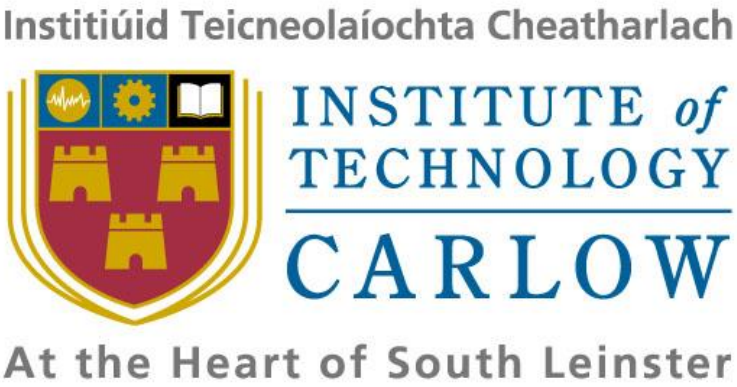
MACHINE LEARNING IN DERMATOLOGY:
USING CONVOLUTIONAL NEURAL NETWORKS TO CLASSIFY MELANOMA AND OTHER SKIN LESIONS



STUDENT NAME: Laura Dempsey

COURSE NAME: M.Sc. Data Science

DEPARTMENT: Department of Computing and Networking

SUPERVISOR: Dr. Oisin Cawley

SUBMISSION DATE: 01-08-2019

ACKNOWLEDGEMENTS

I would like to thank my thesis supervisor, Dr. Oisin Cawley, for his help and guidance throughout this research. I would like to extend this gratitude to the staff and lecturers from the Department of Computing and Networking in IT Carlow for all the skills and knowledge I have gained during my time here. I wish to acknowledge the support and encouragement from my family, friends and partner, without which this research would not be possible.

Table of Contents

LIST OF ILLUSTRATIONS 6
LIST OF ABBREVIATIONS 7
1 INTRODUCTION 9
2 PROBLEM STATEMENT 12
3 LITERATURE REVIEW 14
3.1 Chapter Overview 14
3.2 Skin Cancer 14
3.3 UVR and skin cancer 15
3.4 Diagnosis and treatment of skin cancer 16
3.5 Computer-aided diagnostic systems for Skin Cancer 17
3.6 Before AI and Machine Learning: Early Pattern Recognition Techniques 17
3.6.1 Template Matching 18
3.6.2 Statistical classification 18
3.6.3 Syntactic matching 18
3.7 Artificial Intelligence 19
3.8 Machine Learning 20
3.9 Supervised and Unsupervised learning 20
3.10 Image Processing for Melanoma detection 21
3.10.1 Image pre-processing 21
3.10.2 Feature extraction 22
3.10.3 Feature selection 22
3.10.4 Classification 23
3.11 Support Vector Machines (SVM) 23
3.12 K-Nearest Neighbour (KNN) 23
3.13 Artificial Neural Networks 24
3.13.1 Biological Neural Networks 24
3.13.3 Activation functions in neural networks 26
3.13.4 Hidden layers 26
3.13.5 Back propagation and stochastic gradient descent in neural networks 27
3.14 A move towards Convolutional Neural Networks in Image Processing 28
3.15 Convolutional Neural Networks 29
3.15.1 Convolutional layer 29
3.15.2 Activation Layer 30
3.15.3 Pooling layer 31
3.15.4 Fully connected layers and classification 31

3.15.5 Fully convolutional neural networks 32
3.16 Transfer Learning 32
3.17 Popular pre-trained models for Transfer Learning 34
3.18 Hyperparameter choice in CNNs and Transfer Learning 36
3.18.1 Learning rate 36
3.18.2 Momentum 36
3.18.3 Batch Size 37
3.18.4 Adam Optimizer 37
3.19 Model Validation in Machine Learning 37
3.20 Under fitting and over fitting in machine learning 38
3.21 Techniques to reduce overfitting 39
3.21.1 Data augmentation 39
3.21.2 Early stopping 39
3.21.3 Dropout 39
3.21.4 Global average pooling layer 40
3.22 Class imbalance 40
3.23 Machine Learning in mobile health applications 41
3.24 Previous work using CNNs in the literature for skin cancer detection 41
4 RESEARCH METHODOLOGY 44
4.1 Chapter Overview 44
4.2 Research design 44
4.3 Data Collection 45
4.4 Data exploration and solving class imbalance 48
4.5 Selecting a suitable machine learning model 51
4.5.1 Selecting a Deep Learning Platform: Keras and TensorFlow 51
4.6 Training the network: determining the optimal method of TL and hyperparameters 52
4.6.1 Transfer Learning 53
4.6.2 Fine Tuning 54
4.6.3 Retraining the entire architecture 54
4.6.4 Finding the optimal hyperparameters 55
4.6.5 Final model training: Training the entire dataset 58
4.7 Evaluating the final models on the test set. 60
4.8 Building the model into a Flask Application 61
4.9 Ethics of Methodology 62
5 RESULTS AND FINDINGS 63
5.1 Chapter Overview 63

5.2 Results of Transfer Learning and Fine Tuning on the subset of data 63
5.3 Results of Altering Hyperparameters 66
5.4 Results of Final Model Training on Entire dataset 69
5.5 Overview of class accuracy in MobileNet 71
6 EVALUATION 72
6.1 Chapter Overview 72
6.2 Research Questions 72
7 DISCUSSION 74
7.1 Chapter Overview 74
7.2 The impact of the study on Dermatology 74
7.3 A comparison of this model with previous skin lesion classification models 75
7.4 Pretrained CNNs and Transfer Learning 76
7.5 The proposed Flask application 77
8 CONCLUSIONS 78
8.1 Chapter overview 78
8.2 Concluding remarks 78
8.3 Strengths and Limitations of the study 79
8.3.1 Strengths 79
8.3.2 Limitations 80
8.4 Further work and recommendations 81
REFERENCES 82
APPENDICES i

LIST OF ILLUSTRATIONS

Figure 1: Global melanoma incidence rates per 100,000 according to the Global Cancer Observatory (2018). 9
Figure 2: *Global NMSC incidence rates per 100,000 according to the Global Cancer Observatory (2018).* 10
Figure 3: Neurotransmitter release between neurons across the synaptic cleft.. 25
Figure 4: A single artificial neuron. 25
Figure 5:Artificial Neural Network architecture 27
Figure 6: Stages and layers of a CNN model. 29
Figure 7:A Convolutional layer. 30
Figure 8 and 9: Max Pooling Operation. 31
Figure 10: VGG Net. 34
Figure 11:Depthwise Separable Convolutions. 35
Figure 12: Lesion classes in HAM10000. 47
Figure 13:Class imbalance in HAM10000. 49
Figure 14: Data augmentations 49
Figure 15: The created dataframe 50
Figure 16: Results of data augmentation. 50
Figure 17: An overview of the three methods of transfer learning explored. 53
Figure 18: Transfer Learning snapshot. 54
Figure 19: Retraining the architecture training snapshot. 55
Figure 20:Optimal learning rate. 56
Figure 21: Adam Optimizer. 56
Figure 22: Nesterov Momentum. 57
Figure 23: Increasing batch size. *57*
Figure 24: Adding a dropout layer. 58
Figure 25: Model Performance of MobileNet. 59
Figure 26: Model Performance of Xception. 60
Figure 27: Flask App Home Page. 61
Figure 28: Flask App Insert File HTML Page 61
Figure 29: Accuracy of Transfer Learning on subset. 64
Figure 30: Accuracy of Fine Tuning on subset. 64
Figure 31:Accuracy of Retraining the model on the subset. 65
Table 1: Results of TL, Fine Tuning and Retraining the model. 66
Table 2: Results of different hyperparameter settings. 67
Figure 32: Addition of momentum. 68
Figure 33: Addition of Nesterov Momentum 68
Table 3: results of final model training on the entire dataset. 70
Table 4: Accuracy score for each class in HAM10000 using the MobileNet model on the test set. 71

LIST OF ABBREVIATIONS

AANN: Auto Associative Neural Network

Adam: Adaptive Moment Estimation

AF: Activation Function

AI: Artificial Intelligence

AK: Actinic Keratosis

AN: Artificial Neuron

ANN: Artificial Neural Network

BCC: Basal Cell Carcinoma

BGD: Batch Gradient Descent

BK: Benign Keratosis

BNN: Back Propagation Neural Network

CL: Convolutional Layer

CNN: Convolutional Neural Network

DF: Dermatofibroma

DSC: Depthwise Separable Convolution

ELM: Epiluminescence Light Microscopy

FCL: Fully Connected Layer

GPU: Graphics Processing Unit

KNN: K-Nearest Neighbour

MBGD: Mini-Batch Gradient Descent

ML: Machine Learning

MLP: Multi-Layer Perceptron

MN: Melanocytic Nevi

MSC: Melanoma Skin Cancer

NAG: Nesterov Accelerated Gradient

NMSC: Non-Melanoma Skin Cancer

NN: Neural Network

PL: Pooling Layer

ReLu: Rectified Linear Unit

SCC: Squamous Cell Carcinoma

SF: Softmax Function

SGD: Stochastic Gradient Descent

SL: Supervised Learning

SVM: Support Vector Machines

TL: Transfer Learning

USL: Unsupervised Learning

UV: Ultra Violet

VASC: Vascular Lesions

WHO: World Health Organisation

1 INTRODUCTION

Across the globe, skin cancer prevalence is rising at an unprecedented rate. The World Health Organisation (WHO) estimates that approximately 2-3 million cases of non-melanoma skin cancer (NMSC) and 130,000 incidences of malignant melanoma (MSC) occur each year. It is the most common form of cancer in Ireland (Irish Skin Foundation, 2019). Ultra-Violet Radiation (UVR) from the sun and global warming are the most significant contributors to these alarming statistics (Narayanan et al., 2010). These rising rates bring a multitude of problems including NMSC surgery, subsequent scarring and disfigurement, a burden on the healthcare system and high cost to the state. MSC can be fatal if not caught early as it may spread to surrounding areas (Sheha et al., 2012). As seen in figure 1 and 2, Australia and New Zealand were most significantly affected by both forms of the disease in 2018.

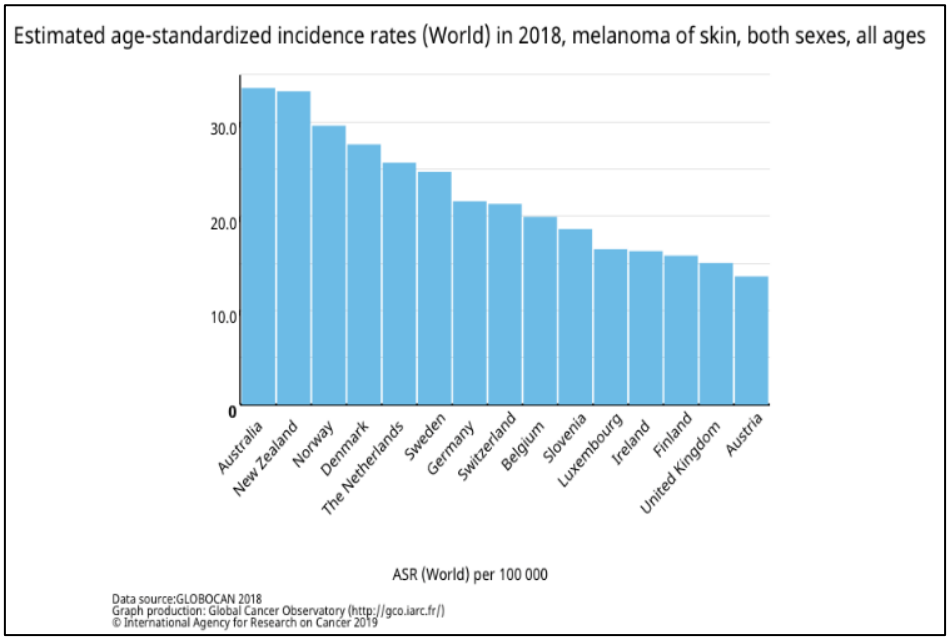


Figure 1: This chart shows the global melanoma incidence rates per 100,000 according to the Global Cancer Observatory (2018).

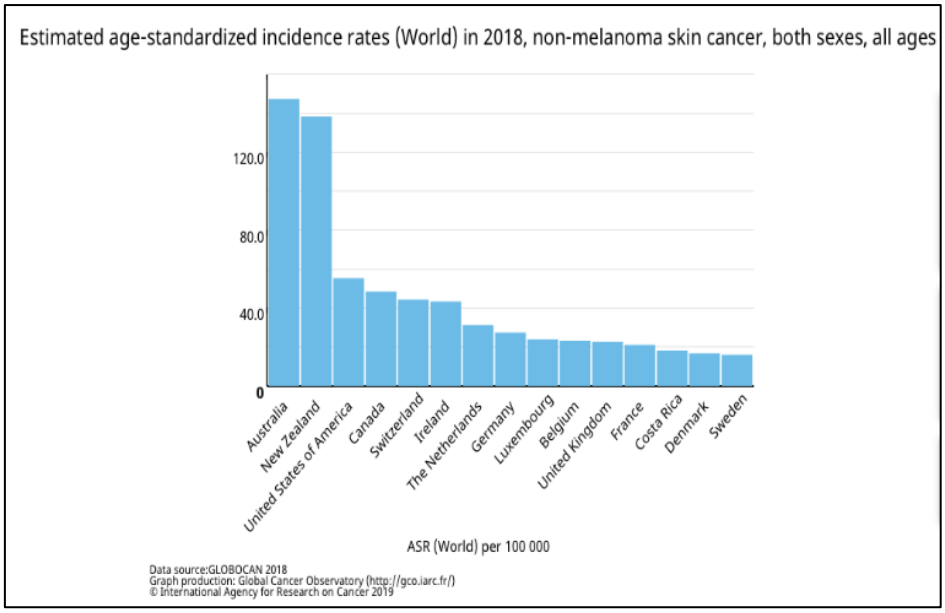


Figure 2: This chart shows the global NMSC incidence rates per 100,000 according to the Global Cancer Observatory (2018).

Skin cancer and other lesions are diagnosed using a technique called dermoscopy, which magnifies a skin lesion image making diagnostic features more distinct (Harangi, 2018). The lack of trained staff and resources in dermoscopy is a major limitation in this area, with diagnostic accuracy reduced if adopted by an inexperienced professional (Celebi et al., 2007). To limit the subjectivity of clinical diagnosis and reduce the burden on the healthcare system, research into using machine learning models to diagnose skin cancer is expanding. Integrating such systems into healthcare could act as a second opinion or pre-screening tool. Automating and speeding up diagnosis would be especially important to detect MSC early, preventing the cancer from spreading and decreasing death rates. Recent advancements in artificial intelligence (AI) and machine learning (ML) could make such automated systems in healthcare a reality. There has been a growing interest in recent years in the use of machine learning in healthcare diagnosis, for example, to diagnose blood tissue samples (Kieffer et al., 2017). This interest also applies to dermatology and dermoscopic image analysis. The most successful ML models for image processing are convolutional neural networks (CNNs). CNNs have achieved much success in recent years, most notably when used in classifying the ImageNet dataset. ImageNet is a dataset of over 15 million labelled images of approximately 22,000 categories gathered from the internet (Krizhevsky et al., 2012). Many of the CNNs developed for use in this dataset (termed pre-trained CNNs) are publicly available, such as Xception, and can be reused to classify a different dataset. This useful technique is called

transfer learning (TL) and has several different methods, depending on the dataset in question.

This study will progress through a number of chapters. In chapter 2, an official problem statement is declared, and a number of research questions are posed, around which the rest of the study will be based on. In chapter 3, an extensive literature review is carried out detailing the significance of skin cancer, ML technologies used in image processing, CNNs and the different methods of TL. In chapter 4, the research design is set out and a detailed account of the methods used are described. In chapter 5, the results of chapter 4 are presented and analysed. This is followed by chapter 6, an evaluation of the findings in relation to the research questions posed in chapter 2. In chapter 7, the findings are discussed in more detail and compared to findings in the literature. The final chapter, chapter 8, will offer the final conclusions, strengths and limitations of the study and recommendations for further work.

2 PROBLEM STATEMENT

As skin cancer incidence rates continue to rise worldwide, there is a compelling argument for the use of AI to aid the diagnosis of dermoscopy images. Currently in the literature, the research has a significant focus on models that can distinguish between MSC and NMSC. However, it is of equal importance to correctly diagnose a benign lesion to avoid unnecessary surgeries and biopsies. It is therefore important that multi-classification models are developed, which include other common skin lesions such as moles, vascular lesions and actinic keratosis. There are a limited number of studies which have attempted to develop a multi-classification model. Many of these studies, including the work of Harangi et al. (2018), only include three classes, which are not representative of the vast variety of skin lesions. These studies are also limited by unbalanced datasets, with certain classes having many more images than others (Kawahara et al., 2016). This can introduce bias to the model, as it is more likely to predict the majority class and thus the model yields an unrealistic accuracy score (Celebi et al., 2007). Further, in the past the number of images available for model training was limited (Esfahani et al., 2016; Shoieb et al., 2016; Kawahara et al., 2016). To build a model which can successfully generalise to unseen data, a large labelled dataset is required. Most studies do not make use of dermoscopic images, meaning the model cannot apply to clinical practice where dermoscopy is widely used.

This study will attempt to build an accurate multi-classification model for dermoscopy images which accounts for the limitations mentioned. The resulting model will be the first multi-classification model for dermatology that is trained on a large dataset of dermoscopy images, accounts for class imbalance and has a wide variety of skin lesion classes. The study will use pre-trained CNNs to develop such a model and investigate the different methods of transfer learning.

Research questions for this study will include:

1. Can an accurate multi-classification model be developed, which can distinguish between multiple skin lesions, with high accuracy?
2. Which method of transfer learning is optimal for this task?
3. Will larger CNN models result in higher accuracy?

It is hypothesised that an accurate multi-classification model can be developed for dermoscopy images. However, after class imbalance has been addressed, I predict a lower overall accuracy than in the literature. The pre-trained CNNs have been trained on the ImageNet dataset, including images such as objects, animals and people, which are very different to images of skin lesions. I hypothesise that due to this difference between the original data and data used in this study, retraining the entire model architecture from scratch is likely to yield the highest accuracy. I anticipate that larger pre-trained CNNs will yield a higher accuracy score as this is frequently suggested in the literature (Howard et al., 2017).

3 LITERATURE REVIEW

3.1 Chapter Overview

This chapter will explore skin cancer in depth including: the types of skin cancer, incidence rates, causes, diagnosis and treatment. It will investigate methods of image processing from past to present and the use of these techniques in dermatology. The chapter will pay attention to convolutional neural networks (CNNs) and any past attempts at applying these models to images of skin lesions. It will attempt to uncover any gaps in the literature, around which this study can build on.

