. Finally, training set was further augmented with all of the remaining images from the ISIC archive whose "diagnosis" tag appeared to fit under any of our 7 classes.

Models were evaluated after 200 training steps using 20 random batches of class-balanced validation data. Accuracy for each model for training data are represented below:

Model	Inception V3	Xception	InceptionResNet v2	ResNet 50 v1	DenseNet 201	Ensemble
Accuracy	87.70%	86.20%	87.80%	85.20%	88.20%	85.2%

Table 5. validation accuracy for each model during training

Precision and recall for validation data are represented in tables below.

Model	Melanoma	Melanocytic nevus	BCC	Actinic keratosis	Bening keratosis	Dermato fibroma	Vascular Lesion
Inception	0.649	0.957	0.809	0.735	0.818	0.85	0.952
Xception	0.591	0.959	0.847	0.672	0.794	0.882	0.952
InceptionResNet	0.573	0.966	0.921	0.76	0.793	0.933	0.87
ResNet	0.689	0.947	0.873	0.667	0.818	0.882	0.875
DenseNet	0.587	0.967	0.763	0.632	0.754	0.895	0.909
Ensemble	0.709	0.965	0.822	0.725	0.827	0.944	0.909

Table 6. Precision of each model by class

Model	Melanoma	Melanocytic nevus	BCC	Actinic keratosis	Bening keratosis	Dermato fibroma	Vascular Lesion
Inception	0.743	0.923	0.923	0.72	0.818	0.944	0.909
Xception	0.76	0.901	0.923	0.78	0.794	0.833	0.909
InceptionResNet	0.802	0.892	0.897	0.76	0.861	0.778	0.909
ResNet	0.689	0.939	0.885	0.68	0.842	0.833	0.955
DenseNet	0.749	0.891	0.91	0.72	0.8	0.944	0.909
Ensemble	0.772	0.93	0.949	0.74	0.867	0.944	0.909

Table 7. Recall of each model by class

Chapter 3: Methodology and dataset

3.1. Infrastructure

For the needs of this thesis, the training of the CNNS was performed on a single pc which concluded in the need of a great deal of time for the CNNs to be trained due to the lack of the desired resources. Firstly, Google Colab was used to compute the necessary training. Google Colab is a free Jupyter notebook environment that runs entirely in the cloud. The biggest benefit by using Google Colab is that it gives you access to GPUs with no setup required. Unfortunately, the most important downside is that the maximum duration of using the GPU is restricted to 12 hours [29]. This encountered many difficulties in the experimentation, so it was decided to perform the training locally even though the computation lasted longer. The pc used was an i7 Intel core system with 12GB Ram.

3.2. Dataset

The dataset used in this thesis is retrieved from the annual ISIC Challenge and most specifically “ISIC 2018: Skin Lesion Analysis Towards Melanoma Detection” grand challenge datasets [30] [31]. The International Skin Imaging Collaboration has a repository of over 13.000 dermoscopic images of skin lesions, which are all screened for privacy and quality assurance. The images origin from HAM10000 and are categorized in seven classes as below:

- Melanoma
- Melanocytic nevus
- Basal cell carcinoma
- Actinic keratosis / Bowen’s disease (intraepithelial carcinoma)
- Benign keratosis (solar lentigo / seborrheic keratosis / lichen planus-like keratosis)
- Dermatofibroma
- Vascular lesion

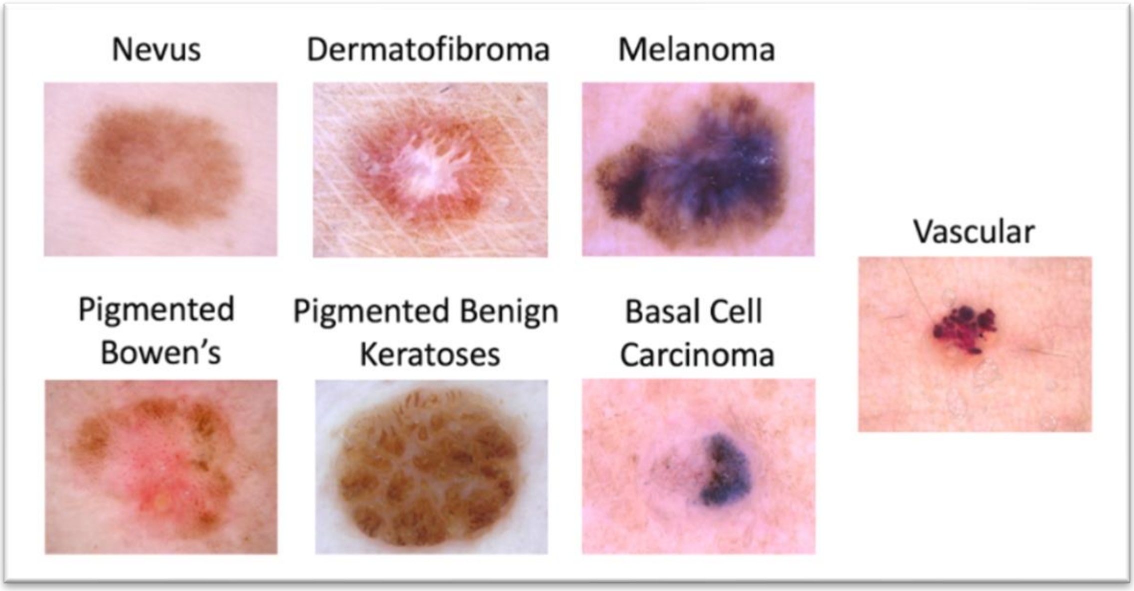

Figure 2. Sample of skin lesions provided by ISIC ARCHIVE

All lesion images are named using the scheme ISIC_.jpg, and is a 7-digit unique identifier followed by a csv file that contains the response data. In the file the respective image is included and the classification response for each image as below:

1. image: an input image identifier of the form ISIC_
2. MEL: "Melanoma" diagnosis confidence
3. NV: "Melanocytic nevus" diagnosis confidence
4. BCC: "Basal cell carcinoma" diagnosis confidence
5. AKIEC: "Actinic keratosis / Bowen's disease (intraepithelial carcinoma)" diagnosis confidence
6. BKL: "Benign keratosis (solar lentigo / seborrheic keratosis / lichen planus-like keratosis)" diagnosis confidence
7. DF: "Dermatofibroma" diagnosis confidence
8. VASC: "Vascular lesion" diagnosis confidence

Diagnosis for each image was provided by one of the following methods:

- Histopathology
- Reflectance confocal microscopy
- Lesion did not change during digital dermatoscopic follow up for over two years with at least three images
- Consensus of at least three expert dermatologists from a single image

Only in cases of malignancy, the diagnosis was also histopathologically confirmed.

The training data consist of 10015 images and a csv file as described above with ground truth. The validation consists of 193 images and the test data consist of 1512 images. Since no ground truth is provided for test and validation sets, only training data set will be used which will be further split to acquire images for test and validation.

3.2.1. Statistics

As the dataset tried to represent a “real-life” model, there are far more benign cases than malignant ones which is obvious in Figure 3, where 0 represents the benign cases and 1 the malignant cases. In Figure 4 a more detailed plot is represented, visualizing explicitly how many cases were registered for each type.

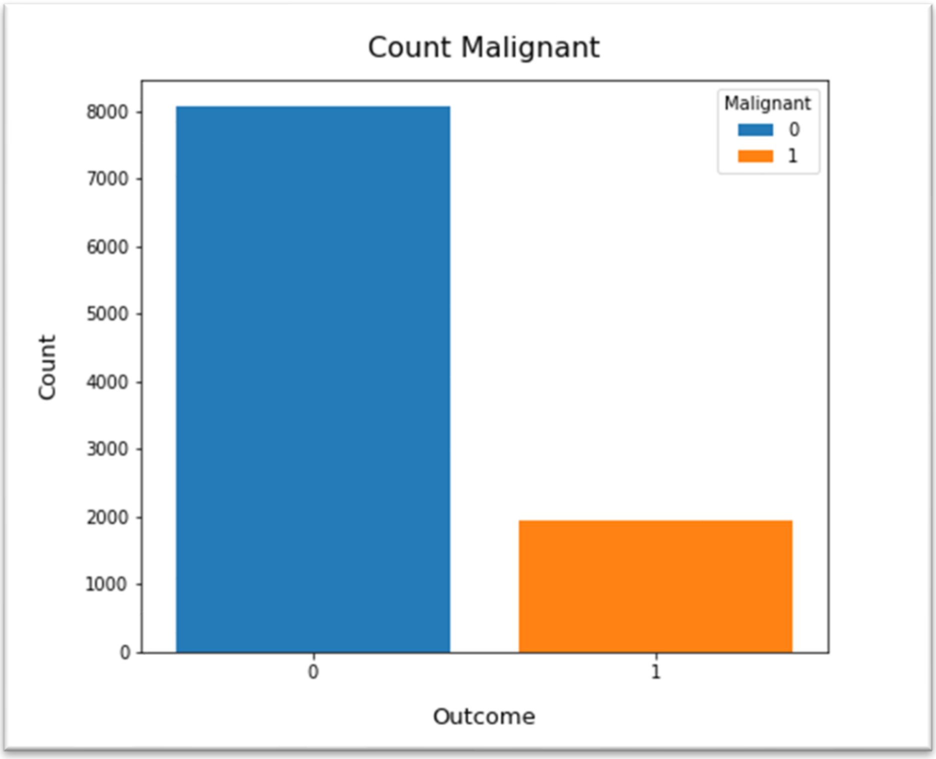

Figure 3. Malignant(1) vs bening(0)