3.2 Skin Cancer

Skin cancer is defined as the uncontrolled proliferation of abnormal skin cells (ICS, 2018). Skin cancer can be divided into two basic classes: melanoma skin cancer (MSC) or non-melanoma skin cancer (NMSC) (Kawahara et al., 2016). NMSC can be further divided into basal cell carcinoma (BCC) and squamous cell carcinoma (SCC) (Narayanan et al., 2010). Both forms of skin cancer have increased dramatically over the past number of decades. BCC occurs in the cells at the bottom of the epidermis (the outside layer of the skin). It is the most common type of skin cancer and most often will not spread to other areas of the body (Irish Skin Foundation, 2019). SCC occurs in the squamous cells, the cells closest to the surface of the skin. It is also unlikely for this form of cancer to spread. Malignant melanoma is a less common but potentially life-threatening form of skin cancer. It occurs in the melanocytes, cells that produce pigment (colour) on the skin (ICS, 2018). Skin cancer prevalence is rising worldwide. In America, one in five people will be diagnosed with skin cancer in their lifetime, and one third of cancers diagnosed are skin cancers (World Health Organization, 2019). Skin cancer is the most common form of cancer in Ireland. In 2017, almost 11,000 NMSC's and over 1000 MSC were diagnosed (Irish Skin Foundation, 2019). Melanoma is attributed to 75% of skin cancer deaths (Ali et al., 2017). Although NMSC is rarely life threatening, it is the most prevalent cancer in light skinned populations, placing a serious burden on an individual's quality of life and health care amenities (Kawahara et al., 2016). NMSC and related surgery can lead to disfigurement causing both physical and psychological issues in patients (Narayanan et al., 2010). It is also costly on the state: in the USA the average cost per year for NMSC treatment is more than 500 million (Soehnge et al.,

1997). It is predicted that skin cancer incidence rates will continue to rise, increasing the burden on the healthcare system (Soehnge et al., 1997). The cause of all skin cancers is multifactorial but UVR exposure is a significant contributor (Narayanan et al., 2010).

3.3 UVR and skin cancer

The skin is composed of multiple layers, each with unique functional and optical features (Maglogiannis et al., 2009). The epidermis layer consists of predominantly connective tissue and melanocytes (Maglogiannis et al., 2009). Melanocytes produce melanin, a pigment that absorbs blue light in the visible and Ultra-Violet (UV) spectrum. It acts as a barrier to the deep layers of the skin, adding protection from the damaging effects of UVR. Light not absorbed by melanin passes into the dermis layer, which is composed of collagen fibres, receptors, nerve ends and blood vessels (Maglogiannis et al., 2009).

UVR exposure is thought to directly damage cells and alter the immune system (Narayanan et al., 2010). Most significantly, UVR can induce genetic mutations in the DNA of these cells. The most important genetic mutations induced by UVR are those to the p53 genes (Narayanan et al., 2010). P53 genes are a form of tumour-suppressor genes that function to inhibit cell division in damaged or abnormal cells through apoptosis, also known as programmed cell death (Campbell, 2014). Apoptosis is induced by the activation of enzymes that break down components of a damaged cell, preventing its division (Campbell, 2014). A mutation induced by UVR to these genes decreases the normal functioning of these tumour suppressor proteins, which promotes abnormal cell proliferation. This proliferation leads to the onset of cancer. Excessive exposure to UVR from the sun and/or artificial sources such as tanning beds can damage melanocytes, resulting in their abnormal growth and an excessive production of melanin (Sheha et al., 2012; Harangi, 2018). Atypical proliferation of melanocytes can develop into a malignant tumour known as MSC (Harangi, 2018). When the abnormal melanocytes and melanin are only present in the epidermis, melanoma is said to be localised or “in situ” (Maglogiannis et al., 2009). While localised, MSC is not life threatening. When abnormal melanocytes and melanin deposits migrate to the dermis, the nature of skin colouration is modified, and melanoma is classed as malignant.

Further, early mutations of p53 have been found in the epidermis which otherwise appears healthy (Boukamp, 2005). It has been suggested that these patches of mutated p53 are precancerous lesions. While BCC is thought to develop independently, SCC develops over multiple stages and often evolves from these precancerous lesions. Actinic Keratosis (AK) is

a well-known precancerous lesion, induced by exposure to UVR, which will often develop into SCC (Boukamp, 2005). Similarly, Bowens disease is a preinvasive stage of SCC.

3.4 Diagnosis and treatment of skin cancer

Although death rates associated with MSC are significant, if detected early it is curable and treatment will be limited to a simple excision (Esfahani et al., 2016; Celebi et al., 2007). If MSC diagnosis is delayed, it can spread to other parts of the body and becomes difficult to treat (Sheha et al., 2012). Further, the 5-year survival rate decreases from 98% for localised melanoma to 15% if metastasis (cancer spreading to other tissues) occurs (Pomponiu et al., 2016). Differentiating between MSC, NMSC and other benign lesions is vital in determining course of treatment (Kawahara et al., 2016).

Dermoscopy, also known as Epiluminescence Light Microscopy (ELM), was developed in 1987 (Lau et al., 2009). It is a non-intrusive imaging method to diagnose skin lesions using a dermoscope. This technique involves planting an oil immersion between the skin and the optics. The lighting magnifies the skin image, improving the vision of pigmentation and colour shades of the lesion not seen by the human eye (Lau et al., 2009). Increased magnification and the removal of surface reflection on the skin provides extra visual information and at deeper levels of the skin (Harangi, 2018). Dermoscopy heightens sensitivity for skin cancer diagnosis and decreases the chances of unnecessary biopsy of a benign lesion (Wolner et al., 2017). Although there has been a significant rise in the use of dermoscopy, the lack of training and resources continues to pose a limitation (Wolner et al., 2017). Diagnostic accuracy in dermoscopy is significantly reduced when adopted by an inexperienced professional/dermatologist (Celebi et al., 2007). The examination relies on a dermatologist's visual perception, and this can lead to subjectivity and insufficient accuracy.

For clinicians and dermatologists, diagnostic methods such as the ABCD (Asymmetry, Border irregularity, Colour patterns and Diameter) rule are generally used in diagnosis (Esfahani et al., 2016). The letter E was added to the ABCD rule in 2004 and represents the word 'Evolving' (Ali et al., 2017). Limitations of this rule include the inability to discriminate between melanoma and atypical moles/nevi and to identify early stage melanoma (Pomponiu et al., 2016). The seven-point checklist is also frequently used in diagnosis (Esfahani et al., 2016). This includes three major criteria and four minor criteria for the diagnosis of skin lesions (Pomponiu et al., 2016). Major criteria include irregular pigment

network, blue-whitish veil, and abnormal vascular pattern. Minor criteria include atypical streaks, pigmentations, dots and regression features (Pomponiu et al., 2016).

3.5 Computer-aided diagnostic systems for Skin Cancer

To limit the subjectivity of clinical diagnosis, computerised dermoscopy image analysis systems are being extensively researched (Celebi et al., 2007). These systems are intended to act as a second diagnostic opinion, a pre-screening tool and an aid to inexperienced technicians. These systems cannot offer an absolute diagnosis but show potential in helping to form biopsy related decisions (Celebi et al., 2007). Differentiating between malignant MSC and other, similar, benign moles can prove challenging even for the most experienced dermatologist (Esfahani et al., 2016). Computerised algorithms could help to aid healthcare professionals in this area. The need for computerised diagnostic systems for clinical and dermoscopy images is increasing rapidly due to the rising rates of skin cancer, subjectivity of human visual diagnosis and the time and costs associated (Esfahani et al., 2016). As early melanoma detection is critical to prevent metastasis and subsequent mortality, automating and speeding up the diagnosis is paramount. With advancements in machine learning and artificial intelligence, this task could become a reality in healthcare in the future.

3.6 Before AI and Machine Learning: Early Pattern Recognition Techniques

Modern image processing techniques using large computational power and machine learning are all largely based on statistical pattern recognition techniques which have been around for decades (Webb et al., 2002). Pattern recognition developed considerably in the 1960s (Webb et al., 2002). Researchers began investigating methods for problems such as character recognition and medical applications. Pattern classifiers are used to either act as a model which can explain different patterns of different classes or to predict the class of an unknown pattern (Webbs, 2002). It involves the study of how computer systems can scan the data presented and learn to detect specific patterns in order to make an informed decision about its category (Jain et al., 2000). There are many basic approaches to pattern recognition including template matching, statistical classification and syntactic/structural matching (Jain et al., 2000).

3.6.1 Template Matching

One of the earliest methods of pattern recognition is template matching. The term matching corresponds to the process of detecting similarities between two points, curves or shapes. A template of the pattern to be detected is available to match new patterns against while considering translations, rotations and scale alterations. As this method is very rigid it is not a highly accurate or popular method (Jain et al., 2000).

3.6.2 Statistical classification

The statistical approach involves viewing every pattern in terms of *n* features and a point in *n* dimensional space. The aim of the statistical approach is to determine the features that can effectively separate classes. It takes a training pattern from each class and determines the boundaries in the *n* dimensional space that can separate patterns from different categories (Jain et al., 2000). The theoretical statistical decision approach involves computing boundaries by means of probability scores of patterns belonging to a specific class. These probability scores must be learned. The discriminant analysis-based approach involves a decision boundary which must be specified, for example, a straight line is based on the training points and is the line that minimises error in the classification task (Jain et al., 2000). In the past, statistical attempts at pattern recognition in images involves mainly facial recognition tasks (Kaufman, 1976). Attempts include defining a feature vector where facial characteristics are represented numerically. For example, features like mouth width were measured with scores of 1 to 5. These early attempts, although time consuming, were quite successful, scoring 70% accuracy on 255 faces (Kauffman, 1976). Other statistical approaches included using Euclidean distances for different points on the face as features and using these features to detect an individual (Kauffman, 1976). In these methods, the researchers manually measured these features in order to create the feature vector. Many of these early facial recognition approaches failed when objects such as beards and glasses were added (Kauffman, 1976).

3.6.3 Syntactic matching

The syntactic approach is for more complex pattern recognition tasks. It is based on the idea that one pattern is composed of small sub patterns which are made up of even smaller sub

patterns (Jain et al., 2000). The simplest sub patterns are called the primitives and the original complex pattern is described in terms of connections between the primitives (Jain et al., 2000). Syntactic pattern recognition is an analogy of the structure of a language. Patterns are represented as sentences of a language, with primitives representing the words. Sentences are composed of several grammar rules.

In the same way, syntactic pattern recognition involves describing complex patterns by primitives and grammatical rules. These grammatical rules will be determined by the training samples. This approach is useful in situations where patterns have a distinct structure which can easily be defined by rules, for example, textured images. The approach also has challenges, for example, determining connections from training data in cases where patterns have background noise (Jain et al., 2000). There have been many advances in all pattern recognition approaches in recent years due to a large amount of computational power and the availability of large datasets. These advancements have led to more efficient pattern recognition systems, machine learning and artificial intelligence.

3.7 Artificial Intelligence

Artificial Intelligence (AI) is a category of computer science. The ambition of developing a non-biological intelligence has been around for hundreds to thousands of years (Spencer, 2006). Negnevisky (2005) defines intelligence as "the ability to learn and understand, to solve problems and make decisions". The goal of AI is to develop machines which can perform tasks that would demand intelligence if completed by humans (Negnevisky, 2005). Various human activities such as understanding language, classifying objects, reasoning, writing computer programs and driving are considered acts that require some degree of intelligence (Nilsson, 2014). Nowadays, many computer systems can perform such clever tasks. There are computer systems capable of recognising and understanding human speech and language, systems diagnosing disease and even systems that can automate parts of driving (Nilsson, 2014). AI can be applied to real life problems in many disciplines such as science, engineering and medicine (Barr et al., 2014).

3.8 Machine Learning

Machine learning is defined as "the study of tools and methods for identifying patterns in data" (Wiens et al., 2017). The identified patterns can help increase our understanding of the data or make predictions about the data. ML incorporates a variety of fields such as statistics and computer science. To solve problems using computers, an algorithm is required (Alpaydin, 2009). An algorithm is a series of rules that transform an input into an output. For a task, more than one algorithm may be suitable, the chosen algorithm should be the most efficient in terms of time and memory (Alpaydin, 2009). ML is commonly categorised as a branch of artificial intelligence (VanderPlas, 2016). According to VanderPlas (2016), it is more useful to consider ML as a method of building mathematical models and algorithms of data to better understand such data. ML uses statistical theory to develop a mathematical model (Alpaydin, 2009). The notion of "learning" is relevant when models are given tuneable parameters. These internal adjustable parameters are termed the weights (LeCun, 2015). These weights are real numbers which establish the input-output function of the model. There can be millions of adjustable weights (LeCun, 2015). The weights can be changed to the specific data and the model is thought to be learning from the data (VanderPlas, 2016). Fitting the models to previously encountered data allows them to be used to predict and comprehend features of new, unseen data. ML features in many areas of modern society. This includes web searches, recommendations, social networking and even smartphones and cameras (LeCun et al., 2015). ML models are capable of image processing, speech recognition and natural language processing. At a basic level, ML is classified into two groups: Supervised learning methods and unsupervised learning methods (VanderPlas, 2016).

3.9 Supervised and Unsupervised learning

Supervised learning (SL) involves modelling the relationship between features/aspects of the data and a label or category associated with the data (VanderPlas, 2016). This model can then be used to determine the labels of new, unseen data. SL can be divided into classification and regression tasks. In classification tasks labels/variables are categorical, for example, healthy state and disease state. In regression tasks, variables are numerical, for example, height and weight values. SL is the most common form of machine learning (LeCun et al., 2015).

Unsupervised learning (USL) involves developing a model from features of the data without specifying the labels of the data (VanderPlas, 2016). There is no supervisor providing the correct output values as in SL (Alpaydin, 2009). Only input values are available to the model. USL attempts to find patterns in the input. Some patterns in the input space will occur more often than others. In statistics, this is referred to as density estimation (Alpaydin, 2009). USL includes clustering and dimensionality reduction. Clustering algorithms divide the data into distinct groups. Dimensionality reduction algorithms reduce the number of features in the feature vector to represent the data in a more compact manner (VanderPlas, 2016).

3.10 Image Processing for Melanoma detection

Attempts to automate the diagnosis of melanoma and other skin lesions using ML techniques is not a recent concept, with literature dating back as early as 1987 (Celebi et al., 2007). In the past, ML methods for classifying skin lesions, and image processing in general, involved four main steps. The first step is pre-processing to remove noise, improving the shape and quality of the image (Lau et al., 2009). The second step is feature extraction, which is followed by the third step, feature selection. The final step involves the selected features being fed into a machine learning classifier or algorithm (Lau et al., 2009).

3.10.1 Image pre-processing

Images of the skin often have illumination and noise effects (Esfahani et al., 2016). Digital images from a regular camera will contain irrelevant features such as hair and air bubbles (Shoieb et al., 2016). This can cause inaccuracy in classification, incorrect prediction results (Shoieb et al., 2016) and inaccuracy in segmentation (Lau et al., 2009). Various pre-processing techniques can reduce these effects (Esfahani et al., 2016). Steps like illumination correction (Esfahani et al., 2016), median filtering (Shoeib et al., 2016), histogram equalization algorithm, wavelet transformation (Lau et al., 2009) and Gaussian filter (Esfahani et al., 2016) are frequently used in the literature. These steps smooth the area around the lesion, reducing the effect of normal skin on classification (Esfahani et al., 2016), removing air bubbles and hairs (Shoeib et al., 2016), enhancing image contrast and defining edges for border detection (Lau et al., 2009). According to Celebi et al. (2007), it is important that borders are detected as their structure, such as asymmetry and irregularity, gives an indication of disease state. Further, it is thought that an accurate border detection algorithm

will also improve the accuracy of feature extraction for colour and texture (Celebi et al., 2007).

Border detection is often followed by segmentation to remove background skin and isolate the lesion. Lau et al. (2009) believe this is a necessary step as the distinctive features of melanoma are situated within the border and many diagnostic features (for example, asymmetry), are defined by the border shape. Techniques for segmentation in the literature include threshold values from RGB colour band and statistical region merging (Lau et al., 2009), and Ant Colony Optimization (Dalila et al., 2017).

3.10.2 Feature extraction

The term "features" in machine learning is similar to the term "variables" in statistics. They are values describing certain aspects of the data which can be used to predict or explain another value/variable (Sarie et al., 1994). Feature extraction is a core step in image processing (Shoeib et al., 2016). The process transforms the input data to low dimensional vectors that can be easily compared or matched. Feature extraction is usually specific to the classification task at hand and can involve high levels of prior knowledge from the feature engineer, as they are typically hand designed (LeCun et al., 1998). Extracting features with the ability to accurately define a class is a difficult task (Esfahani et al., 2016). Too many features can lead to incoherent features in the network. Too few features and significant descriptors could be lost (Esfahani et al., 2016). When classifying skin lesions, features such as shape, colour and texture are often used. Shape features include area, aspect ratio, asymmetry and compactness (Celebi et al., 2007). Colour features are based on calculations on colour spaces such as mean, standard deviation and histogram distance. Texture feature statistics include energy, entropy and contrast.

3.10.3 Feature selection

Following feature extraction, feature selection must then be used to reduce the dimensionality of the feature space (Celebi et al., 2007). This means reducing the number of features to use during training and prediction. Feature selection algorithms are divided into two categories: filter methods and wrapper methods. Filter methods reduce irrelevant features prior to the classification stage. Wrappers evaluate feature performance based on the results of predetermined learning algorithms. Wrappers are expensive but are more suited to

classification tasks when the classifier is pre-established (Celebi et al., 2007). Filter methods are faster, allowing multiple methods to be compared.

3.10.4 Classification

Selected features are fed into a classifier. Common ML classifiers in the literature for detecting skin cancer/lesions include Support Vector Machines (SVM) (Celebi et al. 2007, Yuan et al., 2006), K-Nearest Neighbour (KNN) (Dalila et al., 2017), Back Propagation Neural Network (BNN) (Lau et al., 2009), Auto-associative neural network (AANN) (Dalila et al., 2017, Lau et al., 2009), and Multilayer perceptron classifier (MLP) (Sheha et al., 2012).

3.11 Support Vector Machines (SVM)

SVMs are an efficient and flexible supervised learning algorithm for classification (VanderPlas, 2016). They are kernel-based algorithms, built on statistical learning theory (Celebi et al., 2007). This is an example of discriminative classification whereby instead of modelling each class, a line is identified to separate the classes from one another (VanderPlas, 2016). Often, there is more than one possible line that could separate the training classes. The classification prediction of a new data point will depend on the choice of line. To overcome this variability, SVMs attempt to maximise the margin. This involves drawing a margin around each line, up to the nearest point. SVMS choose the line that maximises the margin and thus optimizes the model (VanderPlas, 2016). A few training points will touch the margin, they are the crucial elements of the fit of the line and are called the support vectors.

3.12 K-Nearest Neighbour (KNN)

KNN is an instance-based classification model whereby each new point is compared with existing points based on a distance calculation, and the nearest data point is used to classify the new point (Witten, 2016). Typically, the distance metric is the Euclidean distance between training data points and the new data point (Peterson, 2009). This is called nearest neighbour classifier. Often, more than one neighbour is used (the number represented as ‘k’),

and the new data prediction class is based on the majority class of nearest neighbours (Witten, 2016). ‘K’ is usually an odd number, so a tie is avoided (Peterson, 2009).

3.13 Artificial Neural Networks

Neural networks (NN) have been considered the gold standard ML approach for decades. Their popularity in recent years can be accredited to the vast availability of data, larger network architectures, faster training times and computing power (Witten, 2016). This computing power is largely due to the development of Graphics Processing Units (GPUs). GPUs were originally motivated by the demand of graphics applications for computational power (Keckier et al., 2011). They have since been developed into powerful processers used for a multitude of tasks including machine learning (Owens et al., 2008). The significant feature of GPU systems is parallelism, meaning many calculations can be computed simultaneously (Owens et al., 2008). GPUs have improved the efficiency of many ML models such as NN. Artificial Neural Networks (ANNs) are a ML algorithm based on biological neural networks of the human nervous system (Basheer et al., 2000).

3.13.1 Biological Neural Networks

A single neuron is made up of three functional parts: dendrite, axon and cell body. The cell body contains the nucleus of the neuron, containing DNA and plasma to hold other cell components (Basheer et al., 2000). Dendrites detect signals from other neurons and relay them to the cell body. The axon carries signals from the cell body to the synapse where the signal is passed to dendrites of neighbouring neurons. The signal or impulse travelling from dendrites to axons is an electrical signal. When the electrical signal arrives at the synapse, it causes a neurotransmitter, a chemical signal, to be released from the synaptic vesicles (Basheer et al., 2000). The proportion of neurotransmitter released is proportional to the strength of the electrical signal. The neurotransmitter is released across the synaptic gap and received by the post synaptic membrane of neighbouring neurons and then enters the dendrite (Basheer et al., 2000). If a threshold is reached on the neighbouring neuron, a new electrical signal will be generated. As a neuron has multiple dendrites and synapses it can receive multiple signals simultaneously (Basheer et al., 2000). The signals will either induce or stop

the firing of a neuron. The process of neurotransmitter release is illustrated below in figure 3. This basic version of message transfer in neurons is the mechanism inspiring ANNs.

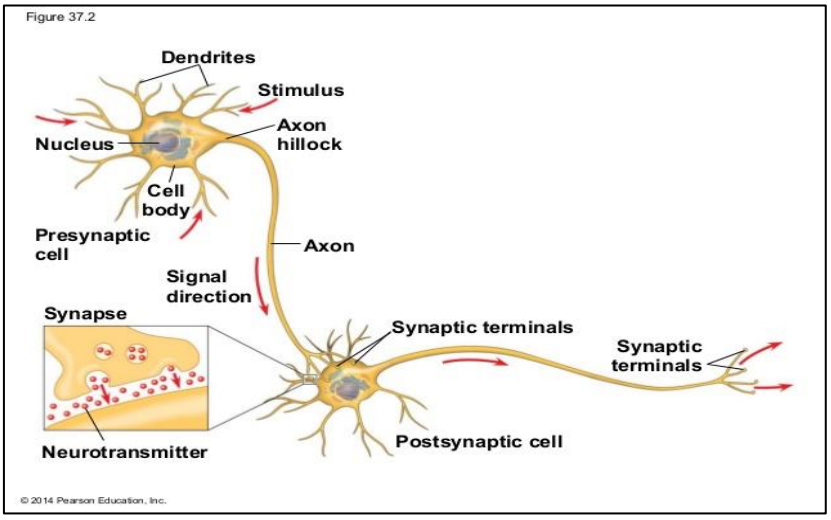


Figure 3: This image shows the process of neurotransmitter release between neurons across the synaptic cleft (Campbell, 2014). This process is the basic mechanism behind ANN.

3.13.2 Neural Networks in Machine Learning.

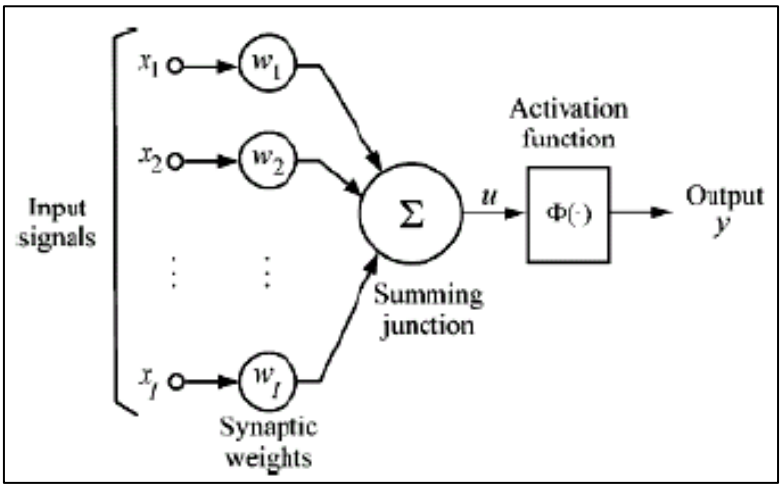


Figure 4: This diagram shows the architecture of a single artificial neuron (Boukadida et al., 2011).

A single artificial neuron (AN) is based on the biological neuron. In ML this AN can be called a neuron, node or unit (Tampere University, 2019). This singular AN is often termed the 'Perceptron' (Basheer et al., 2000). When many individual neurons are connected, it is called a neural network algorithm or a multilayer perceptron (Basheer et al. 2000; Tampere

University, 2019). The connections between these units are an analogy of biological connections between axons and dendrites (Basheer et al., 2000). A single neuron, shown in figure 4, receives an input from another neuron in the network or a data input (Tampere University, 2019). The AN multiplies the received input by a weight (adjustable parameter), denoted by 'w' in figure 4. The multiplied values are added together at the summing junction. This sum is often termed the net input (Sarie et al., 1994). The net input is passed to an activation function (AF) which produces a single output (Tampere University, 2019). This output is either passed to the next unit or used for classification.

3.13.3 Activation functions in neural networks

An AF creates boundaries for neuron outputs (Tampere University, 2019). There are many different activation functions that can be used. Early ANNs typically used a threshold activation function. This function uses a threshold value, for example 0.5. In the case of binary classification, any value over 0.5 will have an output of 1 and less than 0.5 will have an output of 0. In recent years, the threshold function has been swapped for either the sigmoid function or the rectified linear unit (ReLu) function, with the ReLu function considered superior (Tampere University, 2019). The sigmoid function outputs values between 0 and 1, ReLu outputs values between -1 and +1. The AF found in the output layer is usually the softmax function. It outputs the probability that the input falls into each possible class, and all probabilities must add to 1 (Tampere University, 2019).

3.13.4 Hidden layers

Units/neurons not in the input or output layers are called hidden units (LeCun et al., 2015). Hidden units only receive inputs from other nodes or units (Tampere University, 2019). They also only pass an output to another node. They are not connected to the original input or output. Deep learning involves networks of many hidden layers. In the past, this large network style was not used due to lack of training data and computational power (Tampere University,2019). Instead, two hidden layers were considered sufficient to learn patterns. However, deep learning can learn more complex high-level features that smaller networks fail to accomplish.

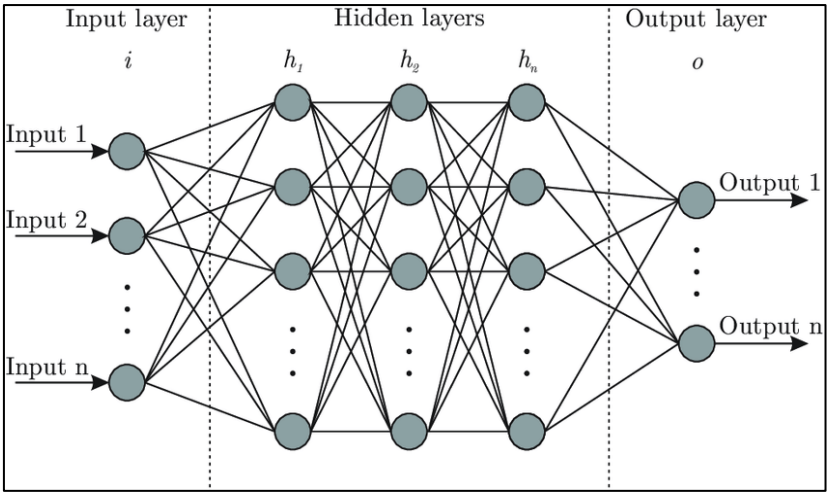


Figure 5: This image shows the connections between input nodes, hidden nodes and output nodes in an ANN (Bre et al., 2017).

3.13.5 Back propagation and stochastic gradient descent in neural networks

When the final output unit outputs probabilities for each possible category, the goal is that the true category would have the highest score (LeCun et al., 2015). This will not occur before training a model. Back propagation is the most common method for training ANNs. It is a group of methods that calculate the difference between the output scores and true values and communicates this difference/loss back through the network, adjusting the weights between nodes to reduce this error (LeCun et al., 2015). The model calculates a gradient vector for every weight/parameter. This defines the level the error would change if the value of the weight was altered by a small number. This involves calculating the error with respect to internal parameters of the layers (LeCun et al., 2015). The weight vector is altered in the opposite direction of the gradient vector (LeCun et al., 2015).

Most NN models using back propagation also use an optimising method called Gradient Descent. There are three versions of the gradient descent algorithm which are based on the level of data required to compute the gradient: Batch gradient descent (BGD), Stochastic Gradient Descent (SGD) and Mini Batch Gradient Descent (MBGD) (Ruder et al., 2016). Choosing the version to use is usually a trade-off between the accuracy of the weight adjustments and the time taken for each adjustment. BGD calculates the gradient of the cost

function with respect to the weights for the entire training epoch i.e. the gradients for the entire training set must be calculated for one update (Ruder et al., 2016). This version is therefore slow. SGD updates the parameters after every training example. BGD performs many redundant calculations as it calculates gradients for many similar samples before weights are updated. As SGD performs one update per training sample it removes redundant calculations and is therefore faster. However, many updates introduce high variance which causes many fluctuations in the cost function. This can complicate convergence (Ruder et al., 2016). MBGD updates weights after every mini batch of *n* training samples (Ruder et al., 2016). This method can reduce fluctuations in convergence. This is usually the version of choice when training NN. The batch size varies depending on the dataset and computing power. MBGD is often termed SGD (Ruder et al., 2016).

3.14 A move towards Convolutional Neural Networks in Image Processing

According to LeCun (1998), more accurate recognition systems can be developed through automatic learning, rather than hand crafted engineering techniques. LeCun (1998) showed that complex feature extraction methods and pre-processing steps can be successfully replaced by machine learning models which utilise pixel images. Kawahara et al. (2016) agree it is better to avoid segmentations and complex pre-processing steps as these can introduce errors if carried out incorrectly and thus require human interference. Convolutional Neural Networks (CNNs) can overcome the need to manually extract features and other image pre-processing techniques as in other ML algorithms (Shoeib et al., 2016). CNNs are accurate classifiers but also act as feature extractors. They can extract distinctive features from images that are most relevant for the classification problem at hand (Shoeib et al., 2016).

3.15 Convolutional Neural Networks

CNNs first arose in the early 1990s and since then, they have shown excellent potential in classification tasks such as image classification and facial recognition (Zeiler et al., 2013). The ability of CNNs to process complex data in multiple arrays makes them extremely suitable for such tasks (LeCun et al., 2015). CNNs differ from conventional multi layered neural networks by their specific layers (Algarap, 2019). The improvement and increased popularity of CNNs is likely due to the accessibility of large labelled datasets and GPU implementations (Zeiler et al., 2013).

The four basic principles of CNNs are: local connections, shared weights, pooling and multiple layers (LeCun et al., 2015). The architecture of a standard CNN (as shown in figure 6) is defined by several stages. Two types of layers represent the first stage. These are convolutional (CL) and pooling layers (PL) (LeCun et al., 2015). CNNs are based on the knowledge that high-level features are found by creating low-level features (LeCun et al., 2015). The inspiration for the convolutional and pooling layers in CNNs is linked to simple and complex cells in neuroscience. The CNN architecture resembles the visual cortex ventral pathway in the brain (LeCun et al., 2015).

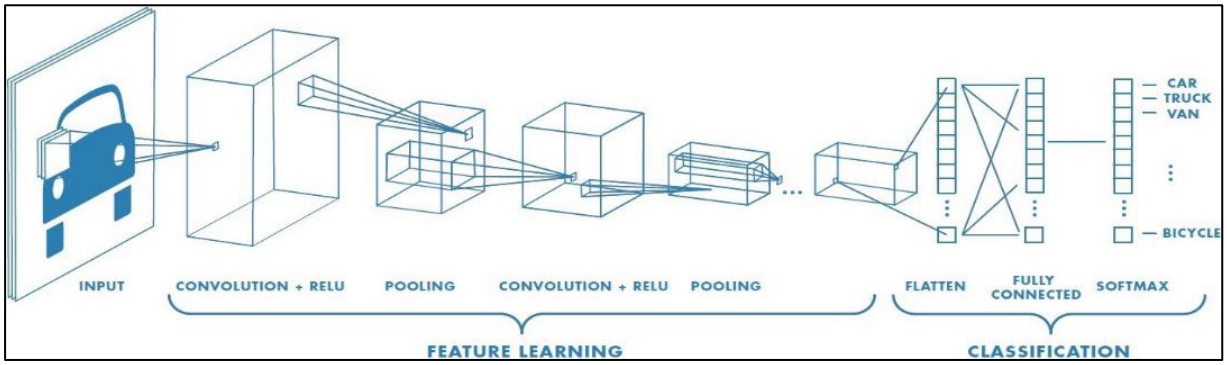


Figure 6: This image shows the different stages and layers of a CNN model (Medium: Towards Data Science, 2019).

3.15.1 Convolutional layer

The general purpose of a convolutional layer (CL) is feature extraction (Zhang, 2018). The units of a convolutional layer are arranged in feature maps (as shown in figure 7). In a feature map, each unit is connected to a place in the feature map of the preceding layer (LeCun et al., 2015). The parameters of a CL are defined by the size and number of feature maps, kernel

size, skipping factors and a connection table (Ciresan et al., 2011). Every layer has several maps of equal size. A kernel of specific size moves over the width and height of the input/ image and calculates the dot product of the local region and the weight learning parameters as shown in figure 7(Algarap, 2019). Skipping factors decide how many units the filter/kernel can skip in horizontal and vertical directions in between convolutions (Ciresan et al., 2011). The parameters of filters and skipping factors are defined so the output maps of the final CL are downsized to one unit per map or a fully connected layer merges the values of the last CL into a 1D feature vector (Ciresan et al., 2011). This connection happens as the result of a set of weights or the "filter bank". Every unit of a feature map share a common filter bank. Differing feature maps of one layer have differing filter banks. In mathematical terms, this filtering operation is a discrete convolution (LeCun et al., 2015).

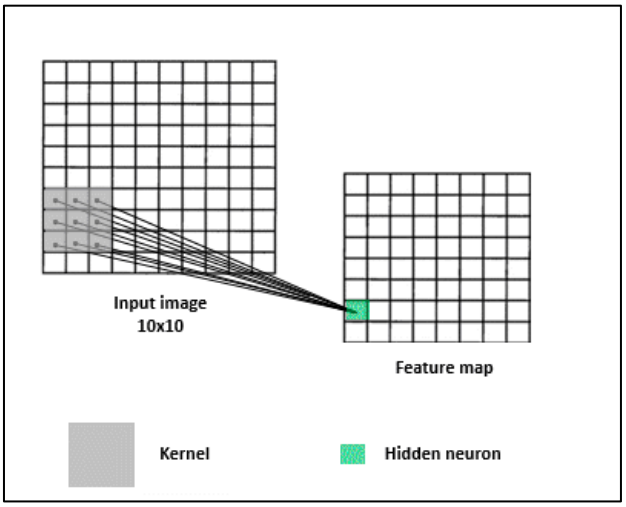


Figure 7: This shows the basic mechanism of a convolutional layer (KD Nuggets, 2017). The grey square represents the sliding kernel that calculates the dot product of the local region to create the feature map.

3.15.2 Activation Layer

The activation layer (AL) is characterised as a layer of neurons (Zhang, 2018). Each neuron in a layer uses an activation function (AF), as in typical ANNs. The AF of CNNs introduces the notion of non-linearities (Agarap, 2019). Without this AF, the model is only capable of learning linear mappings. An AL transforms feature maps to new feature maps and each feature is transformed into a new feature by the AF (Zhang, 2018). The most popular AF is the ReLu function, compared with other AFs such as tanh and sigmoid it quickly promotes

SGD convergence (Algarap, 2019). Typically, ReLu is used directly after a CL and directly preceding a pooling layer to form new non-linear feature maps (Zhang, 2018).

3.15.3 Pooling layer

CLs detect feature associations from the preceding layer, but pooling layers (PL) combine similar features into one (LeCun et al., 2015). This is a form of feature selection, reducing the number of parameters and computations in the network. This is called down sampling (Algarap, 2019). The PL uses the results of the CL filter to reduce the input size (Algarap, 2019). The PL can perform either a maximum calculation, max pooling (as shown in figure 8) or an average calculation, average pooling, on the feature map (Yigit et al., 2018). The PL uses a sliding kernel to apply such operations as shown in figure 9.

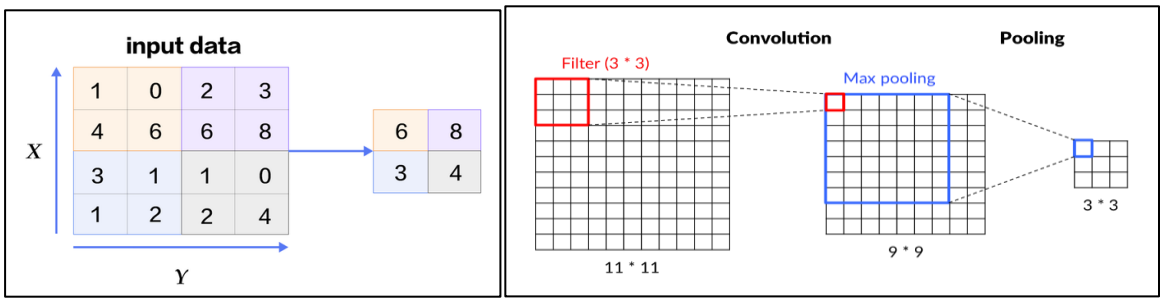


Figure 8 and 9: Figure 8 shows the results of a max pooling operation (Medium, 209). Figure 9 (yosuku.saito, 2017) illustrates the process of moving from a convolutional layer to a pooling layer and the resulting reduction in input size.