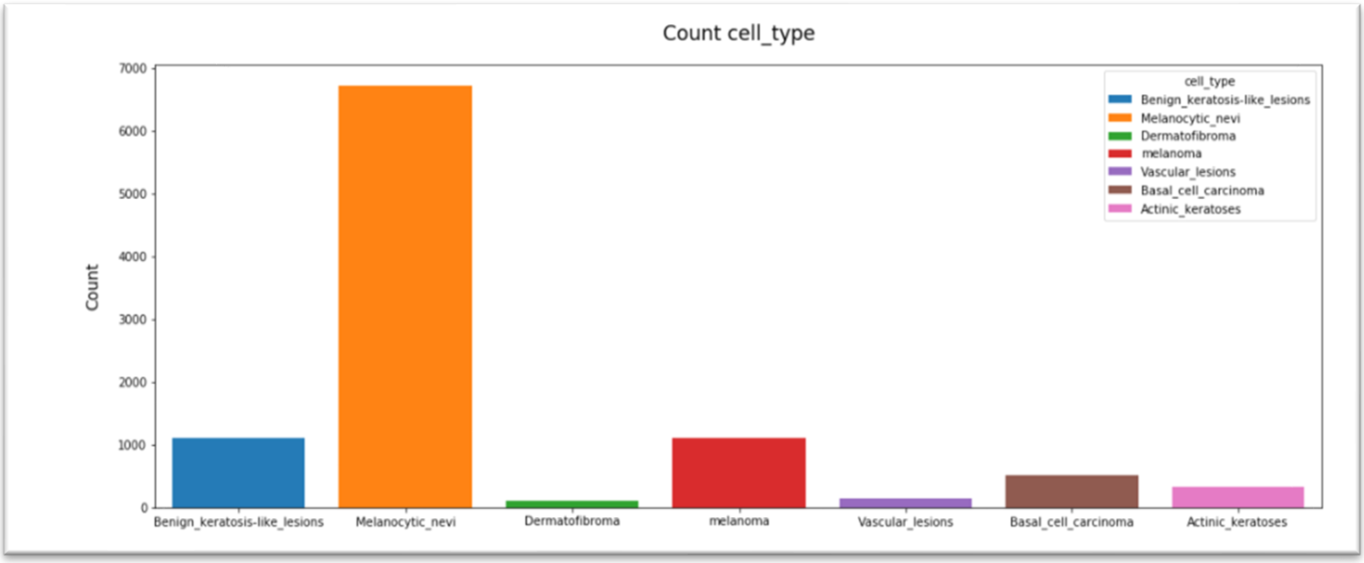

Figure 4. Number of cases foe each disease category

Also, a balance between how many males and how many females was mostly maintained (Figure 5) and a variation in the age of the people whose images were included in the dataset (Figure 6)

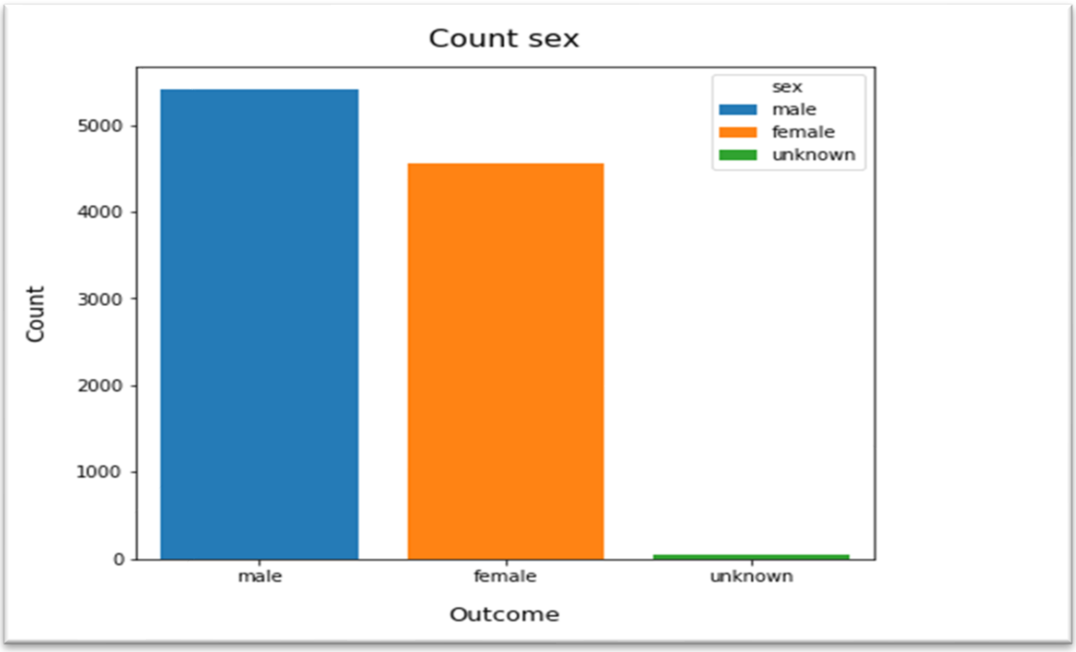

Figure 5. Count of sex

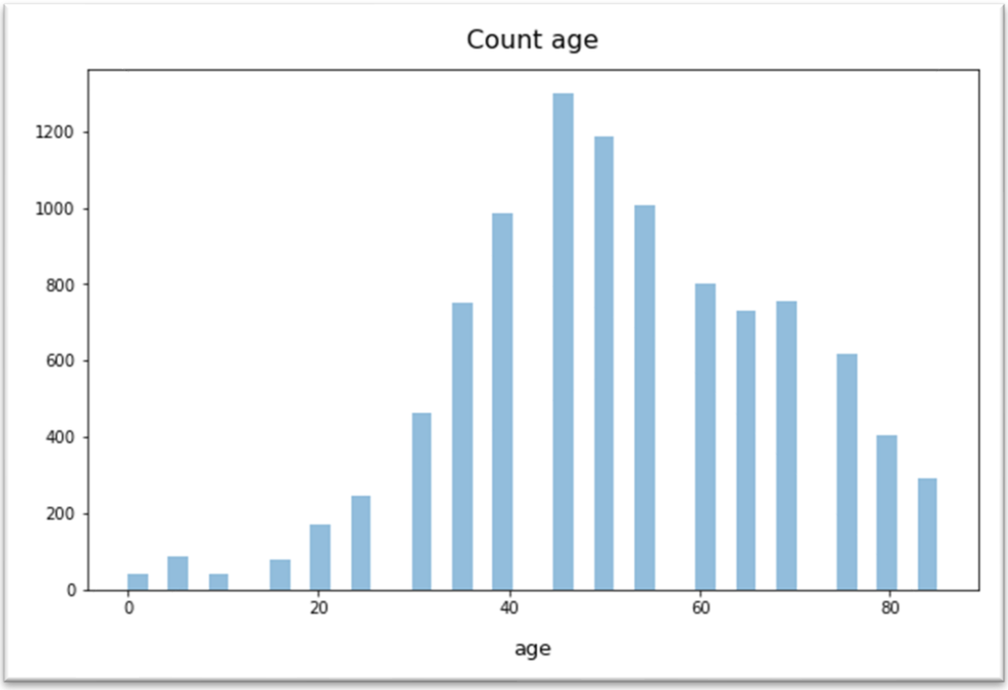

Figure 6. Count of age

A barplot is also used to represent the localization of the skin lesion. The dataset included images of lesions in different parts all over the body, to be able to draw a conclusion as to which are the most likely places for an abnormality to appear. The two places that a skin lesion mostly appeared was the back and the lower extremity.

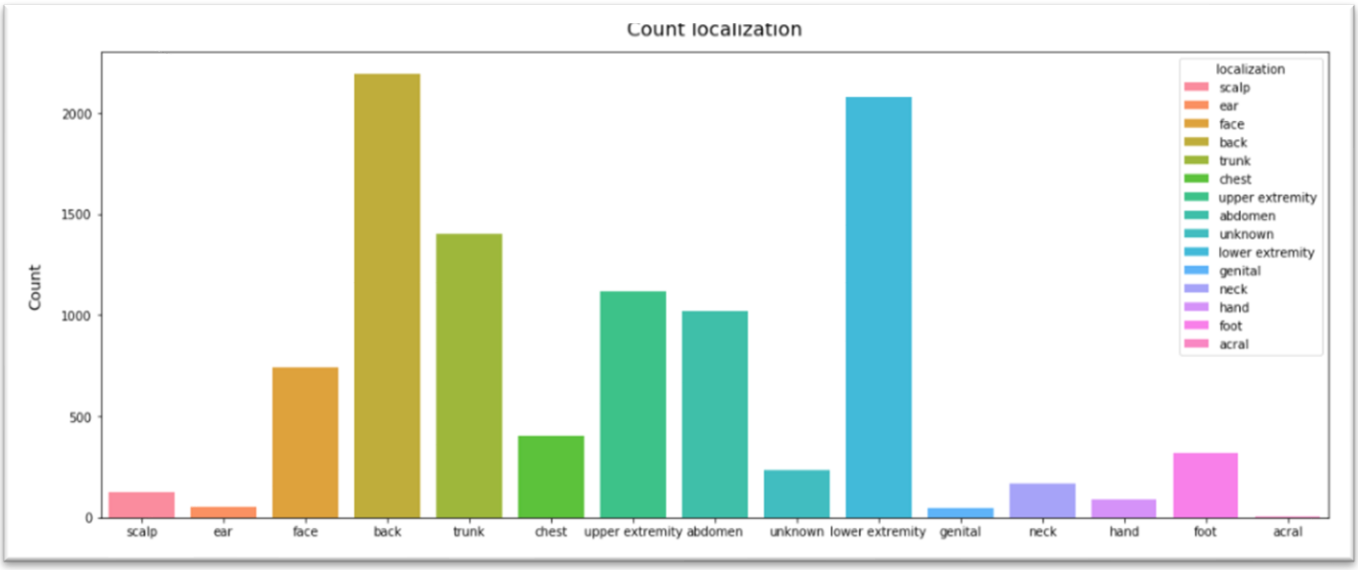

Figure 7. Localization of skin lesions

3.2.2. Pre-process of the data

A resize of the images into 256x192 was necessary to be performed in order to be correctly modified before they were fed in the neural network. Also, the RGB images were converted to grayscale. In the first phase of the experimentation only the training data were used, and they were split into training and test set with a ratio of 90% and 10% respectively, so that the images that would be used for testing would not overlap with the ones that would be used for training. The test set was saved in an npy file, which is a lot faster to process in addition with csv. The remaining training data which is now 90% of original training data will be split further into train and validation set with a ratio of 90% and 10% respectively and these sets are also saved in npy files.