3.15.4 Fully connected layers and classification

Stages of convolution, pooling and non-linearity are stacked upon each other as shown above in figure 6. These are then followed by subsequent CL and one or many fully connected layers (FCL) (LeCun et al., 2015). Prior to this, all layers were involved in extracting and learning features. The final feature map is then flattened into a single vector which can be fed into several FCL, just like in artificial neural networks. These FCL are involved in classification, with the vector being passed through a number of hidden layers until the final output layer. In the final output layer, a softmax function (SF) is generally used (Yigit et al., 2018). The advantage of using a SF in a CNN is that training converges faster (Islam et al., 2018). Neural networks generally use SF activation function for classification. Although SF

is popular, it is not the only option (Tang, 2013). One popular alternative includes using either SVM or KNN for classification in the final layers. Using these ML algorithms in combination with CNNs involves first using CNN as a feature extractor and use these features as the input to the SVM/KNN classifier (Tang, 2013).

3.15.5 Fully convolutional neural networks

Novel CNN research involves interest in semantic segmentation and fully convolutional networks producing CNNs which can make a prediction at every pixel (Long et al., 2014). Semantic segmentation involves labelling each pixel with the class of the object to be detected in the image. Long et al. (2014) suggest that back-propagation and feed forward calculations are more efficient when calculated layer by layer on the whole image rather than independently section by section. Militari et al. (2016) agree that patch wise approaches in CNNs can often be inefficient as many computations can be redundant with unnecessarily long training times. There has been a rise in research using fully convolutional networks for these semantic segmentation tasks which outperform all other models for this task (Long et al., 2014). Fully convolutional layers remove fully connected layers so that the final layers are convolutional layers. A typical CNN will compute a non-linear function (Long et al., 2014). It will take inputs of a fixed size and produce a non-spatial output. The fully connected layers must have fixed dimensions and spatial coordinates are discarded (Long et al., 2014). A fully convolutional network differs from this as it calculates a non-linear filter, will take an input of any size and outputs its associated spatial dimensions (Long et al., 2014). More recent CNNs such as ResNet and GoogLeNet are designed to be fully convolutional and have shown high accuracies in multiple classification tasks (Dai et al., 2016).

3.16 Transfer Learning

Transfer learning (TL) has gained popularity in recent years (Yigit et al., 2018). This involves using a model which has already been trained on another dataset and applying the pretrained weights to another dataset to make a prediction. Deep learning demands large datasets and long training times. Pre-trained CNN models are competent architectures trained using large datasets. Using pre-trained CNNs is of great interest in the medical field as there is often a

lack of large labelled data to train a deep network (Kieffer et al., 2017). One way of utilising pre-trained CNNs is using the pre-trained filters as a feature extractor and modifying the classification layers to suit the dataset in question. These pre-trained filters have proven efficient for extracting features of a previously unseen dataset (Yigit et al., 2018). The only layer involved in training is therefore the final classification layer. This method is referred to as TL in the literature. The second option is to use pre-trained weights for some of the architecture and train other parts of the model from scratch. This method is known as fine-tuning. According to Chu et al. (2014), there is limited guidance available regarding which TL method to use and the number of layers to freeze. Both Chu (2014) and Yigit (2018) agree that method choice is influenced by the content and size of the dataset. Chu et al. (2014) advise that fine tuning layers beyond the final fully connected layers degrades model performance. However, the exception to this rule is when the source and target datasets are very different and there is sufficient labelled data available (Chu et al., 2014). In small datasets, if the contents of the dataset are similar to the original data, the feature extraction/frozen layers method would be sufficient (Yigit et al., 2018). If the dataset is small with very different content, fine tuning shows improvement (Chu et al., 2014). The final method involves only using the pre-trained CNN architecture, but retraining all the weights to suit the new dataset. If the dataset is large and data contents are different, Yigit et al. (2018) suggest this method. It should also be considered that fine tuning a model across all layers will take longer to train (Chu et al., 2014). Long et al. (2015) argues that pre-trained CNNs are only beneficial for learning general features. Long et al. (2015) further indicates that deep features change from general to specific at an abstract stage in the network. The transferability of features appears to drop in higher layers of a network as these layers rely on the specific dataset and task (Long et al., 2015). In general, pre-trained networks used for image classification problems are trained on the ImageNet dataset (Yigit et al., 2018). ImageNet is a dataset of over 15 million labelled images of approximately 22,000 categories gathered from the internet (Krizhevsky et al., 2012). In 2010, the now annual competition, ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) began. This competition uses 1000 of the 22,000 categories and approximately 1000 images per category. Many popular pre-trained CNNs were originally designed for this competition and trained on these image (Krizhevsky et al., 2012).

3.17 Popular pre-trained models for Transfer Learning

Early CNNs began with models, such as LeNet, involving stacked CL for feature extraction followed by max PL for reducing spatial dimensions of the feature map (Chollet, 2017). AlexNet emerged from this style of design with multiple CL between max PL, allowing the network to learn deep features (Chollet, 2017). AlexNet has 8 layers: 3 CL, 2 max PL and 3 fully connected layers. AlexNet has since been successfully used for many computer vision tasks such as video classification and object tracking (Szegedy et al., 2016). The success of AlexNet motivated researchers to design higher performing CNNS (Szegedy et al., 2016). The general trend in developing a CNN is to increase the depth (number of layers) and complexity of a NN to achieve higher accuracy (Howard et al., 2017). This trend of deeper architectures led to the VGG model with 19 layers in 2014 (as seen in figure 10). VGG has achieved high accuracy in many computational tasks. It has a simple architecture, however its many layers require high computational power and long training times (Szedegy et al.).

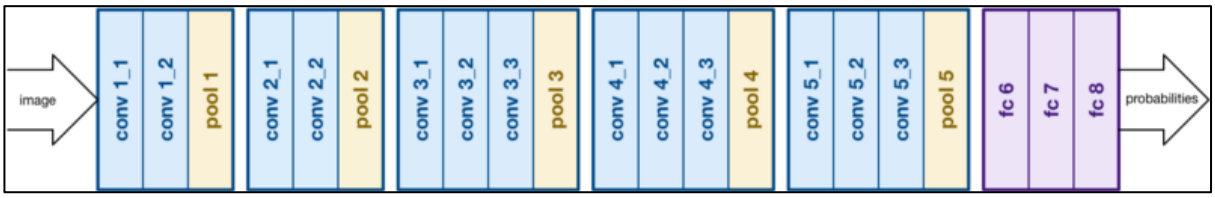


Figure 10: This image shows the architecture of the pre-trained CNN, VGG Net (Medium, 2017).

Howard et al. (2017) argue that these advances with deeper networks are not efficient when considering the larger size and slower speeds of these models. In many real-life applications, computational tasks may need to be processed rapidly and on a computationally lacking platform (Howard et al.,2017). Such tasks became the inspiration for a new design of CNNs: The Inception family of models (Szegedy et al., 2016). The Inception model is designed for high performance in cases of memory and computational constraints (Szegedy et al., 2016) Inception has become one of the highest performing models on the ImageNet dataset (Chollet ,2017). The Inception family of models are based on “Inception modules” which are similar in function to convolutional layers. These modules have shown to outperform convolutions, learning deeper features with fewer parameters (Chollet, 2017). The idea behind the Inception module is that the convolutional process is divided into several operations. Mapping cross channel correlations and spatial correlations are completed individually rather than simultaneously as in regular convolutional layers (Chollet, 2017). This concept is very similar

to depth wise separable convolutions (DSC). DSC aims to decrease computation in early layers (Howard et al., 2017). DSC is a convolutional layer with two steps consisting of a depth wise convolution and a 1x1 convolution (pointwise convolution), as shown in figure 11. The depth wise convolution employs a single filter on every input channel. This is followed by the application of the pointwise convolution, combining the results of the depth wise convolution. Through separating the convolution into a filtering layer and a combining layer, it reduces the model size and computational power demands (Howard et al., 2017). Inception modules differ from DSC in that DSC use a depth wise convolution followed by a 1x1 convolution, but inception modules use the 1x1 convolution first (Chollet, 2017). In Inception modules, both convolutions are followed by a ReLu non-linearity, but DSC do not follow with non-linearities.

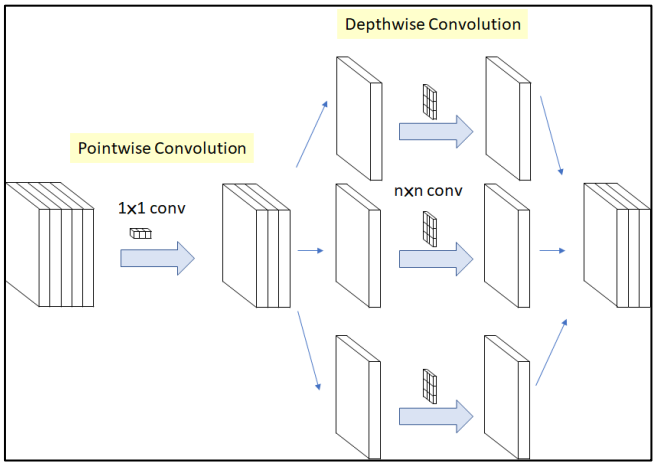


Figure 11: This image shows the mechanism behind DSC as in the pre-trained CNN Xception (Medium: Towards Data Science, 2018).

Chollet (2017) proposed a new model, called Xception, meaning extreme Inception. The Xception architecture replaces all Inception modules in Inception with DSC. This model outperformed Inception v3 on the ImageNet dataset. The computational requirements of the Inception/Xception family is much lower than deeper models such as VGG, making it suitable in cases of limited memory and computational power and for use in mobile applications (Szegedy et al., 2016). MobileNets, like Xception, are built mainly from DSC. MobileNets use ReLu functions for each layer. It uses 3x3 DSC resulting in 8-9 times reduction in computational power compared to normal convolutions with only slight decrease in accuracy (Howard et al., 2017). MobileNet has a total of 28 layers if the depth wise and pointwise convolutions are considered separate layers.

3.18 Hyperparameter choice in CNNs and Transfer Learning

Once a model is chosen, there are still several options and decisions available about the model instance before training begins (VanderPlas, 2016). These choices are often referred to as hyperparameters. They are parameters that are decided before fitting the model to the data rather than parameters learned from the data. According to Smith (2018), it can require years of experience to select the optimal hyperparameters in neural networks. Typical hyper-parameters for CNNs include learning rate, batch size and momentum. Smith (2018) claims these hyperparameters can be set using only a small number of epochs and monitoring the differences in the models. This can help save time compared with running a grid search.

3.18.1 Learning rate

The learning rate is a critical hyperparameter when using SGD (Witten, 2016). The choice of learning rate can strongly influence the final performance of a CNN (Mishkin et al., 2016). Small values such as 0.001 are often efficient (Witten, 2016). The smaller the learning rate, the smaller the adjustment to the weights of the model for each iteration (Haykin, 1994). This can lead to slow training times. If the learning rate is too small, overfitting can occur (Smith, 2018). If the learning rate is too large, the weight adjustments are large for each iteration and the model can become unstable (Haykin, 1994).

3.18.2 Momentum

Momentum seeks to reduce instability in a model (Haykin, 1994). It is an optimization technique for SGD that accelerates SGD in the relevant direction and reduces fluctuations by adding a small percent of the update value from the preceding iteration to the new weight change (Ruder et al., 2017; Witten, 2016). This can help smooth the learning process making changes to the weights less abrupt (Witten, 2016). Both Ruder et al. (2017) and Mishkin et al. (2016) suggest setting momentum to 0.9 or similar. Nesterov accelerated gradient (NAG) is an add on to momentum (Ruder et al., 2017). Momentum calculates the gradient and then accelerates in the direction of the updated gradient. When NAG is used, momentum first accelerates in the direction of the previous updated gradient, measures the current gradient

and then corrects itself. This addition can speed up training and improve convergence (Ruder et al., 2017).

3.18.3 Batch Size

Batch size refers to the number of training examples in one iteration. According to Mishkin et al. (2016), batch size provides a difficult trade-off between computational power and accuracy. In general, GPU systems show a preference for a larger batch size compared to CPU (Mishkin et al., 2016). Mishkin et al. (2016) explore the relationship between batch size and the final accuracy of a model. They found that large batch sizes (>512) reduced model performance. Keskar et al., (2017) reiterate this, stating that large batch sizes limit a model's ability to generalise well to a test set and that there exists a batch size threshold, over which the model quality is reduced. Mishkin et al. (2016) found the optimal model performance using a constant learning rate with a batch size between 64 and 256.

3.18.4 Adam Optimizer

Although SGD is the most common and popular NN optimiser (Ruder et al., 2017), a different optimising technique, such as Adaptive moment estimation (Adam), may be set before NN training. The Adam optimiser was proposed by Kingma et al. (2015). It is a variant of SGD which has a small memory requirement (Kingma et al., 2015). It combines two other optimisers: Adagrad and RMSprop. Adagrad is a gradient-based optimisation algorithm that adapts the learning rate to the model parameters (Ruder et al., 2017). It involves large updates for irregular parameters and slight updates for regular parameters. It is suited for sparse data and eliminates the need for tuning the learning rate. RMSprop is another adaptive learning rate optimiser which divides the learning rate by an exponential decay (Ruder et al., 2017). Adam combines the benefits of Adagrad and RMSprop, computing the adaptive learning rate for each parameter. Ruder et al. (2017) suggest using Adam is optimal for fast convergence and for use in large NN designs such as CNNs.

3.19 Model Validation in Machine Learning

Model validation involves applying the trained model to a new dataset and comparing its predictions to the actual values (VanderPlas, 2016). If a model is both trained and validated

on the same data it is likely to achieve an accuracy of 100%, as the model has already been exposed to this exact data. Holding back a subset of data from model training allows us to check how a model is realistically performing and how it would generalise to another dataset (VanderPlas, 2016).

When using machine learning, it is vital to separate the data into three separate subsets: a training set, validation set and test set (Witten, 2016). The test set should be set aside for the final evaluation. The validation set is involved in running tests to find the optimal model before the final model evaluation. This involves tuning the hyper parameters (Witten, 2016). When the optimal hyperparameters are decided, the model can be evaluated against the test set. It is important that the validation set is not also used as the test set, as it would give a misleading estimate on the generalisation of the model.

3.20 Under fitting and over fitting in machine learning

The notion of finding the best model in ML involves finding the optimal spot between bias and variance (VanderPlas, 2016). A model that has under fit the data does not have enough flexibility to account for all the data and features within the data. Often such a model attempts to find a straight-line fit through the data when the data points are much more complex than a straight line. This is known as bias (VanderPlas, 2016). Bias measures how much a model output differs from the true or desired function (LeCun, 1998). In early training, this measurement is large as the output differs significantly from the true value (LeCun, 1998).

Variance measures to what degree the output varies between datasets (LeCun, 1998). A model that over fits the data has accounted for random errors and noise within the data as well as the underlying structure of the dataset (VanderPlas, 2016). Such a model will not generalise well to test data. This model is said to have high variance. In early training this value is small as the data has not influenced the model yet (LeCun, 1998). Late in training, bias is much smaller as it has succeeded in learning the fundamental function. However, if the model is trained for too long, it will have learned other noisy features only specific to that dataset (LeCun, 1998). Minimal total error occurs when the sum of bias and variance are also minimal (LeCun, 1998). High bias models can be seen when the accuracy of the validation set is like that of the training set. High variance models are seen when the accuracy of the validation set is much lower than on the training set.

Deep learning involves large architectures that make them prone to over-fitting, even with large datasets (Witten, 2016). A number of techniques exist to optimise a model's ability to generalise to unseen datasets and reduce overfitting, for example, early stopping and dropout layers (LeCun, 1998).

3.21 Techniques to reduce overfitting

There are many techniques proposed to reduce overfitting in CNNs. Popular methods include data augmentation, early stopping, dropout, and a global average pooling layer.

3.21.1 Data augmentation

Using transformation/augmentation techniques to expand the size of a dataset is a pivotal technique in deep learning. The data available for training is increased which prevents over-fitting and allows the model to better generalise to unseen data (Witten, 2016). Data augmentation is important for optimal results. Transformations for image classification includes rotations, horizontal and vertical flips, and cropping.

3.21.2 Early stopping

Early stopping can also help reduce over-fitting in training (Witten, 2016). This involves plotting learning curves for the average loss as a function of the epoch for both training and validation sets (Witten, 2016). The point where validation average loss starts to decrease is the early stopping point.

3.21.3 Dropout

Dropout involves randomly deleting units and corresponding connections during training. This reduces the level of coadaptation in hidden layers and reduces over-fitting (Witten, 2016). Hinton (2012) suggests a dropout rate of 50%. More than this tends to reduce performance. Mishkin et al. (2016) agrees that a dropout rate of 50% is efficient, adding that the dropout layer should be placed before the final two layers.

3.21.4 Global average pooling layer

According to Lin et al. (2014), fully connected layers make the network prone to overfitting. They propose to replace fully connected layers with a global average pooling layer. This creates one feature map for each class in the last convolutional layer. Rather than adding fully connected layers on top of the feature map, it computes the average of each feature map and creates a vector of the averages. This is fed into the SF layer (Lin et al., 2014).

3.22 Class imbalance

Class imbalance happens when one or more classes in a dataset outnumber the other classes (Celebi et al., 2007). Classifiers tend to direct their attention to learning the large classes, causing low classification accuracy in smaller classes (Celebi et al., 2007). This is particularly common in medical classification tasks, with the healthy state being more common than the unhealthy state (Buda et al., 2017). This imbalance can negatively impact on convergence during training and decreases the model's ability to generalise to the test set (Buda et al., 2017). In medical diagnosis, classifying a minority class such as melanoma as a majority class such as a benign lesion would have frightening consequences (Celebi et al., 2007). It is generally assumed that increasing the number of training samples decreases the difference between the error rate of the test set and training set (LeCun, 1998). There are many methods of dealing with class imbalance (Buda et al., 2017). Over sampling is the most popular method in deep learning. The simplest version is called random minority over sampling. This replicates images in the minority class at random (Buda et al., 2017). This method is known to be effective but may lead to over fitting. Other over sampling methods include augmenting images in the minority class. Under sampling involves randomly removing images from the majority class. A clear disadvantage is that a large proportion of data is lost (Buda et al., 2007).

3.23 Machine Learning in mobile health applications

According to Harangi et al. (2018), mobile dermatoscopes are becoming more available and affordable. These could be attached to smartphones, improving patient care. High resolution cameras are a feature of most modern smartphones, making an app to assess suspicious skin lesions a possibility (Harangi et al. 2018). Although there appears to be the potential for the development of such apps, there appears to be a lack of enthusiasm from skin cancer patients around these applications. In a study by Steeb et al. (2019), 67% of patients would prefer more advice to be readily available, outside of hospital environments. Several skin care related apps have been launched to meet these demands for information. These apps include information on how to self-examine, tracking skin changes, information on preventing sunburn and computer-based algorithms to detect skin cancer. Interestingly, only 8.7% of skin cancer patients in the study had tried such applications. A limited number of patients had also tried a skin cancer related app called Skin Vision. Skin Vision is a skin cancer detection app. It allows the user to submit a photograph of their skin and outputs the risk of the user developing skin cancer (low to high). It also helps the user to schedule a doctor's appointment if necessary. Patients voted the most important aspect of health apps is access to scientifically credible information, convenience of use and data security.

43% of patients in the study believed skin cancer apps can support skin cancer diagnosis by a medical professional. However, most patients (76%) believe that these apps, however useful, cannot replace a medical opinion. 50% of patients would be interested in learning more about these apps, with 60% of patients agreeing that they would download an app if recommended by their physician. This study suggests so far that the awareness and demand for such apps may be low. However, it is important to note that the mean age for this study was 66.

3.24 Previous work using CNNs in the literature for skin cancer detection

In the literature, there are several studies which apply CNNs and pre-trained CNNs to skin lesion classification. These tasks include both binary and multi-classification tasks. Binary classification tasks include the work of Esfahani et al. (2016), building a CNN and training it from scratch in order to classify lesions into either MSC or nevus. Further, Esteva et al. (2017) designed a CNN model for two binary classification tasks: the first to distinguish between keratinocyte carcinomas and benign seborrheic keratosis and the second to

distinguish between MSC and benign nevi. Pre-trained CNNs and transfer learning have also been applied to binary classifications in this area. Shoeib et al. (2016) used the pre-trained CNN, AlexNet, as a feature extractor and a SVM as the classifier to classify clinical images into either MSC or NMSC. Similarly, Pomponiu et al. (2016) utilised a pretrained CNN as a feature extractor and a KNN classifier to classify clinical images into malignant or benign. Ali et al. (2017) used the pretrained model, LightNet, as both a feature extractor and a classifier in the classification of benign and MSC images.

Multi-classification tasks include the work of Kawahara et al. (2016), using AlexNet as a feature extractor and logistic regression to classify dermoscopy images into 10 classes. The classes were not balanced, with five majority classes and five minority classes. They achieved a score of 85.8% for the five majority classes and 60% for the classes with limited images. The average accuracy for the 10-class model was 81.8%. Revantalab et al. (2018) compared the accuracy of different pre-trained CNNs with the accuracy of expert dermatologists to classify multiple classes of skin lesions. They found that these pre-trained models had a higher accuracy than dermatologists at an average of 11%. However, there was significant class imbalance in the dataset and these results may be unrealistic. Harangi (2018) introduced a novel method for the classification of nevus, MSC and NMSC. They used an ensemble CNN architecture. This involved merging the outputs of GoogLeNet, AlexNet, ResNet and VGG Net. All votes of classifiers in an ensemble method are considered to compensate for individual limitations. They achieved an accuracy of 86% using this method. This appears to be the highest scoring multi-classification model in this area. However, it only includes three classes. It is likely that including other lesion classes as in the other multiclass studies would reduce this score.

Limitations of the previous work include small datasets (Esfahani et al., 2016; Shoieb et al., 2016; Kawahara et al., 2016), which are often unbalanced (Revantalab et al., 2018). There are also a limited number of experiments using CNNs to classify multiple skin lesions, not just melanoma. In a clinical setting, a lesion is not classified into either MSC or NMSC, as there are many other classes of lesions. Multi-class algorithms would be more insightful and more likely to be integrated in healthcare. I intend to use a pretrained CNN to classify multiple skin lesions, balancing my dataset to achieve a realistic and accurate result. I will include methods regarding how the transfer learning method (i.e. freezing the layers or fine tuning) and suitable hyperparameters were chosen as this information is often missing in the literature. It will investigate if the promising results so far are simply a result of class imbalance or if a

successful model can be developed when all classes are equally represented. Further, the pre-trained models of the Inception style family, including Xception and MobileNet, have not been used for these tasks. As they have a different design it would be interesting to see how these models compare. This study intends to improve on the existing literature, with the hope of building a case for the integration of AI into healthcare.

4 RESEARCH METHODOLOGY

4.1 Chapter Overview

This chapter states the overall research design, details the choice of methodology and justification of choices in the study. This study attempted to develop a model, using pre-trained CNNs, which can accurately differentiate between dermoscopic images of multiple skin lesion classes including MSC and NMSC. It has addressed previously overlooked issues such as class imbalance, identifying the most suitable method of transfer learning in CNNs, and the process of hyperparameter choice. The study progressed through a number of stages:

1. Data collection.
2. Data exploration and solving class imbalance.
3. Selecting a suitable machine learning model.
4. Training the network: determining the optimal method of TL and hyperparameters
5. Evaluating the final model on the test set.
6. Building the model into a Flask App.

All steps were completed using a MacBook Air which does not have a GPU. The methodology includes the challenges associated with this lack of computational power. It also includes an evaluation of the ethical considerations surrounding the study. All steps are carried out using Python in a Jupyter notebook environment. Technologies used in the study include Pandas, MatPlotLib, PIL Image, Scikit-Learn, TensorFlow, Keras and Flask.

4.2 Research design

The research philosophy or research paradigm defines the assumptions of how a researcher will learn during a study (Creswell, 2003), shaping the overall research design (Saunders & Tosey. 2012). Here the research philosophy of the researcher is considered positivism, also known as the “scientific method” (Creswell, 2003). Positivism involves beginning the research with a research question and hypothesis, as has been stated at the beginning of this study, and data is collected and analysed which will either support or refute the hypothesis. Positivism is usually interested in structured data which is measurable and thus the researcher does not influence the overall research (Saunders & Tosey, 2012). The research methods used

in the study will mainly be quantitative. Quantitative methods involve numerically analysing the data collected. It is associated with experiments, mathematical models and statistical analysis. This study used CNNs to develop an accurate model which can classify skin lesions. CNNs are, at a basic level, mathematical models. They are therefore considered a quantitative method. Although the methods are quantitative, it is more accurate to consider this research as "mixed-model research" which combines both quantitative and qualitative approaches at different points in the research. The data collected was qualitative, in the form of dermoscopy images. The images were then converted to quantitative data (pixel values) which can be statistically analysed (Saunders et al., 2009). The strategy of enquiry gives a research design a specific direction for the methodology (Creswell, 2003). In this study the research strategy is experimental. Data was analysed and trained in order to support or refute the original hypothesis. The data collected was in the form of secondary data, data which has already been collected for another purpose (Saunders et al, 2009). The data is documentary and raw in the form of unprocessed dermoscopy images. The availability of secondary data was established during the literature review and specific secondary data was found through subsequent internet searches. The time period over which the research took place is known as the time horizon. As this research was limited by time constraints the time horizon was cross-sectional, the research problem was investigated at a particular point in time and not over a number of years as in longitudal studies (Saunders et al., 2009).
This overall research design influenced the data collection and data analysis which are detailed below.

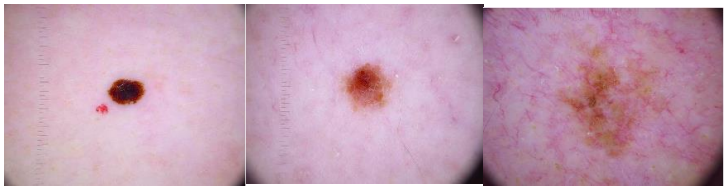
4.3 Data Collection

The dataset, titled "HAM10000: Human Against Machine with 10000 Training Images" was sourced from Kaggle.com. The data was originally gathered and published by Vienna Dermatologic Imaging Research. The dataset includes a total of 10000 dermatoscopic images and a corresponding CSV file (see appendix A for details). The images include seven categories of skin lesions collected from a number of different populations: Actinic keratosis/Bowen's disease (AK), basal cell carcinoma (BCC), benign keratosis-like lesions (BK), dermatofibroma (DF), melanoma (MSC), melanocytic nevi (MN) and vascular lesions (VASC). Images in the folders are not labelled with their lesion class, they are instead named by their image ID. The CSV file contains columns for image ID, lesion class, age of patient,

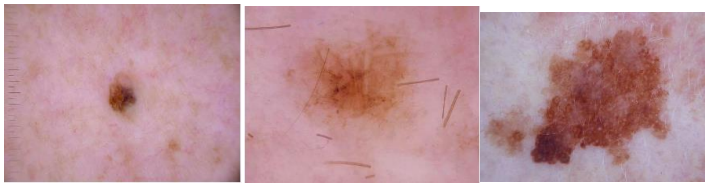
gender of patient, method of diagnosis and location of the lesion on the body. This data was released with the intention of improving the problem of lack of size and diversity of dermatoscopic images available to the machine learning community. The images have not been pre-processed, background noise and hairs remain, which is representative of real-life clinical practice. For an overview of the different skin lesions not detailed in the literature review, see appendix B.

Below in figure 12, shows a number of sample images from each lesion category in the dataset.

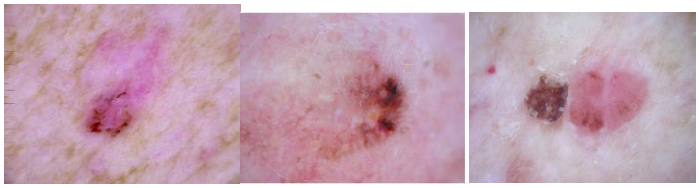
Melanoma



Benign keratosis like lesions

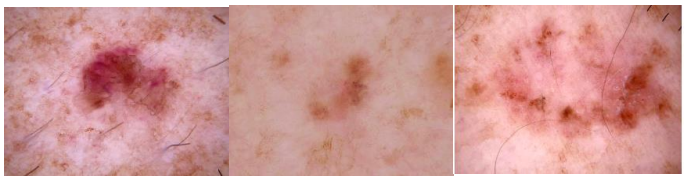


Basel Cell Carcinoma

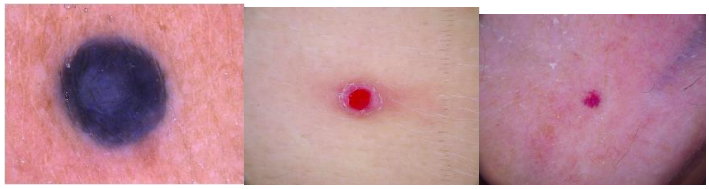


Melanocytic Nevi

Actinic keratosis/Bowen's disease



Vascular lesions



Dermatofibroma

Figure 12: This image shows an overview of the different lesion classes in "HAM10000".

To connect images to their respective labels, the CSV file (as shown in appendix A) was converted to a Pandas Dataframe. Pandas is an open sourced library in python which includes tools to structure data for easy data analysis. A Pandas Dataframe is a two-dimensional data structure with labelled rows and columns. It allows many operations to be applied such as deletions, additions and search operations. A new column was added to the dataframe called "path". This was created by setting out the default path to the folder the data was stored in and adding the image ID for each row of the Dataframe. Now each row had both an image path and a corresponding label, as shown in figure 13.

	lesion_id	image_id	dx	dx_type	age	sex	localization	path
0	HAM_0000118	ISIC_0027419	bkl	histo	80.0	male	scalp	/Users/lauradempsey/documents/Dissertation/dat...
1	HAM_0000118	ISIC_0025030	bkl	histo	80.0	male	scalp	/Users/lauradempsey/documents/Dissertation/dat...
2	HAM_0002730	ISIC_0026769	bkl	histo	80.0	male	scalp	/Users/lauradempsey/documents/Dissertation/dat...
3	HAM_0002730	ISIC_0025661	bkl	histo	80.0	male	scalp	/Users/lauradempsey/documents/Dissertation/dat...
4	HAM_0001466	ISIC_0031633	bkl	histo	75.0	male	ear	/Users/lauradempsey/documents/Dissertation/dat...

Figure 13: The CSV file converted to a Pandas DataFrame with new column 'path'.

4.4 Data exploration and solving class imbalance

Following a value count of the unique classes in the Dataframe, it was discovered that the data was extremely unbalanced. As shown in figure 14, 6,700 images were from a single class, MN, with other minority classes, for example DF, containing 115 images.

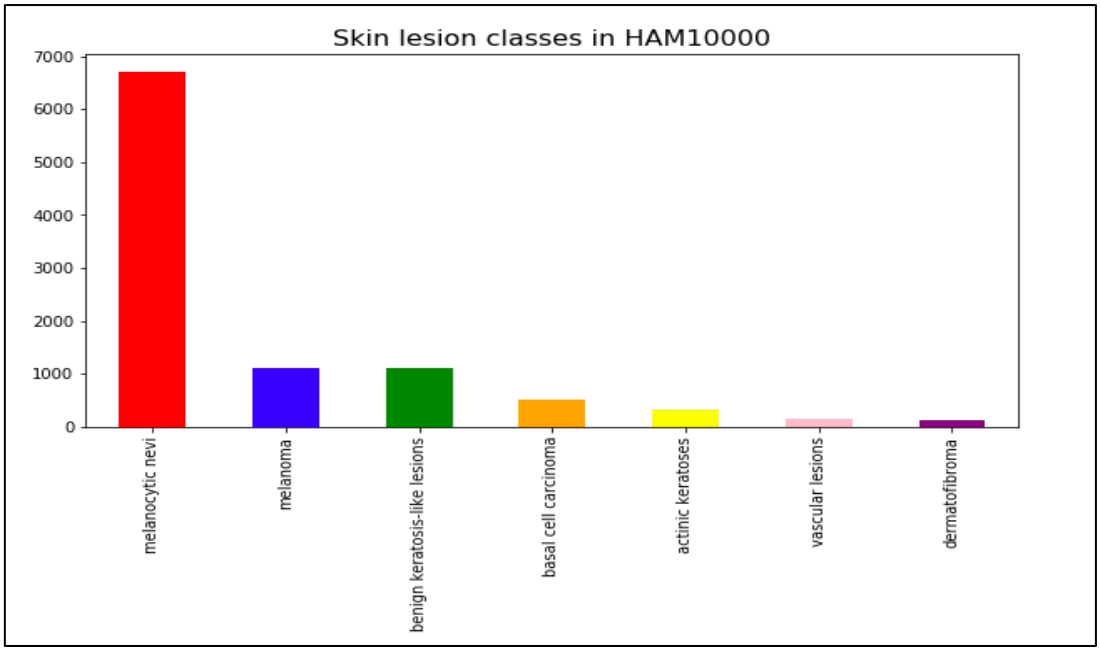


Figure 14: This chart shows the distribution of class types in the dataset "HAM10000".

As stated by Celebi et al. (2007), class imbalance causes a model to direct its attention to learning the larger classes, resulting in low accuracy for the minority classes. This limitation is often not addressed in the literature. For this reason, it was decided to balance the image dataset. This involved both oversampling/artificially expanding the minority classes (VASC, DF, AK, BD) and undersampling/reducing the majority class (MN). Although random minority oversampling is a popular method for reducing class imbalance, it can result in overfitting (Buda et al., 2017). As CNNs and other large model architectures are at risk of overfitting, it was decided to artificially expand the minority classes using rotations and flips, rather than randomly replicating images (see figure 15). The images belonging to the classes BCC, AK, VASC and DF were all exposed to rotations of 90, 180 and 270 degrees, along with vertical and horizontal flips. These transformations were carried out using Pillow (PIL), a Python image library. PIL has an image module which can open, display, rotate, flip and crop images in Python.

Figure 15: Above shows an example of data augmentation using rotations and flips.

MN images were subjected to undersampling by randomly selecting 1,000 MN images to keep, and removing all other images belonging to this class. All images were then saved to one of seven folders labelled by their class. A new Dataframe was then created with two columns, one for the image label and one for the path to that image (as shown in figure 16). The class balance in this dataframe is now much improved and more representative of each skin lesion.

	label	image_path
0	nv	/Volumes/memory/data/nv/1100.jpg
1	nv	/Volumes/memory/data/nv/4662.jpg
2	nv	/Volumes/memory/data/nv/6572.jpg
3	nv	/Volumes/memory/data/nv/6256.jpg
4	nv	/Volumes/memory/data/nv/516.jpg
5	nv	/Volumes/memory/data/nv/2089.jpg
6	nv	/Volumes/memory/data/nv/965.jpg
7	nv	/Volumes/memory/data/nv/4058.jpg
8	nv	/Volumes/memory/data/nv/6233.jpg
9	nv	/Volumes/memory/data/nv/3682.jpg
10	nv	/Volumes/memory/data/nv/3868.jpg
11	nv	/Volumes/memory/data/nv/5337.jpg
12	nv	/Volumes/memory/data/nv/3109.jpg
13	nv	/Volumes/memory/data/nv/6461.jpg
14	nv	/Volumes/memory/data/nv/1719.jpg
15	nv	/Volumes/memory/data/nv/768.jpg

Figure 16: This image shows the first 15 rows of the new dataframe to be used in the study. It includes a class label and the direct path to the image.

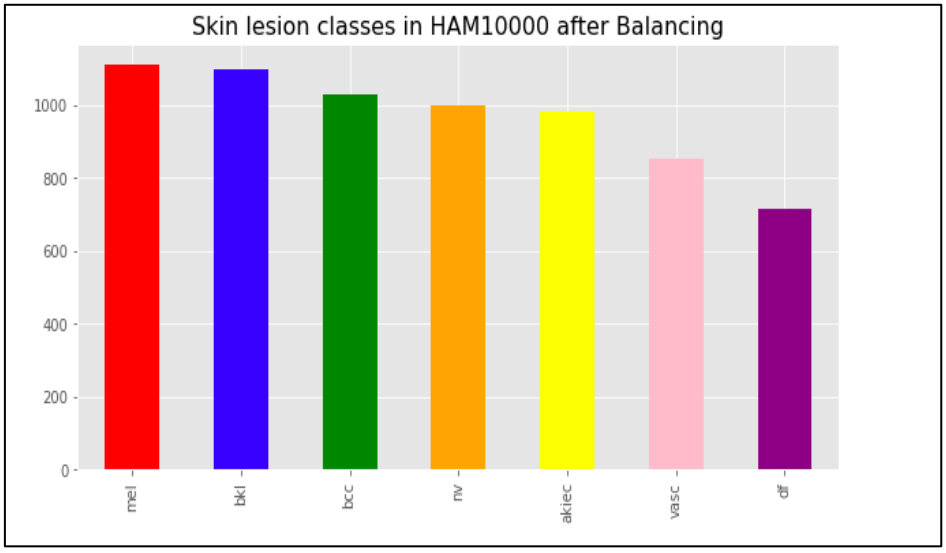


Figure 17: This graph illustrates the improved balance within the dataset classes.

4.5 Selecting a suitable machine learning model

As traditional image processing methods involve complex pre-processing techniques and expert feature engineering before classification, it was decided that CNNs were the most appropriate choice of model. CNNs overcome the need to manually extract features and do not require image pre-processing techniques (Shoeib et al., 2016). Due to insufficient computational power and inexperience in neural network design, it was decided to use pre-trained CNNs which would save time on model design and training. This is a popular choice in the medical field (Kieffer et al., 2017). Deeper (models with more layers and parameters) pre-trained CNNs such as VGG Net typically result in higher accuracy but demand long training times and large computational power (Howard et al., 2017). Due to time constraints and lack of computing power (no access to GPU), these deeper models would not be ideal. It became clear from the literature that the Inception family of models, designed for cases of memory and computational constraints, was the optimal choice. Specifically, the Xception model was chosen as it outperforms all other Inception models to date. Xception has a size of 88MB and a depth of 126. The structure of MobileNet is similar to Xception, built on DSC. It has a memory of 16MB and a depth of 88. Due to the compact size of MobileNet, it provides faster training times than Xception. As time and computational power was restricted, it was decided to use MobileNet for initial model training on a small subset of images in order to find the optimal model hyperparameters in the fastest possible time.