3.2.3. Data Augmentation

The dataset consists of 10015 images. As mentioned above because of the lack of ground truth for test and validation sets only the training set was used and was further split, concluding in making our dataset even smaller. It is commonly known that bigger datasets result in more accurate models but, being able to create such a big dataset can be time consuming and too expensive as well. As convolutional networks have shown great potentials in medical image analysis, they are extensively used in areas such as liver lesion classification and brain scan analysis. To acquire a sample for these analyses, expensive test should be performed such as a computerized tomography. For neural networks in order to achieve an accurate result the size of the dataset must be expanded which would normally mean that too many expensive tests should be performed to acquire the desired dataset. To overcome this problem data augmentation was presented. Data augmentation is also one of the techniques widely used to avoid overfitting in neural networks. By this procedure, we are able to alter and enlarge the dataset in real time, instead of performing this beforehand that would require much more time and even encounter performance issues. More specifically, in this study Keras ImageDataGenerator was used. Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. It was developed with a focus on enabling fast experimentation [32]. ImageDataGenerator, allows the user to perform multiple different transformation in the images live, while they are being fed into the neural network, making the model more robust to slight variations, and saving memory at the same time. In this way the original image is being transformed and copies of each transformation are treated as an entirely new image to be processed by the model [33]. The transformations that were performed for each image are:

Rotation range

Each image is rotated in a range between 0 and 360 degrees as defined by user. By this rotation, some pixels are moved outside the image, so an area is left empty. This is filled according to the input of another variable “fill_mode”. In this study we have used the value “nearest” which basically means that the nearest pixels of the image will be used for the empty area to be filled in.

Shift

Random shifts are also used to move the object. The arguments used were width_shift_range and height_shift_range, which shift the width and height accordingly. Specifically, all image pixels are moved to one direction while the dimension of the image remains the same. The value could be either a float number that represents the percentage between 0 and 1 of the shift, or an integer that represents the number of pixels to be shifted by. In this study a float number of 0.2 was used for both width and height shift.

Flip

A vertical or horizontal flip could be performed to an image that would reverse the rows or columns of the pixels of the image. These two arguments can either be set to True or False, but they should be treated with caution only when it would make sense to perform such a flip according to the object tested. In this study, no flip was performed.

Brightness

An image can be chosen to be darkened or brightened by this argument. A min and a max range of brightness is defined as float numbers. Values lower than 1.0 darken the image and values higher than 1.0 brighten the image. 1.0 precise value has no effect in image. In this study, this argument was not used as all images were converted to grayscale.

Zoom

Another augmentation method is zoom into or out of the image by “zoom_range”. A single float can represent the percentage of the zoom, or an array can be used. If a float is specified, then the range for the zoom will be [1-value, 1+value]. Values above 1.0 will zoom out of the image and values below 1.0 will zoom in the image, while values of exact 1.0 will have no effect to the image. We have used for our experiments a zoom range of 0.2.

Shear

With shear transformation, one angle is fixed, and the image is stretched at a certain angle known as the shear angle. It is used to augment images in a way that the computer perceives the image in the same way as humans do from different angles. In shear_angle, the angle is specified in degrees. In this study a value of 0.2 is used.

There are too many data augmentation methods for a researcher to use from, each one with each own benefits. Some of which are [34]:

Images

- Affine transformations
    - Rotation
    - Scaling
    - Random cropping
    - Reflection
- Elastic transformations
    - Contrast shift
    - Brightness shift
    - Blurring
    - Channel shuffle
- Advanced transformations
    - Random erasing
    - Adding rain effects, sun flare...
    - Image blending
- Neural-based transformations
    - Adversarial noise
    - Neural Style Transfer
    - Generative Adversarial Networks

Audio

- Noise injection
- Time shift
- Time stretching
- Random cropping
- Pitch scaling
- Dynamic range compression
- Simple gain
- Equalization

Natural Language Processing

- Thesaurus
- Text Generation
- Back Translation
- Word Embeddings
- Contextualized Word Embeddings
- Voice conversion

Time Series Data Augmentation

- Basic approaches
  - Warping
  - Jittering
  - Perturbing
- Advanced approaches
  - Embedding space
  - GAN/Adversarial
  - RL/Meta-Learning

In this master Thesis two Convolutional Neural Networks will be used, Densenet and Inception and they will be presented below.

3.3. Densenet

Dense Convolutional Network (Densenet) connects each layer to every other layer in a feed-forward fashion [35]. Densenet uses dense connections between layers through Dense Blocks. It is the extension of Resnet. The logic of Resnet is that its previous layers are merged with future layers. The difference of Densenet is that the network can identify errors between the layers and concatenates the output of the previous layer with the next layer. In contrast with traditional neural networks that only have L connections meaning that each layer is only connected with the future one, Densenet has $\frac{L(L+1)}{2}$ direct connections, meaning each layer, is connected with every other layer (Figure 8).

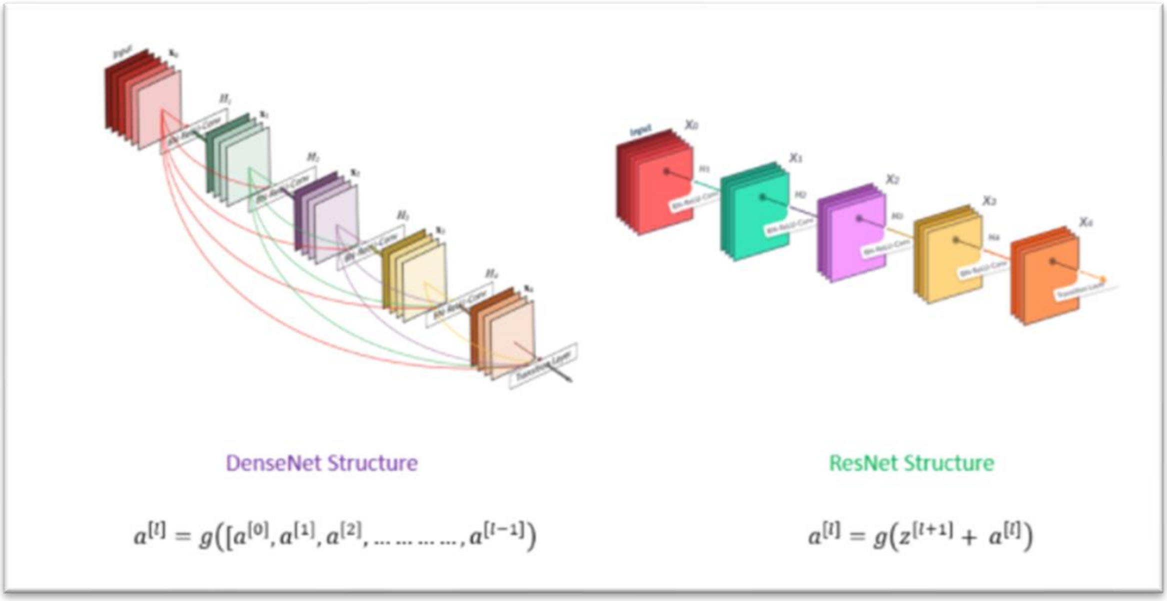

Figure 8. Densenet & Resnet architecture

One of the great benefits and main reason of creation of Densenet is that it overcame the problem of the vanishing gradient in neural networks. Due to the high number of layers and the distance between the input and the output layers, the information was lost. [36]. Densenet also requires fewer parameters than other neural networks such Resnet and there is no need to learn redundant feature maps, it only has 12 filters per layer and only require a small set of new feature-maps. Feature reuse by concatenating feature maps learned by different layers makes the network much more efficient. Lastly, each layer has direct access to the gradients from the loss function, making the network easier to train.

Architecture of Densenet

Dense connectivity: The network comprises L layers, each of which implements a non-linear transformation. $H_{(\ell)}$, where **ℓ** indexes the layer. $H_{(\ell)}$. The $\ell\ th$ layer receives the feature-maps of all preceding layers, x0, . . . , $x_{(\ell-1)}$, as input:

$$x_{(\ell)} = H_{(\ell)}([x0, x1, \ldots, x_{(\ell-1)}]), \quad \textbf{(1)}$$

where [x0, x1, . . . , $x_{(\ell-1)}$] refers to the concatenation of the feature-maps produced in layers 0, . . . , $\ell - 1$

Dense blocks: In dense blocks all layers are directly connected with each other, and feature maps are passed to all subsequent layers. Basically, a dense block consists of multiple convolution blocks. They consist of a group of layers connected to all previous layers. They are mainly used because of the difference of the size of feature maps between the layers. Feature maps between layers have different sizing due to downsampling, so dense blocks are used so as the size of the feature maps will remain the same inside the blocks to perform feature concatenation [35].

Transition Layers: Transition layers are the layers between the dense blocks that are responsible for convolution and pooling. They are used to reduce the complexity of the model. [35]

Growth rate: Since the feature maps are concatenated and not summed up, too many input channels are created. The number of output feature maps of a layer is defined as growth rate. Each layer has access to all previous feature maps and therefore to the collective knowledge of the network and adds feature maps of its own. Thus, the growth rate reflects how much new information each layer adds to the global knowledge [35].

Bottleneck layers: Each layer, especially the later ones, has many input feature maps, even though it only produces k (growth rate) output, because the network is very deep. To improve efficiency, bottleneck layers are introduced. A 1x1 convolution can be inserted before each 3x3 convolution, to reduce the input feature maps. [35]

Compression: Another way to improve efficiency and compactness of the network is to reduce the number of the feature maps in the transition layers. For m feature maps the transition layer generated [θm] output feature maps where $0<\theta<1$ is the compression factor. When $\theta=1$, no change is performed in the number of the input feature maps. [35]

Densenet networks can have different versions based on the number of layers in the network. For example, Densenet-121 is computed from below layers:

- 5 Convolution and Pooling Layers
- 3 Transition layers (6,12,24)
- 1 classification layer (16)
- 2 Denseblock (1x1 and 3x3 conv)

Therefore 5+(6+12+24+16) * 2= 121 [37]

In this thesis Densenet201 will be used.

3.4. Inception

Inception networks were first proposed in Going Deeper with Convolutions paper by Szegedy. The concept of most CNNs was to stack more and more layers to achieve better results. This would eventually result in the need of too much training time and computational resources would increase tremendously. Even if run time and expenses were to be overcome, deep neural networks are prone to overfitting and gradients cannot be easily passed to the entire network. Based on that, inception models came to the surface in the need of better performance without all the heavy cost mentioned before and in order to be used in less powerful machines.

The naïve inception module was based on the idea of networks getting “wider” instead of “deeper”. The author of the inception designed the model to perform convolution with multiple size-kernels at the same time, instead of stacking convolutional layers. In this way, the model can search for features in different sizes in parallel. Figure 9 represents this concept. The 1x1 kernel searches for features at a more localized area than 5x5 that will search for more global features.

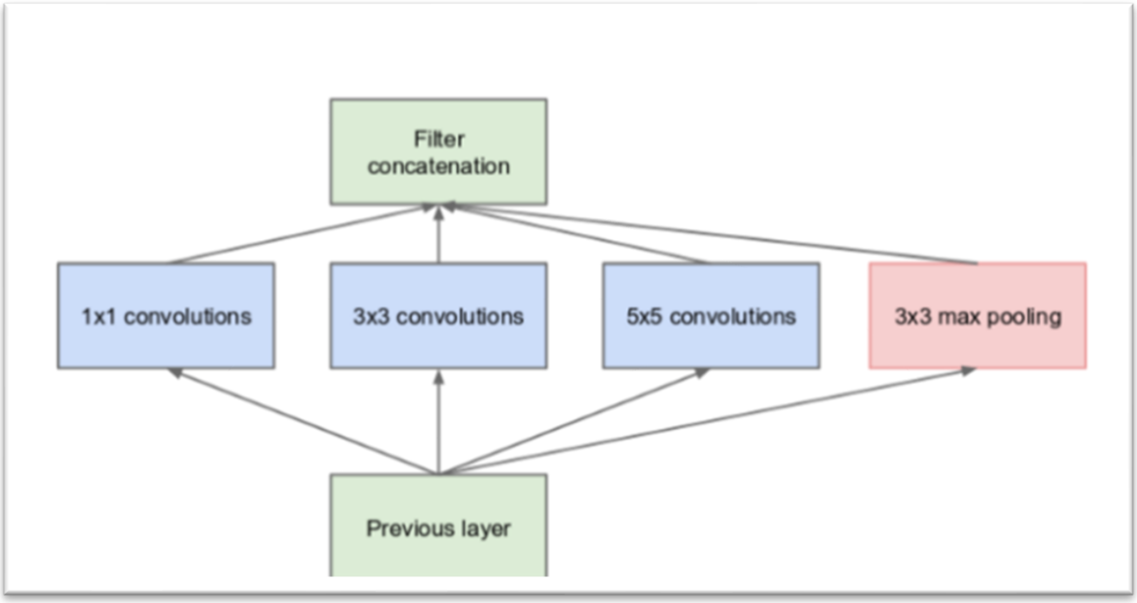

Figure 9. The naïve inception module (Source: “Going Deeper with Convolutions paper” by Szegedy)

In Figure 10, an extra 1x1 kernel is added before the 3x3 and 5x5. These kernels are far cheaper that 3x3 and 5x5 and the goal is to reduce dimensions and allow to use more modules of this type to increase the number of learnable parameters.

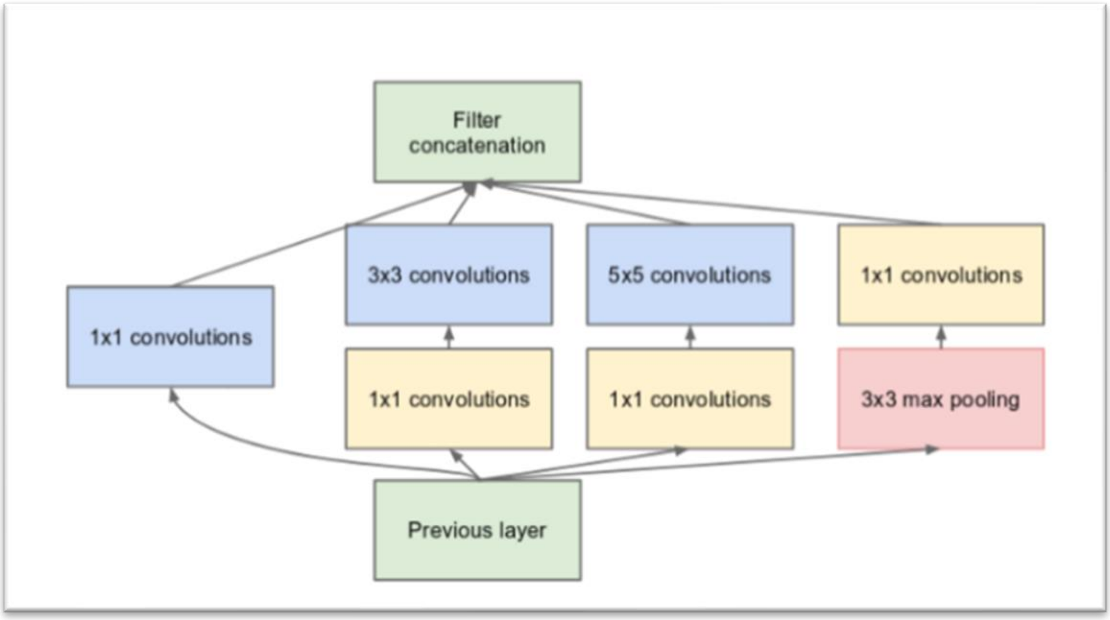

Figure 10. Inception module with dimension reduction (Source: "Going Deeper with Convolutions paper" by Szegedy)

For example, a module of below architecture, has 71704 learnable parameters.

• 1 × 1 convolutions with 8 kernels: 2056 parameters

• 3 × 3 convolutions with 8 kernels: 18440 parameters

• 5 × 5 convolutions with 8 kernels: 51208

With the inception philosophy the above module could be changed into below architecture that would result in a significant reduction in learnable parameters for a total of 10416.

• 1 × 1 convolutions with 8 kernels: 2056 parameters

• 1 × 1 followed by the 3 × 3 convolutions: 2640 parameters

• 1 × 1 followed by the 5 × 5 convolutions: 3664 parameters

• 3 × 3 max pooling followed by the 1 × 1 convolutions: 2056 parameters

An inception network is simply built by stacking lots of those modules one after the other. [38]

3.4.1. Googlenet

Googlenet (Inception V1) was the name of the neural network that won the ILSVRC14 competition. This module used various inception modules, 9 to be specific, stacked linearly. The network has 22 layers when counting only layers with parameters (or 27 layers if we also count pooling). Since the network was considered to be quite deep, the vanishing gradient problem occurred by which the middle layers tended to "die". To overcome this problem the authors introduced the auxiliary classifiers which were connected to these intermediate layers. These classifiers behaved as small convolutional networks put on top of output of two of the Inception modules. The total loss function is a weighted sum of the auxiliary losses evaluated by below formula (the losses of the auxiliary classifiers were weighted by 0.3):

*Total Loss = Cost Function 1 + 0.3 * (Cost Function 2) + 0.3 * (Cost Function 3) (1)*

Final architecture of GoogleNet network is depicted in Figure 11.

The structure of the neural network as designed by the authors is the below:

- An average pooling layer with 5×5 filter size and stride 3, resulting in an 4×4×512 output for the (4a), and 4×4×528 for the (4d) stage.
- A 1×1 convolution with 128 filters for dimension reduction and rectified linear activation.
- A fully connected layer with 1024 units and rectified linear activation.
- A dropout layer with 70% ratio of dropped outputs.
- A linear layer with SoftMax loss as the classifier (predicting the same 1000 classes as the main classifier but removed at inference time).

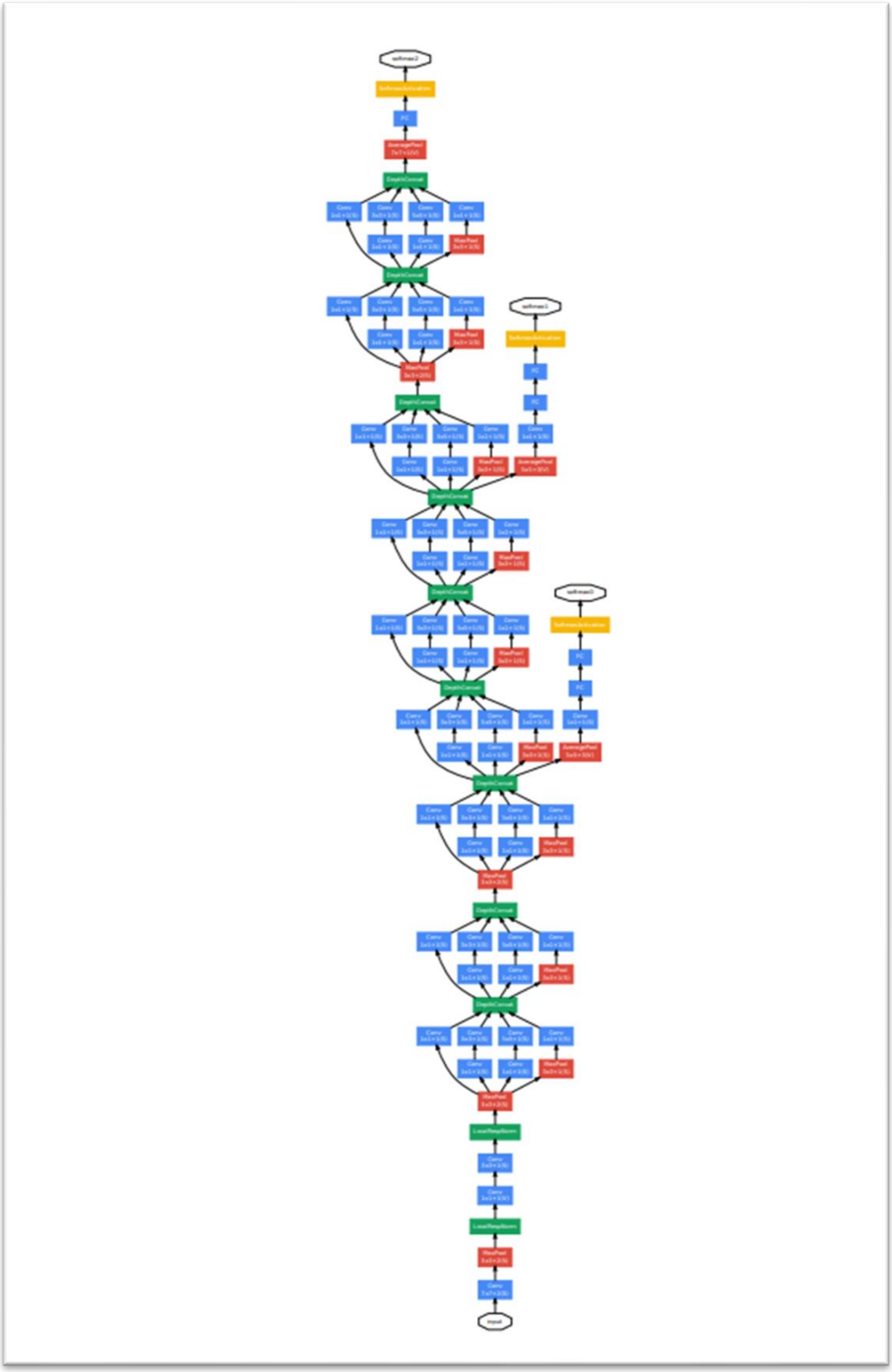

Figure 11. GoogleNet

Apart from GoogleNet which was the first neural network from the Inception family, there were also other versions developed, like Inception v2 which introduced batch normalization (Ioffe and Szegedy 2015) and Inception V3 which introduced additional factorization. In this thesis Inception v3 will be used.