4.5.1 Selecting a Deep Learning Platform: Keras and TensorFlow

TensorFlow is a large-scale machine learning system. It can work with CPUs, GPUs and TCUs. It is open sourced, released by Google and widely used in machine learning research (Abadi et al., 2016). Keras is a popular API deep learning framework for neural networks. It is written in Python and can run on top of TensorFlow. It is user friendly, supports CNNs and is efficient at running on both GPU and CPU. Both Xception and MobileNet are available through Keras. Keras has a ‘flow from data frame’ function which takes a class label and path to the image as an input and stores the specified image. This function is suitable for the structure of the data frame used in the study. For model training, Keras with TensorFlow backend was used.

4.6 Training the network: determining the optimal method of TL and hyperparameters

Prior to training the final model, to get a good indication of the appropriate model parameters and transfer learning methods to use in a reasonable amount of time, a small subset of approximately 1,000 images were used. The subset of images was split into a training set and validation set using Scikit-Learn (training:814, validation: 204). As MobileNet requires an image shape of 224 x 224, the images were resized to this shape. This was completed using the Keras Image Data Generator. This tool, combined with the 'flow from dataframe' module, retrieves the image specified by a path in the Dataframe, generates the image and applies any necessary processing techniques such as resizing.

Three tests were carried out to explore the optimal method of transfer learning for this task. These tests included transfer learning which involved freezing all layers except the classification layer and using all the pretrained weights. The second test and sub tests involved fine tuning. This involved freezing parts of the model and retraining the rest to this specific dataset. The final test involved retraining the entire architecture from scratch. An overview of these tests can be seen in figure 18. The hyperparameters used for each method remained constant. Batch size was set to 32 because CPU systems usually require smaller batch sizes compared with GPU. The MobileNet model was loaded with weights/parameters of ImageNet and the top classification layer removed. Rather than applying additional FCL, a global average pooling layer was added, followed by a SF classification layer. The use of global average pooling layer instead of FCL is suggested by Lin et al. (2014) as it reduces overfitting compared with FCL. The model used a SGD optimizer with an initial default learning rate of 0.01 as SGD is the most popular optimizer in the literature (Ruder et al., 2017).

Figure 18: An overview of the three methods of transfer learning explored.

4.6.1 Transfer Learning

The fastest and most popular method of using pre-trained neural networks is using the pretrained layers as a feature extractor and modifying the classification layer for the specific task (Yigit et al., 2018). As this is the simplest form of transfer learning, this is the first method attempted. All layers except the global average pooling layer and SF layer were frozen. The model was trained using early stopping and validated by the validation set. Early stopping reduces the probability of overtraining the model, which can lead to overfitting and unnecessarily long training times. The training set accuracy gradually increased to 100% accuracy, however the validation accuracy was fluctuating between 18-22% (see figure 19).

```
Epoch 1/75
25/25 [==============================] - 81s 3s/step - loss: 2.0176 - acc: 0.2525 - val_loss:
2.0953 - val_acc: 0.2031
Epoch 2/75
25/25 [==============================] - 69s 3s/step - loss: 1.6385 - acc: 0.3701 - val_loss:
1.9670 - val_acc: 0.2442
Epoch 3/75
25/25 [==============================] - 71s 3s/step - loss: 1.4612 - acc: 0.4372 - val_loss:
1.9587 - val_acc: 0.2267
Epoch 4/75
25/25 [==============================] - 68s 3s/step - loss: 1.2562 - acc: 0.5255 - val_loss:
2.0789 - val_acc: 0.1977
Epoch 5/75
25/25 [==============================] - 67s 3s/step - loss: 1.2144 - acc: 0.5455 - val_loss:
2.1598 - val_acc: 0.1860
Epoch 6/75
25/25 [==============================] - 67s 3s/step - loss: 1.1541 - acc: 0.5831 - val_loss:
2.1335 - val_acc: 0.1628
```

Figure 19: This shows a snapshot of the training with the transfer learning method using MobileNet and Keras.

Considering the model has a 1/7 (14%) chance of correctly classifying the lesion by chance, this is far from impressive. The large difference between training accuracy and validation accuracy is an indicator of overfitting.

4.6.2 Fine Tuning

The second method attempted involved using the pretrained weights for some of the architecture and training other parts from scratch. This is known in the literature as fine tuning. As there is limited knowledge available regarding the number of layers to freeze and in general it is dependent on the dataset, trial and error was used here. MobileNet has 28 layers. Tests were carried out fine tuning on the final 2, 6, 10, 15, 20 and 25 layers. This means all layers were frozen except, for example, the final 12 layers. As general features are usually learned in early layers, it would be expected that freezing the first number of layers and retraining on the rest of the model would offer some improvement. However, even after 25/28 layers were retrained, validation accuracy only reached 52% while training accuracy was still reaching 100%.

4.6.3 Retraining the entire architecture

As the fine tuning and transfer learning methods were not proving effective, the final test involved fine tuning on the entire architecture i.e. training the model from scratch. The first attempt involved removing the top layer, adding a global average pooling layer and SF layer. Using the SGD optimizer with default learning rate (0.01) and batch size of 32. This model was trained on all layers. The validation accuracy significantly improved using this method,

fluctuating between 60-70% (as seen in figure 20). However, training accuracy was reaching 100%, indicating the model was still overfitting. Although overfitting was occurring, it was clear that this method of transfer learning was optimal for this task.

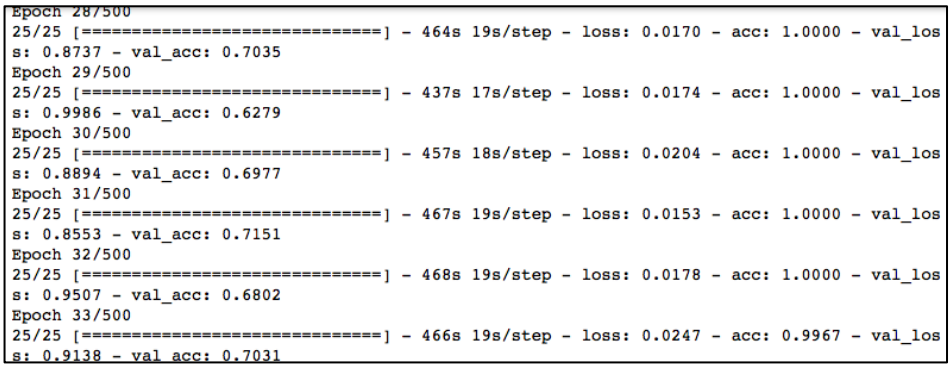


Figure 20: This image shows a snapshot of model training for retraining the entire MobileNet architecture.

4.6.4 Finding the optimal hyperparameters

As retraining the entire architecture from scratch was deemed the optimal method of transfer learning, the remainder of tests on the subset of images involved exploring different hyperparameter settings to improve the model. These included: momentum, learning rate, Adam optimizer, Nesterov momentum, batch size and dropout layers.

4.6.4.1 Adding Momentum

This test uses the same model architecture as before but adding momentum. Momentum is an optimisation technique for SGD which can reduce instability in the model and reduce overfitting. Frequently in the literature, a value of 0.9 is suggested for momentum. Using momentum of 0.9 and keeping all other parameters the same, the model was trained again. Momentum slightly improved model performance.

4.6.4.2 Adjusting the Learning Rate

Up until this point the default learning rate, 0.01 had been used. It is known from the literature that learning rate can impact strongly on the final model performance. LR of 0.001,

0.0004, 0.0005, 0.0006 and 0.0001 were tested with momentum of 0.9. The optimal learning rate was found to be 0.0005 achieving validation accuracy of 72%, as shown in figure 21.

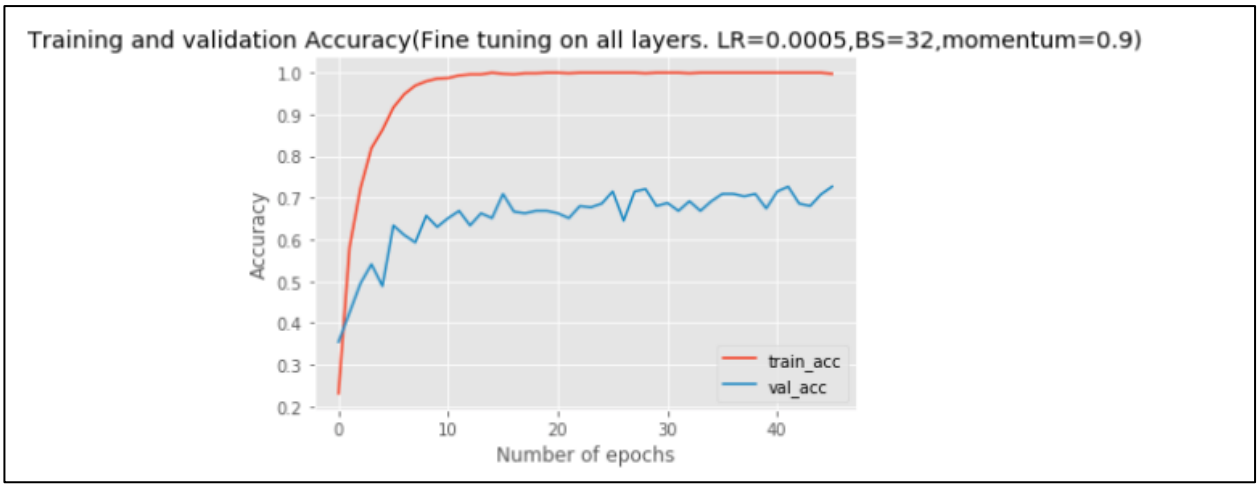


Figure 21: This graph shows the training history of the model with the optimal learning rate 0.0005. The left side shows the model accuracy and the right side shows model loss.

4.6.4.3 Replacing SGD with Adam Optimizer

According to Kingma et al. (2015) the Adam Optimizer can often outperform SGD and has a smaller memory requirement. This was tested with the MobileNet model but found that model performance and stability were significantly reduced, as shown in figure 22.

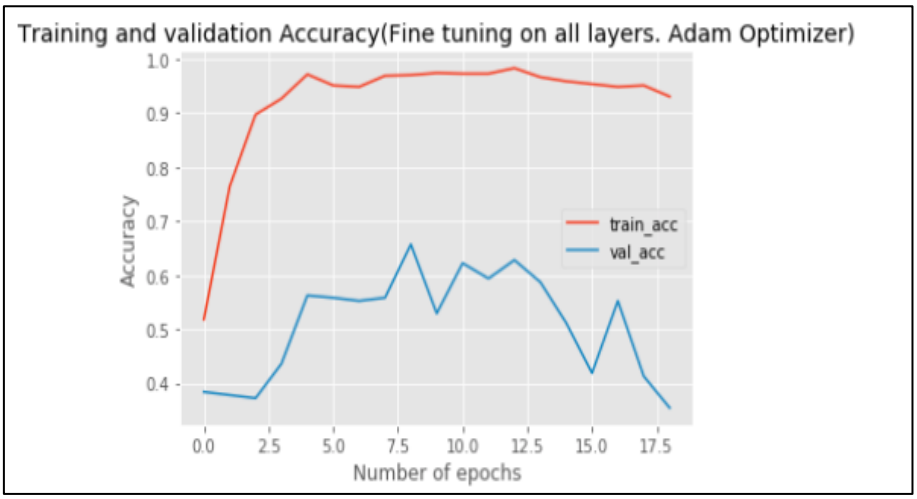


Figure 22: This graph shows the training and validation accuracy using the Adam Optimizer.

4.6.4.4 Adding Nesterov Momentum

As stated in the literature, Nesterov is an effective addition to momentum, known to speed up training times and improve convergence. When tested on this model, using the same hyperparameters, Nesterov appeared to improve model performance with validation accuracy peaking at 77% as shown in figure 23.

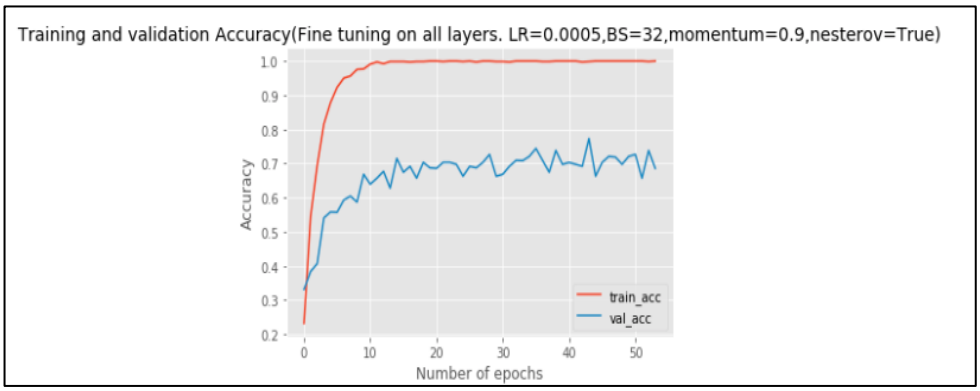


Figure 23: This graph shows the model accuracy using Nesterov Momentum.

4.6.4.5 Increasing Batch Size

Batch size is known to affect model performance, with larger batch sizes tending to reduce performance. However, in the literature, a batch size of 64 appears optimal for both CPU and GPU systems. Using the same hyperparameters as in the Nesterov test, the batch size was increased to 64. This appeared to decrease model performance with validation accuracy reduced to 60%, as shown below in figure 24.

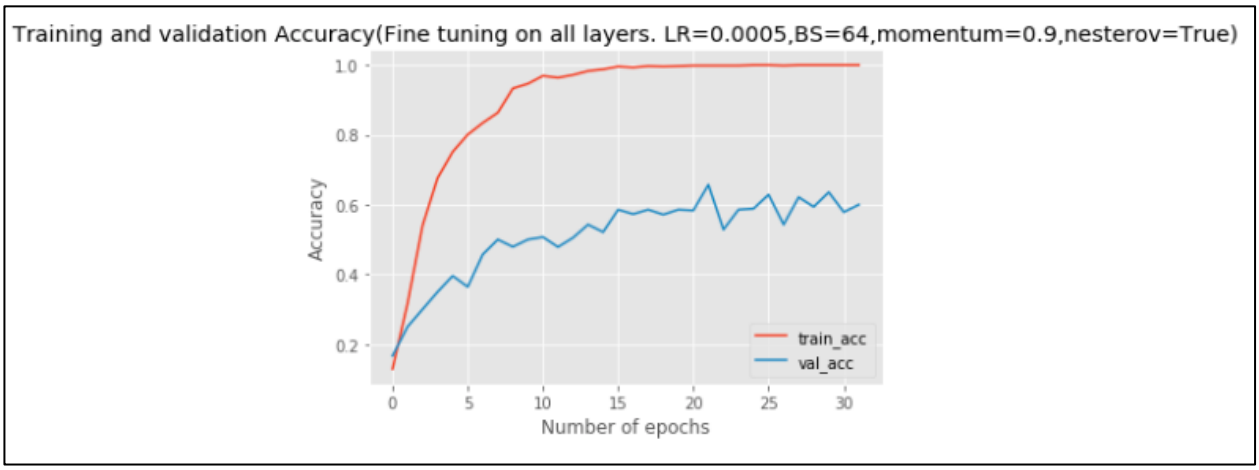


Figure 24: This graph shows the performance of the model using a larger batch size (64).

4.6.4.6 Adding Dropout Layers

Dropout layers are known to reduce overfitting. In the literature it is suggested to use a dropout rate of 50%, before the final two layers. A dropout layer of rate 50% was added before the global average pooling layer and final SF layer. Dropout slightly decreased performance in the model with validation accuracy peaking at 73%, as shown in figure 25.

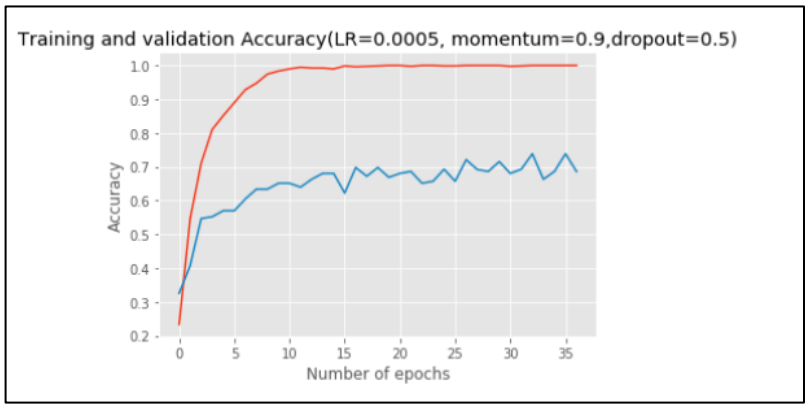


Figure 25: This graph shows the performance of the model with a dropout layer.

4.6.5 Final model training: Training the entire dataset

Final training involved training the full dataset, using the optimal hyperparameters and transfer learning methods derived from the subset of images. The optimal hyperparameters found were a learning rate of 0.0005, with Nesterov momentum of 0.9, batch size 32 and fine tuning on the entire architecture. The final dataset was split into three subsets: Training (5499), validation (679) and testing (612). The two pre-trained CNNs, MobileNet and Xception, were compared. An additional test, replacing the SF classification layer of MobileNet with a KNN classifier was also compared. As additional data increases a model's ability to converge and reduces overfitting, it was hoped that significant improvements in model performance would be observed.

4.6.5.1 MobileNet

All images were resized to 224x224. Early stopping was used, and the model was trained from scratch. The training stopped after 23 epochs. This model took 12.5 hours to train, an average of 33 minutes per epoch. The model reached a validation accuracy of 85%, significantly improving on the model trained on a smaller dataset (see figure 26 and 27).