3.4.2. Inception v3

Inception V3 has 42 layers, in comparison to VGG16 which has only 16 and was proven to be much more efficient. Additionally, in Inception v3 the computation cost in only 2.5 higher than that of Googlenet but it also reduces the error rate to 4.2%.

The architecture of Inception V3 is explained in Figure 12

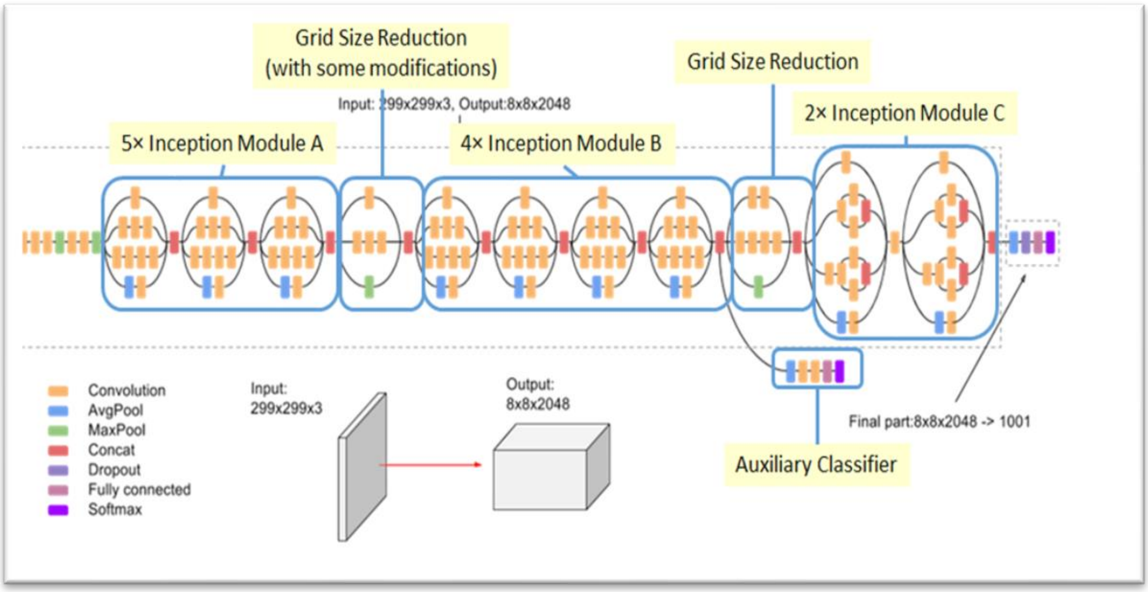

Figure 12. Inception v3 architecture

An Inception v3 neural network consists of the below:

Factorized Convolutions

Factorizing convolutions are used to reduce the number of parameters in the network without reducing the efficiency.

Smaller convolutions

Smaller convolutions are replacing larger ones, since the parameters used are less and the training is much faster. For example, a 5x5 convolution has 25 parameters, while two 3x3 convolutions have 19 parameters.

Asymmetric convolutions

Instead of replacing a larger convolution with a smaller one, in some cases replacing larger convolution with an asymmetric one can prove much more efficient as parameters are less. For example, is it more efficient to replace a 3x3 convolution with a 1x3 convolution and a 3x1 convolution after that, than to replace it with two 2x2.

Auxiliary classifier

Auxiliary classifier are architectural components, which attach to layers before the end of the network that drive gradient layers to the lowest layers, thereby fighting the vanishing gradient problem and improving efficiency in training. Auxiliary classifiers were also used in Googlenet, to make the network deeper. In inception V3 one auxiliary classifier is used on top of the last 17x17 layer, but this time it acts as a regularizer.

Grid size reduction

Grid size reduction is normally done by max pooling. But since this has caused problems, like the procedure being too expensive a new more efficient grid size reduction was proposed in Inception V3 as 320 feature maps are done by convolution with stride 2 and 320 are done by max pooling.

All above are visible in Figure 12 where Inception v3 architecture is introduced [39] [40].

Chapter 4: Experiments and Results

4.1. Transfer learning

Transfer learning is popular in deep-learning, due to the fact, that neural networks require too much time and too many resources to be trained or because the data available for the analysis are not sufficient. In this concept, this machine learning method was developed, where pre-trained neural networks are used as the starting point on computer vision. These pre-trained models are used in different but related problems. One, can use the pre-trained model without any fine- tuning, or can choose to tune some of the layers and keep the rest untouched, in order for the weights to adjust accordingly to best fit the problem in hand. In this thesis, transfer learning was used, and models were fine-tuned for better results

4.2. Densenet

Pre-trained Model Densenet201 was used, with weights of ImageNet and *Include_top= False*, in order to exclude the top layers of the neural network. For the last layers, a Globalmaxpooling layer is added in order to Flatten the output layer and fully connected layer with 512 hidden units with ‘relu’ as activation function. A dropout rate of 0.7 is used to avoid overfitting as well as a sigmoid layer for classification with SoftMax function. The model is then compiled and an Adam optimizer with learning_rate=0.0001 is used. A model summary is presented in Figure 13.

```
conv5_block32_concat (Concatena (None, 6, 8, 1920)   0           conv5_block31_concat[0][0]
                                                                 conv5_block32_2_conv[0][0]
__________________________________________________________________________________________________
bn (BatchNormalization)         (None, 6, 8, 1920)   7680        conv5_block32_concat[0][0]
__________________________________________________________________________________________________
relu (Activation)               (None, 6, 8, 1920)   0           bn[0][0]
__________________________________________________________________________________________________
global_max_pooling2d_1 (GlobalM (None, 1920)         0           relu[0][0]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, 512)          983552      global_max_pooling2d_1[0][0]
__________________________________________________________________________________________________
dropout_1 (Dropout)             (None, 512)          0           dense_1[0][0]
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 7)            3591        dropout_1[0][0]
==================================================================================================
Total params: 19,309,127
Trainable params: 1,216,199
Non-trainable params: 18,092,928
__________________________________________________________________________________________________
```

Figure 13. Model Summary for Densenet201

To increase the number of samples in the dataset, as mentioned is previous chapters, data augmentation is used with following parameters:

- rotation_range=60
- width_shift_range=0.2
- height_shift_range=0.2
- shear_range=0.2
- zoom_range=0.2

4.2.1. Re-train whole model

In first attempt, the experiment was to retrain the whole model instead of using transfer learning with following configuration:

- Set layer_trainable=True
- Add a ReduceLROnPlateau to monitor the validation accuracy.

The new model summary is presented in Figure 14.

```
conv5_block32_concat (Concatena (None, 6, 8, 1920)  0           conv5_block31_concat[0][0]
                                                                 conv5_block32_2_conv[0][0]
__________________________________________________________________________________________________
bn (BatchNormalization)         (None, 6, 8, 1920)  7680        conv5_block32_concat[0][0]
__________________________________________________________________________________________________
relu (Activation)               (None, 6, 8, 1920)  0           bn[0][0]
__________________________________________________________________________________________________
global_max_pooling2d_1 (GlobalM (None, 1920)        0           relu[0][0]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, 512)         983552      global_max_pooling2d_1[0][0]
__________________________________________________________________________________________________
dropout_1 (Dropout)             (None, 512)         0           dense_1[0][0]
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 7)           3591        dropout_1[0][0]
==================================================================================================
Total params: 19,309,127
Trainable params: 19,080,071
Non-trainable params: 229,056
__________________________________________________________________________________________________
```

Figure 14.Model Summary for Densenet201 after fine-tuning

For the training of the neural network to start, a batch size =32 is set and epochs=30.

The validation accuracy accomplished of the above neural network is 0.883592 and loss is 0.588912. A list of training and validation accuracy, as well as accuracy and validation loss are presented in Figure 15.

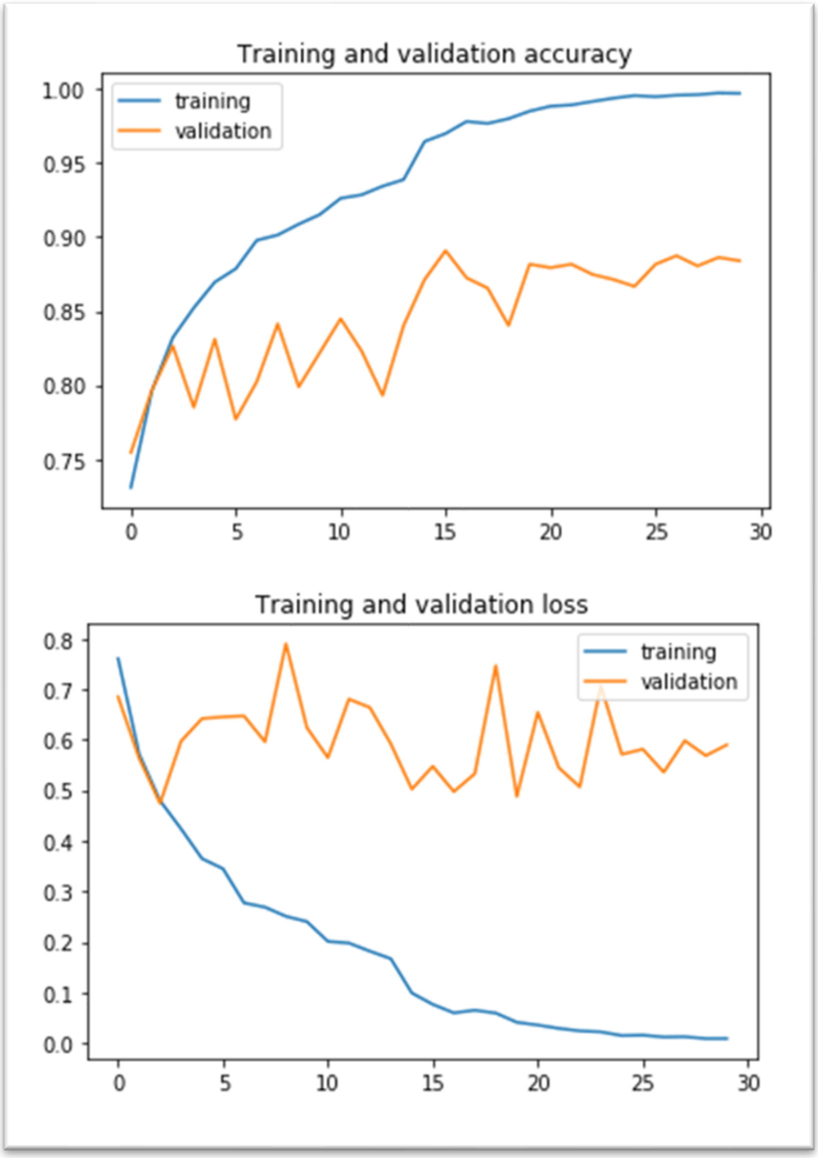

Figure 15. Accuracy and loss history per epoch for Densenet201 train whole model

The model is then saved in a json file and the weights in an h5 file to be used in the future without the need to retrain the model.

The saved model is used in test set and the accuracy accomplished is 0.8912 18 and loss is 0.563957.

Although the accuracy accomplished by Densenet is very satisfying, it is obvious from the figure 15 that the neural network overfitted in the training data. For this reason, in the second attempt instead of retraining the whole network, transfer learning was used.

4.2.2. Transfer learning from pretrained model

In the first experiment Densnet201 pre-trained model was used, same way as above, but it was chosen only a few layers to be trainable and not the whole model. Layers above 481 were set as Trainable=True in order to Fine tune the model. The model was again trained with a batch size of 32 and for 30 epochs.

The validation accuracy accomplished of the above neural network is 0.853659 and loss is 0.661591. A list of training and validation accuracy, as well as accuracy and validation loss are presented in Figure 16.

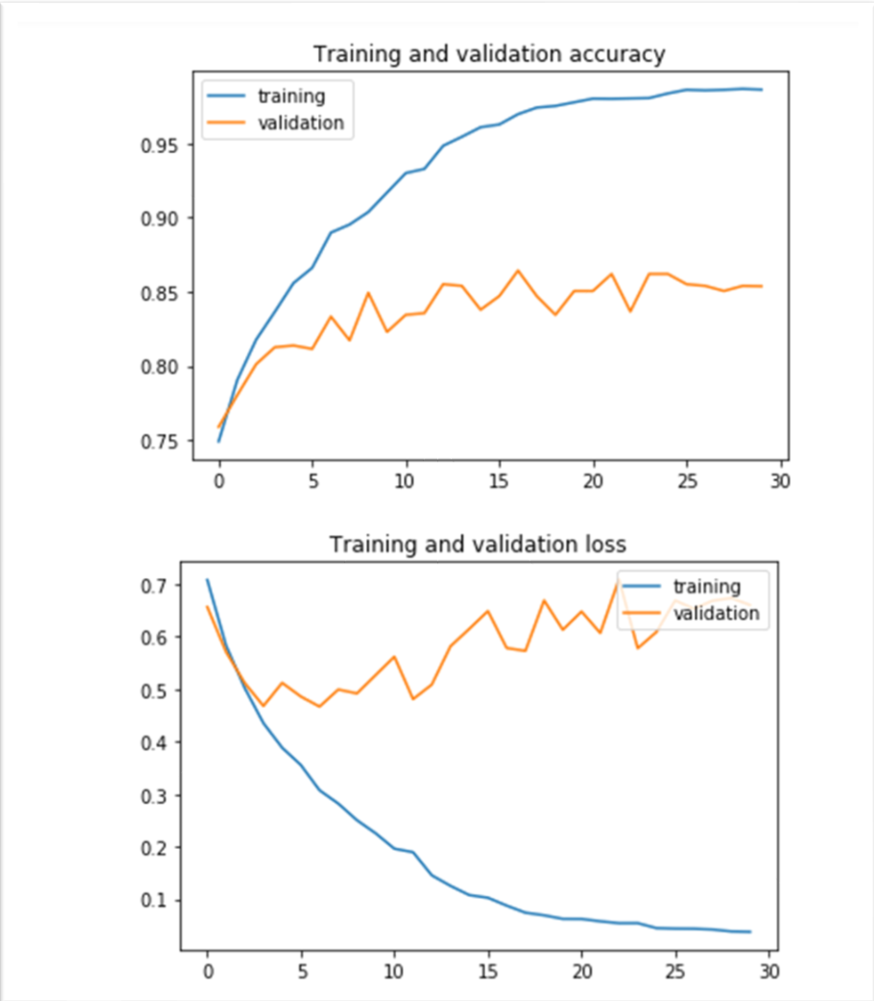

Figure 16. Accuracy and loss history per epoch for Densenet201 using transfer learning

The model is then saved in a json file and the weights in an h5 file in order to be used in the future without the need to retrain the model.

The saved model is used in test set and the accuracy accomplished is 0.853293 and loss is 0.661162.

In the second experiment, the pretrained model was used with weights of Imagenet and Include_top= False, in order to exclude the top layers of the neural network. For the last layers, a Globalmaxpooling layer is added in order to Flatten the output layer and fully connected layer with 512 hidden units with 'relu' as activation function. A dropout rate of 0.5 is used to avoid overfitting as well as a sigmoid layer for classification with SoftMax function. The model is then compiled and an Adam optimizer with learning_rate=0.0001 is used. To avoid overfitting early stopping is introduced. Training is stopped at the point where performance on the validation set starts to degrade. The performance is measured by validation loss. Another technique used is model checkpoint, in order to be able to continue training the model in a later time. A checkpoint contains the architecture of the model, the weights of the model, the training configuration, and the state of the optimizer, all of which are essential to resume the training of the model from the exact point it was stopped. In this specific experiment it was mostly used to save the model instead of resuming the training at a later time. Lastly, ReduceLROnPlateau was used to monitor the validation accuracy.

In order to increase the dataset, the same method as for Densnet201 is used, data augmentation with same parameters:

- rotation_range=180
- width_shift_range=0.1
- height_shift_range=0.1
- zoom_range=0.2
- shear_range=0.2
- horizontal_flip=True
- vertical_flip=True

As previously, models were trained for 30 epochs and results showed overfitting occurred, for this model it was chosen to only train for 15 epochs with a batch size of 32 and save only the best model. The validation accuracy accomplished of the above neural network is 0.821508 and loss is 0.650849.

In test set and the accuracy accomplished is 0.830339 and loss is 0.638730.

The model is saved in json format to be used later in an ensemble model.

4.3. Inception v3

4.3.1. Retrain whole model

The second neural network that will be used is InceptionV3. The neural network was retrained without using transfer learning. Top layers were excluded and a Globalmaxpooling layer is added in order to Flatten the output layer and afterwards a fully connected layer with 512 hidden units with ‘relu’ as activation function. A dropout rate of 0.5 is used to avoid overfitting as well as a sigmoid layer for classification with SoftMax function. The model is then compiled and an Adam optimizer with learning_rate=0.0001 is used. A model summary is presented in Figure 17.

```
activation_94 (Activation)      (None, 4, 6, 192)   0          batch_normalization_94[0][0]
________________________________________________________________________________
mixed10 (Concatenate)           (None, 4, 6, 2048)  0          activation_86[0][0]
                                                               mixed9_1[0][0]
                                                               concatenate_2[0][0]
                                                               activation_94[0][0]
________________________________________________________________________________
global_max_pooling2d_1 (GlobalM (None, 2048)        0          mixed10[0][0]
________________________________________________________________________________
dense_1 (Dense)                 (None, 512)         1049088    global_max_pooling2d_1[0][0]
________________________________________________________________________________
dropout_1 (Dropout)             (None, 512)         0          dense_1[0][0]
________________________________________________________________________________
dense_2 (Dense)                 (None, 7)           3591       dropout_1[0][0]
================================================================================
Total params: 22,855,463
Trainable params: 1,069,895
Non-trainable params: 21,785,568
```

Figure 17. Model Summary for InceptionV3

In order to increase the dataset, the same method as for Densnet201 is used, data augmentation with same parameters:

- rotation_range=60
- width_shift_range=0.2
- height_shift_range=0.2
- shear_range=0.2
- zoom_range=0.2

In order for the training of the neural network to start, a batch size =64 is set and epochs=30 same as Densenet201 for comparison.

The validation accuracy accomplished of the above neural network is 0.712860 and loss is 0.797383, which is far worse that Densenet201, but in this case the data did not overfit. A list of training and validation accuracy, as well as accuracy and validation loss are presented in Figure 18.

Figure 18. Accuracy and loss history per epoch for Inception V3

The model is then saved in a json file and the weights in an h5 file in order to be used in the future without the need to retrain the model.
The saved model is used in test set and the accuracy accomplished is 0.7128 60 and loss is 0.797383.