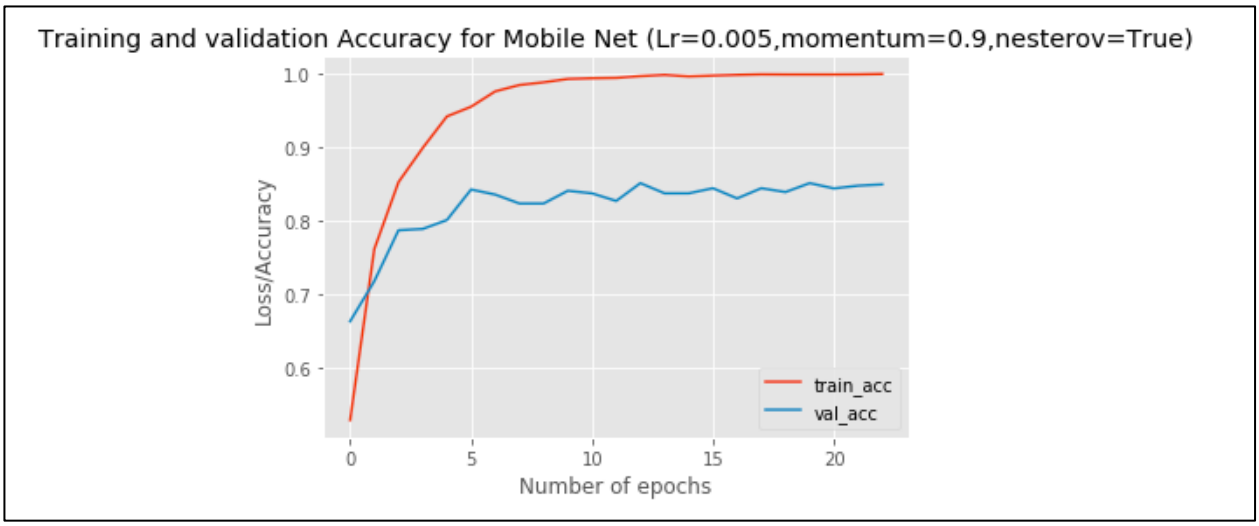


Figure 26: This figure shows the model performance of MobileNet trained on the entire dataset.

4.6.5.2 Xception

Prior to training Xception, images were again resized to 299x299 using PIL as this is the format Xception expects. This model stopped training after 24 epochs. The model took 105.14 hours to train (over four days), an average of 4.38 hours per epoch. The validation accuracy was slightly lower than MobileNet, peaking at 82% (see figure 27).

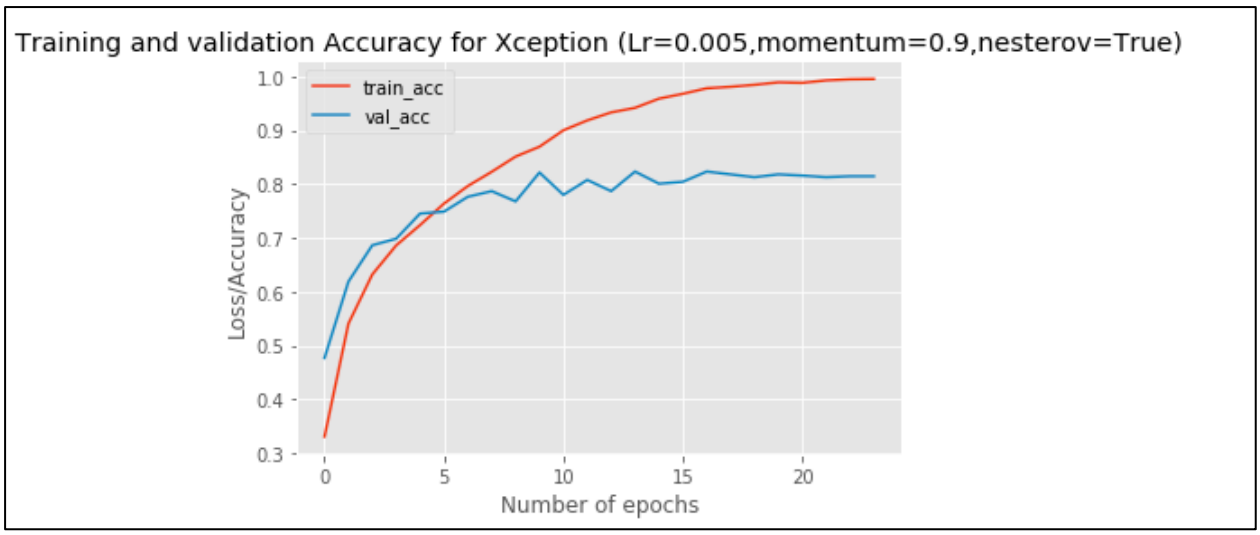


Figure 27: This graph shows the model performance of Xception, trained on the entire dataset.

4.6.5.3 MobileNet with a KNN classifier

In the literature, CNNs have been effectively combined with other machine learning classifiers such as SVM (Shoeib et al. ,2016) and KNN (Pomponiu et al. 2016) for binary classification of MSC and NMSC. It would be interesting to observe if these results transfer to a multi-classification model. The final test attempted was using the trained MobileNet model (from 4.6.5.1) as a feature extractor on the test set, removing the final SF layer and replacing it with a KNN classifier. To find the optimal value for 'k', the features of the validation set were extracted using the trained MobileNet model and a grid search using Scikit-Learn was carried out to test which value of k would yield the highest accuracy. The results favoured a value of 6 nearest neighbours. This model would now be applied to the test set.

4.7 Evaluating the final models on the test set.

The trained MobileNet model, Xception model, and MobileNet-KNN model were all tested for their accuracy on the test set. The MobileNet model achieved the highest accuracy, table of results can be found in the next chapter.

4.8 Building the model into a Flask Application

The MobileNet model was saved, with the intention of using it in a proof of concept application. To build the web application, first two HTML pages were created. The first HTML page consists of the homepage and a description of the model. The second HTML page, which is linked to the first, allows the user to upload their own image. To link the HTML pages, the uploaded image, and the trained MobileNet model, Flask was used. Flask is a web framework. It contains tools and technologies for developing a web application. The user uploads an image through the second HTML page, which the Flask app redirects to the MobileNet model. The model then classifies the image into one of seven learned categories and outputs a message. Snapshots of the webpage can be seen below in figure 28 and figure 29. For all HTML and Flask code, see appendix C.

Figure 28: This is an image of the Home page of the Flask App. Clicking on the 'here' link redirects the user to the second HTML page as shown below in figure 28.

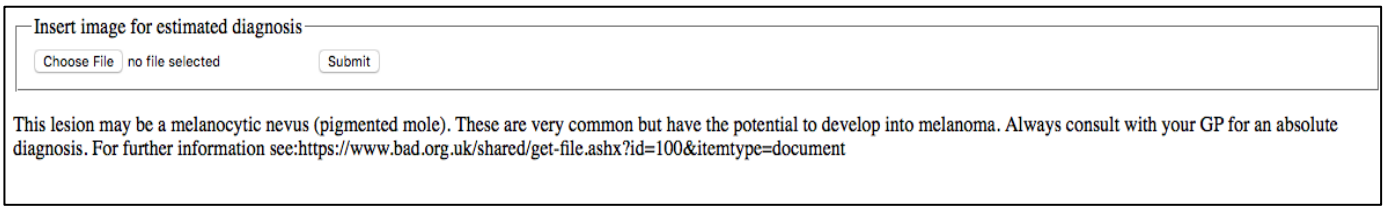


Figure 29: This is an image of the second HTML page from the Flask App. It allows the user to upload a file, sends the file to the MobileNet model and outputs a message depending on the prediction.

4.9 Ethics of Methodology

All research involving data collection from human subjects involves ethical considerations (Oliver, 2010). Oliver (2010) asks researchers to consider the moral justification of research methods. Here, the goal is to contribute to and add knowledge to the subject of the use of AI in healthcare and specifically dermatology. By contributing to the research of computerised diagnostic systems, the use of AI in this field moves closer to reality. The implications of this include AI as a second diagnostic opinion, mobile applications to give warning signals to patients and faster diagnosis which could ultimately save a life. It is important to state that this research is not intended to suggest that dermatologists and medical professionals should be replaced by automated systems. It is merely suggested as a tool to improve efficiency of diagnosis.

The use of data mining techniques involving data about people should be considered. The data used should be collected ethically and participants must be clear on how their data is used (Witten, 2016). Participants are entitled to anonymity. This is especially true when dealing with medical data. If anonymity is applied successfully, there should be no way a participant can be recognised or reidentified from any piece of information, for example, date of birth. In this dataset, the CSV file contains only an ID for the lesion, lesion class, age of patient, gender of patient and location on the body. None of this information would lead to the identification of the original patients. The original researchers removed any information from their database which could potentially identify a patient. The original data collection was also approved by the ethics committee in both sites in Vienna and Queensland. It is also important to consider the ethics of reusing data from the previous study. The consent given for the original study may not necessarily transfer to this study. However, as the aim of this original research was to gather data to help the machine learning community, it is assumed that participants were aware of its intended use.

5 RESULTS AND FINDINGS

5.1 Chapter Overview

This chapter details the results from the methodology discussed in the previous chapter. It includes the table of results for the different transfer learning methods, an overview of the results of different hyperparameter settings, and the results of model performance on the test set of the final pre-trained models: MobileNet, Xception and MobileNet-KNN.

5.2 Results of Transfer Learning and Fine Tuning on the subset of data

Three methods using the pre-trained CNN, MobileNet, were tested on a subset of 1000 dermoscopic images. The first method, transfer learning, involved using the pre-trained weights for all layers except the final layer which was replaced with a new softmax layer for seven different categories. This method was unsuccessful (as shown in figure 30), only reaching 25% accuracy on the validation set while achieving 100% accuracy on the training set. This large gap in accuracy between the training set and validation set is a clear indicator of overfitting, a model which will not generalise well to an unseen dataset. This model has perfectly accounted for the exact detail and background noise of the training set. It learns the training set so precisely that it cannot be used on a new dataset (VanderPlas, 2016). This method used the pre-trained weights as a feature extractor. These features were originally learned from the ImageNet dataset, containing many categories of images such as objects and animals. These general objects in the ImageNet dataset are very different from medical images such as the dermoscopic lesions in HAM10000. This difference is probably the primary reason these features did not transfer to HAM10000.

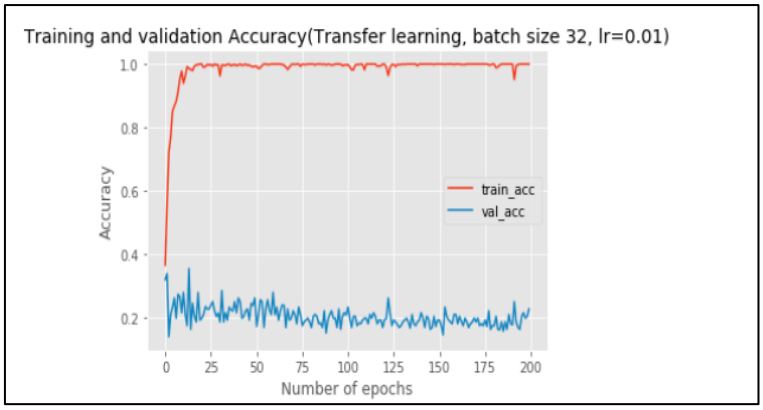


Figure 30: This is a graph showing the training and validation accuracy of the transfer learning methods using MobileNet on 1000 dermoscopic images.

The second method involved fine tuning the model: freezing some layers and training the remainder from scratch. The fine tuning was carried out on the final 2, 6, 10, 15, 20 and 25 layers out of a total of 28 layers in MobileNet. The resulting accuracy of each fine-tuning test is shown below in figure 31 and summarised in table 1.

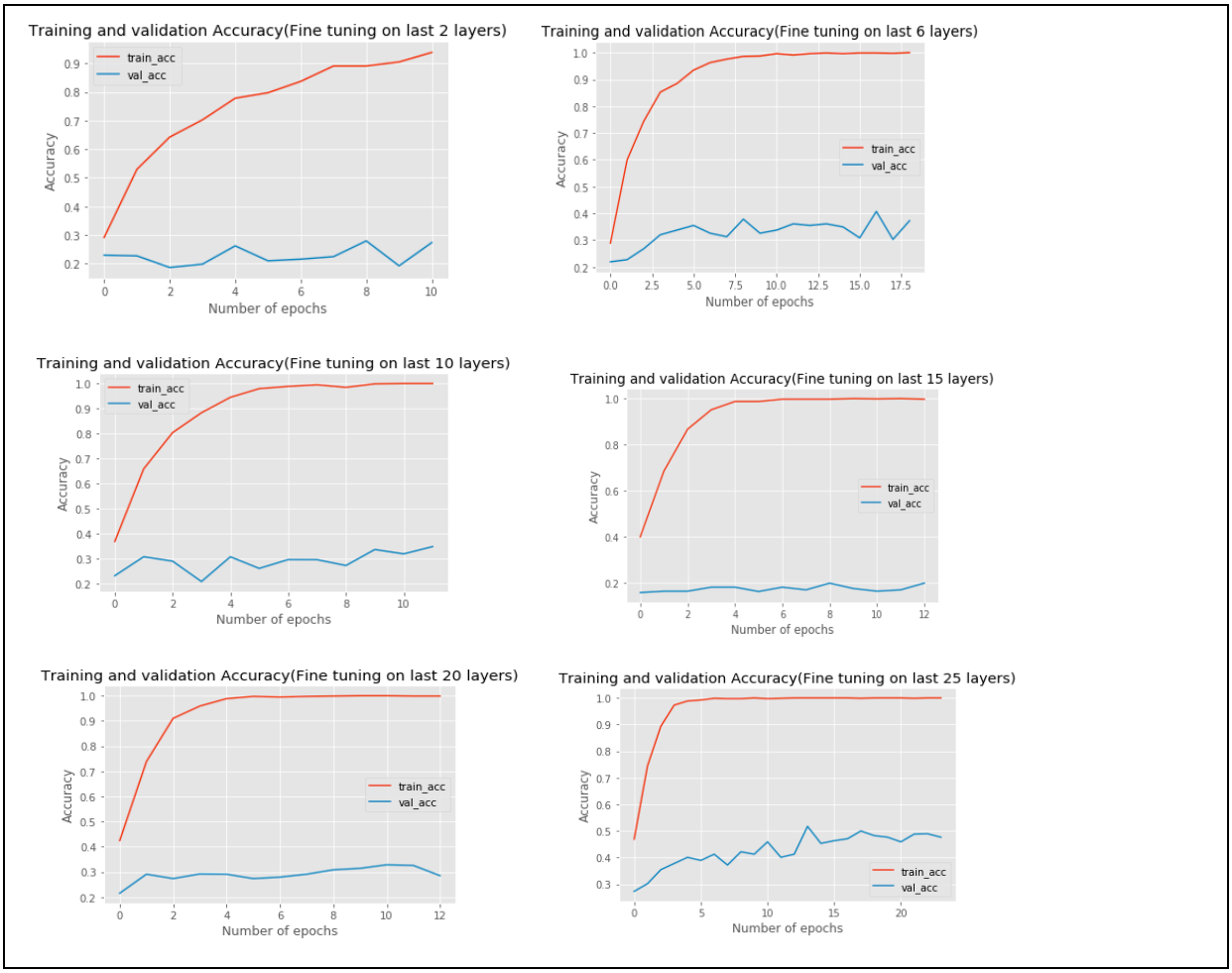


Figure 31: These graphs show the model performance when fine tuning on different layers of the subset of HAM10000.

Interestingly, there was no significant improvement from transfer learning until fine tuning on the last 25 layers, achieving a validation accuracy of 52%. If 25/28 layers of MobileNet require fine tuning, it further reinforces that the transferability of layers in pre-trained CNNs is drastically reduced when there is high difference between the original and new dataset. It seems that only the first three layers of MobileNet have features which are useful to this dataset. Leaving these three layers frozen and fine tuning the remainder of the model only results in a validation accuracy of 52%, which is still not sufficient for use in healthcare. It is also interesting to observe, in table 1, that increasing the number of layers that are fine-tuned does not always result in a higher validation accuracy. For example, when fine tuning on the final 2 layers the model achieved a validation accuracy of 27%. This jumped to 40% when fine tuning on the final 6 layers and dropped back to 35% when fine tuning on the final 10 layers. It would be expected that the more layers that are exposed to retraining, the more detail the model can learn about the new dataset.

The final test on the subset of data, fine tuning on all layers/ training the MobileNet architecture from scratch, achieved the most promising results, with 68% validation accuracy. This is not surprising, considering the difference between the datasets. We can see from below in figure 32 that there is still quite a prominent gap between training accuracy and validation accuracy, indicating the model is still overfitting. This is probably due to the small size of the data subset. A validation accuracy score of 68% is promising but is not high enough for use in a medical setting.

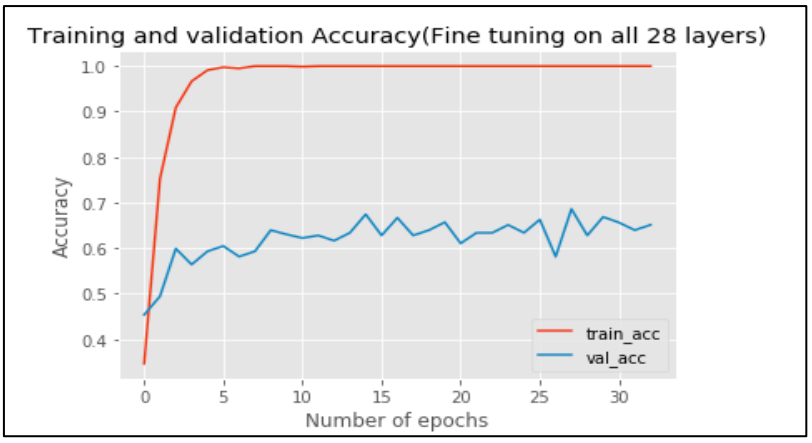


Figure 32: This graph shows the training and validation accuracy on MobileNet when fine tuning the model from scratch.

Method	Maximum Training Accuracy	Maximum Validation Accuracy
Transfer Learning	100%	25%
Fine Tuning (Last 2 layers)	93%	27%
Fine Tuning (Last 6 layers)	100%	40%
Fine Tuning (Last 10 layers)	100%	35%
Fine Tuning (Last 15 layers)	100%	20%
Fine Tuning (Last 20 layers)	100%	32%
Fine Tuning (Last 25 layers)	100%	52%
Fine Tuning Entire Architecture	100%	68%

Table 1: This table shows an overview of results of fine tuning and transfer learning on the subset of data from HAM10000.

5.3 Results of Altering Hyperparameters

As fine tuning on all 28 layers of MobileNet was deemed optimal for this task, several hyperparameters (as discussed in the methodology chapter) were altered in order to improve the validation accuracy before training the model on the entire dataset. The results of all alterations are shown below in table 2.

Hyperparameter settings	Validation Accuracy
Original model (lr=0.01, no momentum)	68%
Momentum (0.9)	72%
Learning rate (with momentum of 0.9)	
0.001	71%
0.0004	70%
0.0005	73%
0.0006	72%
0.0001	67%
Adam Optimizer	65%
Nesterov Momentum (0.9)	77%
Batch size increase (64)	62%
Dropout layer (0.5)	73%

Table 2: This table shows the results of different hyperparameter settings for the model MobileNet.