4.3.2. Transfer learning from pretrained model

In the next experiment transfer learning was used with weights of Imagenet and Include_top= False, in order to exclude the top layers of the neural network. For the last layers, a Convolution layer is added followed by a MaxPooling layer to flatten the output layer to 1 dimension. A dropout rate of 0.4 is used to avoid overfitting and a Flatten layer and 'relu' as activation function. A dropout rate of

0.4 is then added and lastly a layer for classification with SoftMax function. The model is then compiled and an Adam optimizer with learning_rate=0.0001 is used. In order to avoid overfitting early stopping is introduced. Training is stopped at the point where performance on the validation set starts to degrade. The performance is measured by validation loss. Model checkpoint is used in order to save the model. Lastly, ReduceLROnPlateau was used to monitor the validation accuracy.

In order to increase the dataset, the same method as for Densnet201 is used, data augmentation with same parameters:

- rotation_range=60
- width_shift_range=0.2
- height_shift_range=0.2
- shear_range=0.2
- zoom_range=0.2

The model was trained for 20 epochs with a batch size of 64. The model started to reduce its learning rate from epoch 13 and again in epoch 15 and 18. The validation accuracy accomplished of the above neural network is 0.874723 and loss is 0.546727.

In test set and the accuracy accomplished is 0.863273 and loss is 0.646843.

The model is saved in json format in order to be used later in an ensemble mo del.

4.4. Ensemble model

Ensemble learning is a machine learning technique, that combines more than one models to make a prediction. The models used for machine learning can be of different type and can also be trained on different data. Although ensemble methods can be more complex and expensive to compute, they can achieve better predictions, than a single model and reduce the spread of dispersion. [41]

However, ensemble model may not always lead to better results than a single model. It depends on the models used, the dataset and other factors.

In this thesis, an ensemble model including Densenet201 and InceptionV3 is used. As mentioned before the accuracy of each single model has been calculated and the models have been saved for later use in order to avoid train the models again from scratch. By using these saved models and inserting them in an ensemble model the validation accuracy achieved is 0.881375 and validation loss 0.428316.

In the test set the accuracy achieved was 0.888224 and the loss = 0.428636.

4.5. Results

In Table 8, there is a representation of all models, with the validation accuracy achieved as well as the validation loss for each model.

Model	Densnet201	Inception V3	Ensemble	Densnet201 -Transfer learning	InceptionV3 -Transfer learning	Densnet201-Transfer learning_Save best model
Val_Accuracy	**88.36%**	71.29%	88.14%	85.36%	87.47%	82.15%
Val_Loss	**0.588912**	0.797383	0.428316	0.661591	0.546727	0.650849

Table 8.Validation set results for all Models: Densenet, Inception, Ensemble

In Table 9, there is a representation of all models, with the test accuracy achie ved as well as the test loss for each model.

Model	Densnet201	InceptionV3	Ensemble	Densnet201 -Transfer learning	InceptionV3 -Transfer learning	Densnet201-Transfer learning_Save best model
Test_Accuracy	**89.12%**	71.86%	**88.82%**	85.33%	86.33%	83.03%
Test_Loss	**0.564**	0.7951	**0.4286**	0.661162	0.646843	0.638730

Table 9.Test set results for all Models: Densenet, Inception, Ensemble

Although transfer learning technique seemed to decrease accuracy and increase loss for Densenet201, for InceptionV3 convolutional neural network the exact opposite happened. This model performs better with Transfer learning than training the whole network from scratch. Also, when trying to use method

Save_best_model for Densnet201 with Transfer learning to avoid overfitting, the model lost both in accuracy and loss in comparison with training the whole neural network.

Taking all result experiments under consideration, it appears Densenet201 is the model that achieved the highest accuracy, and that ensemble model has achieved the lowest loss. Those two metrics do not always align as they measure different things.

Accuracy measures the performance of the model as:

$$Accuracy = \frac{\text{No of correct predictions}}{\text{Total no of predictions}}$$

Loss on the other hand is defined as the difference between the predicted value of the model and the true value. The loss function used in this thesis is categorical cross-entropy. Cross entropy is defined as:

$$\text{Cross-entropy} = -\sum_{i=1}^{n}\sum_{j=1}^{m} y_{i,j}\log(p_{i,j})$$

So, when these two measures do not align it is a matter of deciding which measure values the most, for the specific analysis in hand and choose accordingly. If accuracy is considered more import, Densenet Model is the best model. However, the accuracy of Ensemble model is quite close to the accuracy of the Densenet model and loss is significantly lower than Densenet. On the other hand, it is clear that Inception v3 scored the worse results having the lowest accuracy and the highest loss. In conclusion, Ensemble is considered to be the best model and it is the one used in the application presented next.

4.6. Application

An application has been developed where the user can use any saved convolutional model, load a new image, and get a prediction of the type of skin lesion the image represents. The app has a model predefined that will be used for the prediction and by default only checks for files with extensions” .png” or “jpg” to ensure only images will be uploaded. When the app is executed, a window opens prompting the user to select an image for upload from local file system.

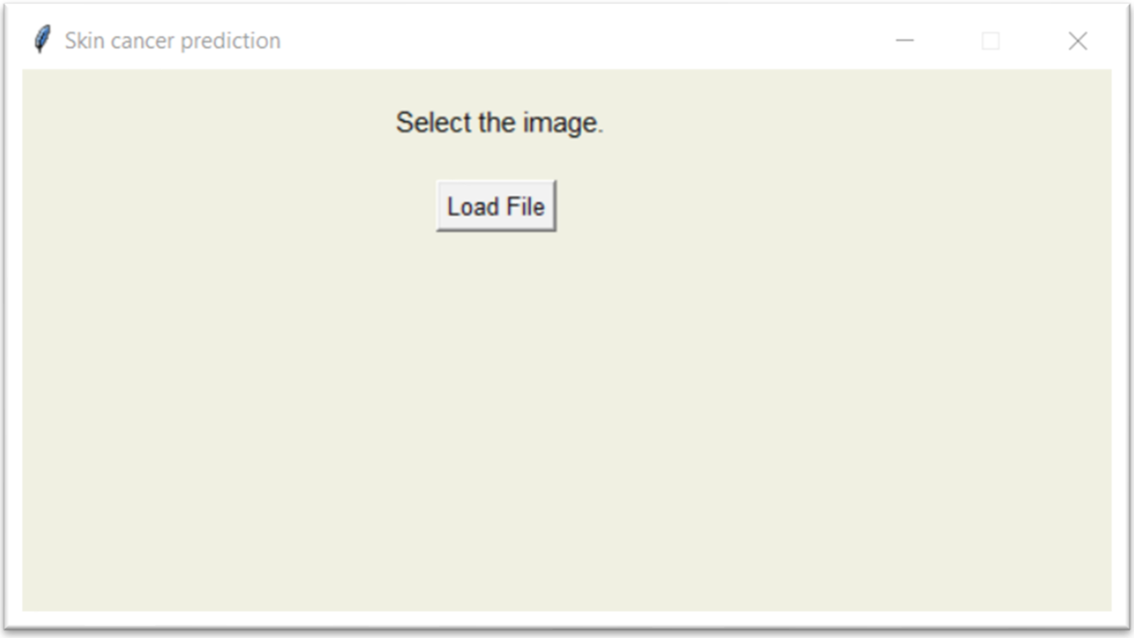

Figure 19. Application developed

The user selects button “Load file” and chooses of an image to be loaded. After a few seconds, the path of the image selected is represented as well as the name of the skin lesion predicted by the neural network.

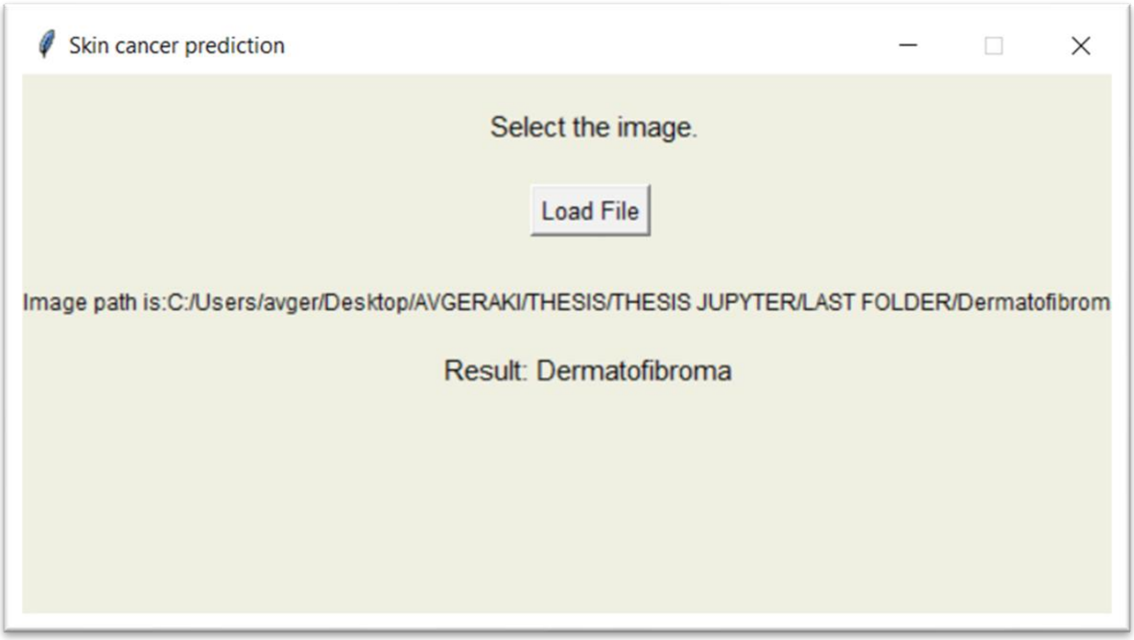

Figure 20. Result of application prediction

Chapter 5: Conclusion and Future Work

5.1. Conclusion

5.1.1. Comparison with first paper

In table 10 a comparison between the first paper and my research is represented. In the first paper Densenet network was used in multiple variations. In Densenet the results of my research where I retrained the whole model are close to the ones of the paper that used Transfer learning, while my attempt to use transfer learning performed worse. This could be a result of different pre-process method like the data augmentation that was performed, and the different parameters used in the model. Densenet-201 of my research is the one which achieved the best accuracy amongst all models, but since it was subject to overfitting, the ensemble would be the next best model.

Evaluation Metrics	Accuracy of Paper	Accuracy of my research
Densenet 161	87.4%	
Densenet 201- Transfer learning	**88.3%**	85.36%
Densenet 161- nestrov	82.6%	
Densenet 201- nestrov	82.7%	
Densenet 169- nestrov	85.7%	
Densenet 121- nestrov	84.7%	
DenseNet201-retrain while network		**88.36%**
InceptionV3		71.29%
InceptionV3-Transfer learning		87.47%
Densenet201-Transfer learning_Save best model		82.15%
Ensemble		**88.14%**

Table 10. Comparison between first paper and this thesis

5.1.2. Comparison with second paper

In table 11 a comparison between the second paper and my research is represented. In the second paper only one algorithm was used, Densenet-121 which was altered by blocks and layers reduction which ultimately resulted in an accuracy of 89.2%. The pre-process of the images as well as the changes in the network's architecture proved beneficial and this model outperformed all of the models used in my thesis.

Evaluation Metrics	Accuracy of Paper	Accuracy of my research
DenseNet121	**89.2%**	
DenseNet201		88.36%
InceptionV3		71.29%
InceptionV3-Transfer learning		87.47%
Densenet201-Transfer learning		85.36%
Densenet201-Transfer learning_Save best model		82.15%
Ensemble		88.14%

Table 11. Comparison between second paper and this thesis

5.1.3. Comparison with third paper

In table 12 a comparison between the second paper and my research is represented. In this paper the researchers also introduced an ensemble model of 5 neural networks. Densnet-201 appears to have achieved the best accuracy in both researches. In contrast there is high difference between Inception V3 of the paper and my research. Although all of the models of the thesis show high accuracy when combining these 5 models the accuracy achieved is slightly lower than combining only the 2 as presented in my thesis.

Evaluation Metrics	Accuracy of Paper	Accuracy of my research
Inception V3	87.7%	71.29%
Xception	86.2%	
InceptionResNet v2	87.8%	
ResNet 50 v1	85.2%	
DenseNet 201	88.2%	88.36%
Ensemble of 5	85.2%	
InceptionV3-Transfer learning		87.47%
Densenet201-Transfer learning		85.36%
Densenet201-Transfer learning_Save best model		82.15%
Ensemble of 2		88.14%

Table 12. Comparison between third paper and this thesis

5.2. Future Work

In order to achieve a better accuracy, an increase in the dataset could be beneficial as more data would be available and neural network could be better trained. History has shown that few data could lead to poor results and although HAM10000 dataset does have many images for a neural network to be trained, an upgrade could be to combine two different datasets for even more images to be available.

Another action would be to use newer neural networks for skin lesion classification like EfficientNet family, which is fast growing and very promising neural networks as they have achieved and outperformed many neural networks in terms of accuracy and efficiency.

Lastly, a better accuracy and avoidance of overfitting could be achieved by performing a different pre-process. In data augmentation different parameters could be used and also data augmentation could also be performed in test data to increase the test set. Also, some pre-process in isolating only the skin lesion of the image and excluding the surrounding could prove beneficial.

Bibliography

[1] "National Geographic," January 2017. [Online]. Available: https://www.nationalgeographic.com/science/health-and-human-body/human-body/skin/.

[2] M. Dyzma, "Machine Learning in Medicine: Predicting Skin Disease with Data," 2018. [Online]. Available: https://www.netguru.com/blog/skin-cancer-detection-using-image-processing.

[3] Haenssle et al., "Man against machine: diagnostic performance of a deep learning convolutional neural network for dermoscopic melanoma recognition in comparison to 58 dermatologists," 28 May 2018. [Online]. Available: https://www.annalsofoncology.org/article/S0923-7534(19)34105-5/pdf.

[4] American Academy of Dermatology, [Online]. Available: https://www.aad.org/public/diseases/skin-cancer/types/common.

[5] American Cancer Society, "“Cancer Facts and Figures 2020”," Atlanta, 2020.

[6] D. S. Sarnoff, "Skin Cancer 101," February 2020. [Online]. Available: https://www.skincancer.org/skin-cancer-information/.

[7] F. D. A. M. R. B. Alon Scope, "Clinical examination," 2020. [Online]. Available: https://dermoscopedia.org/w/index.php?title=Clinical_examination&oldid=17029.

[8] Sonthalia, Sidharth; Yumeen, Sara; Kaliyadan., Feroze, Dermoscopy Overview and Extradiagnostic Applications, 2020.

[9] "What is dermoscopy," 2012. [Online]. Available: https://www.healio.com/news/dermatology/20120620/what-is-dermoscopy.

[10] A. Sharma, "Differences Between Machine Learning & Deep Learning," 15 November 2018. [Online]. Available: https://www.datacamp.com/community/tutorials/machine-deep-learning?utm_source=adwords_ppc&utm_campaignid=898687156&utm_adgroupid=48947256715&utm_device=c&utm_keyword=&utm_matchtype=b&utm_network=g&utm_adpostion=&utm_creative=255798340456&utm_targetid=aud-.

[11] "AI Technologies," [Online]. Available: https://searchenterpriseai.techtarget.com/definition/AI-Artificial-Intelligence.

[12] "Artificial Intelligence and Machine Learning: Policy Paper," 18 April 2017. [Online]. Available: https://www.internetsociety.org/resources/doc/2017/artificial-intelligence-and-machine-learning-policy-paper/?gclid=Cj0KCQjwrIf3BRD1ARIsAMuugNtxpBk3Y2Ou8g0OlF2-284-0rkuuwPp2Lhuj8gDglPG-TSB_cnFV_IaArx3EALw_wcB.

[13] Y. Gavrilova, "Artificial Intelligence vs. Machine Learning vs. Deep Learning: Essentials," 8 April 2020. [Online]. Available: https://serokell.io/blog/ai-ml-dl-difference.

[14] P. Vaish, "10 groups of Machine Learning Algorithms," [Online]. Available: https://adataanalyst.com/machine-learning/10-groups-machine-learning-algorithms/.

[15] J. Brownlee, "A Tour of Machine Learning Algorithms," 12 August 2019. [Online]. Available: https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/.

[16] [Online]. Available: https://en.wikipedia.org/wiki/Decision_tree_learning.

[17] G. P.-S. C. J. M. William J. Frawley, "Knowledge Discovery in Databases: An Overview," *AI Magazine,* 1991.

[18] N. D. A. D. E. P. Athanasios Voulodimos, "Deep Learning for Computer Vision: A Brief Review," 01 February 2018. [Online]. Available: https://www.hindawi.com/journals/cin/2018/7068349/#copyright.

[19] "A Beginner's Guide to Convolutional Neural Networks (CNNs)," [Online]. Available: https://pathmind.com/wiki/convolutional-network.

[20] F. Kunihiko, "Necognitron," 2017. [Online]. Available: https://doi.org/10.4249%2Fscholarpedia.1717.

[21] A. Deshpande, "A Beginner's Guide To Understanding Convolutional Neural Networks," 2016.

[22] C. S. V. H. G. S. M. C. Shaik Khadar Sharif, "Sign Language Recognition," *International Journal of Engineering Research & Technology (IJERT),* vol. 9, no. 05,May-2020, 2020.

[23] Qiuhong Ke et al, "Computer Vision for Human–Machine Interaction," Elsevier, 2018, pp. 127-145.

[24] A. Z. Karen Simonyan, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 4 September 2014. [Online]. Available: https://arxiv.org/abs/1409.1556.

[25] X. Z. S. R. J. S. Kaiming He, "Deep Residual Learning for Image Recognition," 10 December 2015. [Online]. Available: https://arxiv.org/abs/1512.03385.

[26] W. W. Shima Nofallah, "Transfer Learning for Automatic Disease Diagnosis," Seattle, WA, 2018.

[27] A. C. C. D. Pierluigi Carcagn, "A Dense CNN approach for skin lesion," Italy , 2018.

[28] E. B. A. S. J. D. a. N. P. Nicholas Heller, "Computer Aided Diagnosis of Skin Lesions from," University of Minesota, 2018.

[29] "Tutorialspoint - Google colab," [Online]. Available: https://www.tutorialspoint.com/google_colab/what_is_google_colab.htm.

[30] R. C. &. K. H. Tschandl P., "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. Sci. Data 5, 180161 doi.10.1038/sdata.2018.161 (2018)".

[31] V. R. P. T. M. E. C. S. D. D. G. B. H. A. K. K. L. M. M. H. K. a. A. H. Noel Codella, "“Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC)”, 2018; arXiv:1902.03368.".

[32] "Keras," [Online]. Available: https://keras.io/about/.

[33] A. Bhandari, "Image Augmentation on the fly using Keras ImageDataGenerator!," Analytics Vidya, 2020.

[34] M. G. Agnieszka Mikołajczyk, "Data augmentation for improving deep learning in image classification problem," IEEE, Swinoujście, Poland, 2018.

[35] Z. L. L. v. d. M. K. Q. W. Gao Huang, "Densely Connected Convolutional Networks," 2018.

[36] Gao Huang, Zhuang Liu, Kilian Q. Weinberger, "Densely Connected Convolutional Networks," 2016.

[37] G. Singhal, "Introduction to DenseNet with TensorFlow," 2020.

[38] U. Michelucci, Advanced Applied Deep Learning: Convolutional Neural Networks and Object Detection (pp. 131-135), 2019.

[39] V. KURAMA, "A Review of Popular Deep Learning Architectures: ResNet, InceptionV3, and SqueezeNet," 2020. [Online]. Available: https://blog.paperspace.com/popular-deep-learning-architectures-resnet-inceptionv3-squeezenet/.

[40] V. V. S. I. J. S. Christian Szegedy, "Rethinking the Inception Architecture for Computer Vision".

[41] J. Brownlee, "Why Use Ensemble Learning?," Machine learning Mastery, 26 October 2020. [Online]. Available: https://machinelearningmastery.com/why-use-ensemble-learning/.

[42] [Online]. Available: https://www.internetsociety.org/resources/doc/2017/artificial-intelligence-and-machine-learning-policy-paper/?gclid=Cj0KCQjwrIf3BRD1ARIsAMuugNtxpBk3Y2Ou8g0OlF2-284-0rkuuwPp2Lhuj8gDglPG-TSB_cnFV_IaArx3EALw_wcB.