The unaltered model used an SGD optimizer with a default learning rate of 0.01. After adding momentum to the model, validation accuracy increased to 72%. Comparing figure 32 to below in figure 33, the gap between training and validation accuracy is reduced. There is also a general reduction in instability following the addition of momentum, with validation accuracy showing a steady increase, with the exception of one or two fluctuations. Training accuracy does not jump as rapidly to 100% with momentum as it did previously.

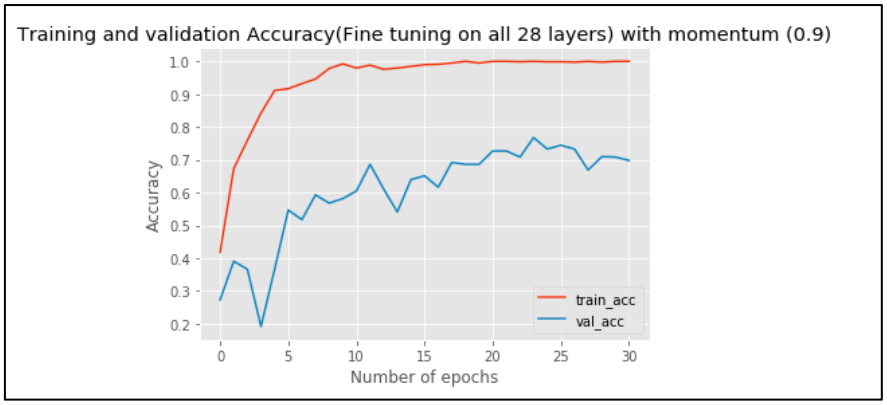


Figure 33: This graph shows the model performance of MobileNet with the addition of momentum.

Using a momentum value of 0.9 while altering the learning rate gave varying results. As stated in the literature review, learning rate has the strongest impact on model performance. From table 2, it is clear that a learning rate of 0.0005 yielded the highest validation accuracy at 73%. It is also observed that as the learning rate decreases from 0.01 to 0.0005, the validation accuracy increases. However, decreasing the learning rate below this (0.0005) to 0.0004 and 0.0001 leads to a decrease in validation accuracy. It is known from the literature review that when the learning rate is smaller, the adjustments to the weights are also smaller, which leads to a very stable model. However, there is a threshold after which decreasing the learning rate causes the model to overfit. It is clear from the results that the threshold here is 0.0005. Beyond this the validation accuracy decreases, a sign of an increase in overfitting.

Replacing the SGD optimiser with the Adam Optimizer led to a significant drop in performance with validation accuracy of 65%. This is surprising as, in the literature, it is stated that Adam typically improves model performance over SGD. The most significant improvement in model performance was seen from using Nesterov Momentum, achieving a validation accuracy of 77% (As seen in figure 34). It is clear from this result that Nesterov is an important addition to regular momentum.

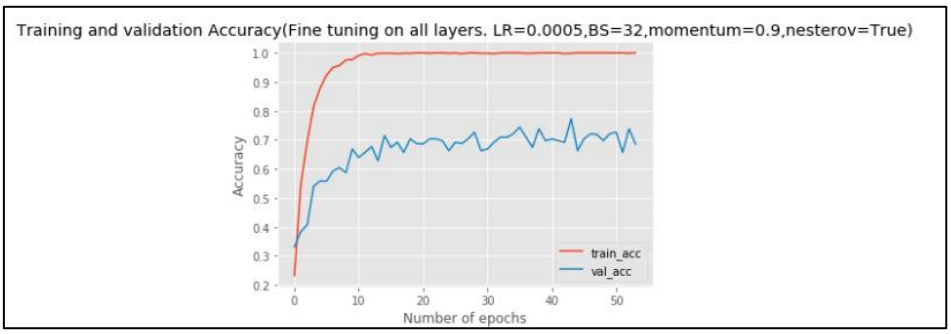


Figure 34: This graph shows the model performance of MobileNet when using Nesterov Momentum

Batch size was altered to explore if the Nesterov momentum model could be improved on. From table 2, it is observed that when batch size was increased from 32 to 64, validation accuracy dropped to 62%. This is not a surprising result as CPUs tend to favour smaller batch sizes. On a GPU system, these results would likely differ as these systems generally favour larger batch sizes. However, the increase in accuracy from the smaller batch size brings with it longer training times. The number of training epochs is just over 30 using a batch size of 64, whereas using a smaller batch size results in over 50 epochs. In a medical setting, high accuracy is critical and thus a smaller batch size, even with longer training times, is considered optimal.

As seen in table 2, the addition of a dropout layer before the global average pooling layer and SF layer, achieved a validation accuracy of 73%. As the models tested up to this point had a significant gap between training accuracy and validation accuracy, it is likely the models were overfitting. As dropout reduces overfitting by reducing coadaptation between layers, it is surprising that validation accuracy decreased slightly rather than increasing accuracy. Although it is suggested in the literature that a dropout rate of 0.5 is the optimal value for this hyperparameter, perhaps more values should have been tested. It is also possible that the dropout layer may be more successful at a different location in the model.

5.4 Results of Final Model Training on Entire dataset

Using the optimal hyperparameters discovered above (Nesterov momentum 0.9, learning rate 0.0005) three models (MobileNet, Xception and MobileNet-KNN) were trained on the entire dataset. The models were then evaluated for their accuracy on the test set. This was calculated using the simple sum:

(Number of correctly predicted lesions / Total number of predictions) * 100%

The results are shown below in table 3. Overall, a significant improvement can be seen in the performance of MobileNet when trained on more images. The validation accuracy increased from 77% on the subset of data, to 85% when trained on the entire dataset. This improvement shows the importance of acquiring a large labelled dataset before applying large networks such as CNNs. By comparing MobileNet and Xception, several observations can be made. Firstly, using early stopping, both models required a similar number of epochs to train.

Secondly, both models achieved high accuracies on the validation set and test set, with MobileNet achieving a slightly higher score of 85% on the test set. The most obvious difference between the performance of the two models is the time taken to train. MobileNet required 12.5 hours of training time, while Xception required considerably longer training time, 104.5 hours. It is a significant finding that a smaller model such as MobileNet, with fewer layers and smaller memory requirements, can achieve comparable (if not higher) accuracy than a large model like Xception. MobileNet, is therefore the optimal choice for model training in terms of time, memory and accuracy.

The results for the MobileNet model with a KNN classifier are disappointing, achieving very low validation accuracy of 39% and 14% test accuracy. The KNN classifier was not tested on the training set, as the feature extractor portion of the model was trained using this subset. It appears that KNN classifiers are not ideal in the case of multi-classification models. Perhaps they would be more suitable for distinguishing between only two classes such as MSC and NMSC.

Model	**No. of epochs**	**Time to train**	**Training Accuracy**	**Validation Accuracy**	**Test Accuracy**
MobileNet	23	12.5 hours	99%	85%	85%
Xception	24	104 hours	99%	82%	83.4%
MobileNet-KNN	N/A	N/A	N/A	39%	14%

Table 3: This table shows the results of final model training on the entire dataset.

5.5 Overview of class accuracy in MobileNet

Below shows the individual accuracy for each skin lesion class in the highest performing model, MobileNet. It is seen that the categories of BKL and MSC had the lowest accuracy score, with VASC and DF reaching very high scores.

Lesion	**Accuracy**
BKL	75%
NV	81%
MSC	77%
AKIEC	85%
VASC	100%
BCC	85%
DF	93%

Table 4: This table shows the accuracy score for each class in HAM10000 using the MobileNet model on the test set.

6 EVALUATION

6.1 Chapter Overview

At the beginning of this study three research questions were proposed, as reiterated below. This chapter revisits these research questions, evaluating what has been learned during this study.

6.2 Research Questions

1. Can an accurate multi-classification model be developed, which can distinguish between a number of different skin lesions, with high accuracy?

From this study it has been shown that a multi-classification model can be developed with high accuracy, encompassing a variety of skin conditions. At the beginning of the study it was hypothesised that by balancing the dataset, the resulting overall accuracy would not be as high as in the literature as these studies may have yielded unrealistically high accuracy scores. However, it was found that high accuracy could be achieved by artificially expanding the minority classes and removing images from the majority class. By using the pretrained CNNs, MobileNet and Xception, it was possible to develop sophisticated models capable of predicting the correct skin lesion class to within 85% and 83% respectively. The highest performing model was achieved by using the pre-trained architecture from the CNN MobileNet. These scores match, and even exceed, the high accuracy scores of many models in the literature.

2. Which method of transfer learning is optimal for this dermatology task?

For medical tasks, including this specific dermatology task, it appears that transfer learning will only yield high accuracy if the entire model is trained from scratch, fine-tuning on the entire architecture. The transfer learning method proved ineffective, achieving a validation accuracy score of 25%. When using fine tuning, improvements were only observed when 25/28 layers in MobileNet were fine tuned. This model achieved an accuracy score of 52% which is still underwhelming for use in healthcare. As medical data is strikingly different

from the ImageNet dataset, it appears transfer learning will only be successful in this instance if the entire architecture is trained from scratch. This allows the model to learn the unique features of dermoscopy images.

3. Will the number of layers in the pre-trained CNN affect accuracy?

The number of layers, or the depth, of a pretrained CNN does not appear to affect accuracy. The Xception model, consisting of 71 layers would be considered a deeper model than MobileNet, consisting of only 28 layers. Both models achieved similar accuracy scores on the unseen dataset, the test set. As the MobileNet model has smaller memory requirements and faster training times than Xception, it is considered the optimal model.

7 DISCUSSION

7.1 Chapter Overview

This chapter will discuss the findings of the study in more detail and how these findings have contributed to the existing knowledge in the literature. It will detail the problems faced in dermatology due to the rising rates of skin cancer and how AI and the developed model can be used to help these challenges. It will include an account of how this model is unique to the literature and how it overcomes a number of limitations in other dermatology based classification models.

7.2 The impact of the study on Dermatology

This study has contributed to the conversation surrounding the rising rates of skin cancer and how AI could be a useful tool in managing the associated demands on the healthcare system. The Irish Skin cancer society have defined the current state of skin cancer incidence rates in Ireland an "epidemic". Unfortunately, in Ireland and many other countries worldwide, this definition is appropriate. The WHO (2007) define an epidemic as "The occurrence in a community or region of cases of an illness, specific health-related behaviour or other health-related events clearly in excess of normal expectancy". The number of cases of skin cancer worldwide are certainly in excess of normal expectancy. In Ireland skin cancer occurrences are expected to rise by two thirds by 2040, increasing the financial burden of the disease on the country. If these rates will rise as rapidly as predicted, there will not be enough dermatologists to meet the demand. Further, there is already an insufficient number of medical professionals trained to use dermoscopy. Relying on the visual perception of a professional with lack of training in this area is dangerous, particularly considering the fatal consequences of an undiagnosed MSC lesion. This study explored diagnostic aids for dermoscopy images using machine learning. The most successful model developed during this study used the pre-trained CNN MobileNet. This model can classify seven different classes of skin lesions to an accuracy of 85%. This level of accuracy is certainly sufficient as a diagnostic aid or a second opinion. This model is an excellent example of the potential of AI in dermatology and healthcare in general. The proposed model has many potential uses in the medical sector including a diagnostic aid, a method of determining urgency of patients on waiting lists, and for use in a skin care application.

7.3 A comparison of this model with previous skin lesion classification models

Although this model is not the first attempt in the literature to classify skin lesions, it is unique in a number of ways. Firstly, the multi classification model can classify seven different classes of skin lesions. The focus of much research in this area is on binary classification (Esfahani et al., 2016; Esteva et al., 2017; Shoieb et al. ,2016; Pomponiu et al., 2016). There is a wide diversity of skin lesions which a patient could have. A model which assumes a lesion is either MSC or NMSC cannot be integrated into healthcare. The multi-classification model developed in this study accounts for many classes of benign and precancerous lesions, yielding a more specific and accurate diagnosis and reducing the chance of unnecessary surgery. Secondly, the model has been trained on dermoscopy images. This is significant because in the literature, the majority of CNN models are trained on clinical images taken using an ordinary camera (Esfahani et al., 2016; Esteva et al., 2017; Shoieb et al., 2016; Pomponiu et al., 2016). As dermoscopy is the gold standard approach in diagnosing skin cancer in healthcare, a ML model developed for use in this area must be trained using these images.

Further, the model developed in this study has made significant improvements on the two existing models in the literature which use dermoscopy images. The first is by Harangi (2018) where an ensemble of CNNs were used to classify skin lesions to 86% accuracy. However, this model is limited to the three classes it was trained on: MSC, NMSC and Nevi. Other benign lesions such as dermatofibroma are not accounted for in this study. The model in this study yields comparable accuracy while accounting for seven different classes of skin lesions. Another multi-classification model in the literature was developed by Kawahara et al. (2018). This model used the pretrained CNN AlexNet as a feature extractor and a logistic regression classifier to classify 10 different classes of skin lesions. Although it achieved an average accuracy of 82%, the model was subjected to an unbalanced dataset. The classes with limited images only achieved an accuracy of 60%, showing the overall accuracy of a model cannot account for the overall performance of a model when class imbalance is not addressed. The MobileNet model in this study is the first time class imbalance has been addressed in a multi-classification model using dermoscopy images. By artificially expanding the dataset, the model achieves comparable overall accuracy, 85%, to the three-class model of Harangi (2018), and higher accuracy than Kawahara et al. (2016). The originally under

represented classes of AK, VAS, and DF achieved accuracies of 85%, 100% and 93% respectively. This study portrays the importance of balancing data in any ML task to prevent a model from directing its attention to majority classes. It is disappointing, however, that melanoma achieved one of the lowest overall accuracies in the model at 77%. The model is also the first in this area to use the pre-trained CNNs MobileNet and Xception which proved comparable to other popular models such as VGG Net and AlexNet.

7.4 Pretrained CNNs and Transfer Learning

This study is also the first in this area to include a comparison of the different methods of transfer learning. The results show that simply using the pre-trained network as a feature extractor is inefficient for this task, achieving a validation accuracy of 25%. It is clear that when the content of a dataset is sufficiently different from the data the pretrained CNN was trained on this method is ineffective. The study compares the fine tuning method on a number of different layers. As discussed in the literature review, pre-trained CNNs may only be useful for learning general features, not distinct medical features (Long et al., 2015). As the features in a pretrained network are thought to change from general to specific at some arbitrary point in the pre-trained architecture, one would expect some success for fine tuning to extract general features in the dataset such as borders and colours. However, in this study, improvements were not visible until almost all layers (25/28) of MobileNet were fine-tuned (52% validation accuracy). These results coincide with the recommendations of Yigit et al. (2018) that if a dataset is large and the contents are very different to the data the model was trained on, it is better to train a new model from scratch or fine tune the entire architecture of a pretrained CNN. It is likely that this is the reality in most medical tasks using pre-trained CNNs. However, in a similar study by Kieffer et al. (2017), investigating the different methods of TL for histopathology images (tissue samples), this is not the case. Kieffer et al. (2017) compared three methods of using CNNs for classifying medical tissue samples. They achieved an accuracy of 50% using transfer learning, much higher than the score in this study. This increased to 57% when fine tuning but reduced to 42% when training a model from scratch. It appears that as a general rule, if the dataset content is extremely different to ImageNet, fine tuning the entire architecture from scratch or building a new model is the most appropriate method. However, as every task is individual, this may not always be the

case and it is worth investigating the different methods to ensure highest model performance. It is generally accepted in the literature that increasing the depth (number of layers), in a model leads to a higher model performance. However, in this study it was found that models with less depth such as MobileNet, can achieve similar or higher accuracies than similar models with more layers such as Xception. This is promising in cases where there is a lack of computational power. A small model with comparable accuracy will speed up training time without compromising performance.

7.5 The proposed Flask application

This study is the first in this area to show the potential of the proposed CNN, integrating the model into an application. This application, although basic, shows the potential of the model in a real life setting. Although useful for patients, giving them advice on when to see a doctor, it is most useful for medical setting as the model has been trained on the fine detail of dermoscopy images. The small size of MobileNet makes it perfect for a medical mobile application. Mobile Dermatoscopes are becoming more readily available, this attached to the phone of a GP with the downloaded application could be a powerful tool integrating health, AI and technology. As 60% of patients in the study by Steeb et al. (2019) would be interested in benefiting from these apps if recommended by a medical doctor, it would require encouragement and promotion from individuals in healthcare.

8 CONCLUSIONS

8.1 Chapter overview

This chapter will include the concluding remarks relating to the research questions. It will also include an overview of the strengths and limitations of the study and recommendations for further work.

8.2 Concluding remarks

This work has successfully contributed to research regarding the integration of AI into dermatology to aid diagnosis of dermoscopy images. During this study, an accurate multi-classification model was developed which can distinguish between 7 classes of skin lesions with 85% accuracy. It is the first CNN model in this area which is both trained on a large dataset comprising of dermoscopy images, accounts for class imbalance and uses the pretrained CNN MobileNet. This resulted in a well-rounded model with high accuracy in all classes, not just the majority classes as in the model by Kawahara et al. (2016). The study has determined that in a dermatology classification task, the method of transfer learning most suitable is fine tuning over the entire architecture. Finally, it has been found that a compact network such as MobileNet has comparable accuracy to deep networks such as Xception used in this study but also comparable accuracy to more popular CNNs such as VGG Net in this area. The resulting proof of concept Flask application is evidence of the real life potential of this model in healthcare. For AI in healthcare to become commonplace it will require the collaboration of medical professionals, computer/data scientists and the government.

8.3 Strengths and Limitations of the study

8.3.1 Strengths

8.3.1.1 Availability of a large dataset

The methodology of this study has many strong points. Firstly, the availability of a large dataset containing 10,000 dermatoscopic images will lead to better model performance and reduce overfitting. The images have also been labelled by medical professionals which removes any uncertainty about the image classes. In the literature, there is often a problem with a lack of labelled dermatoscopic images to train a model. This dataset is also a strong choice as it contains seven classes of skin lesions, making it ideal for developing a multi-class model.

8.3.1.2 Lack of preprocessing yields realistic results

The images are not subjected to preprocessing techniques as often appears in the literature. This is important as in real life scenarios, images will not be preprocessed. Hairs and background noise are the norm in these images, so by keeping them in their original form the results are more realistic.

8.3.1.3 A strong choice of model

CNNs are a suitable choice of model as they are successfully used in many image processing tasks. They are particularly useful in situations where expert subject knowledge is required to create a feature extractor. In this study the researcher does not have a medical background and so manually extracting features is likely to introduce errors. As CNNs automatically extract features without any prior subject knowledge, it is a strong choice of model. Using pretrained models is particularly suitable as it saves time building a network from scratch.

8.3.1.4 A reproducible study

This methodology is useful as it explores and documents how the choice of hyperparameters were decided. This is something that is often missing from the literature, however, it is

important for the reproducibility of the study. It also explores the various methods of transfer learning in order to make an informed decision on choice of method.

8.3.1.5 Demonstration of a real-life application

This study goes further than most studies in this area as it demonstrates how the final model could be used in a real-life application, showing the potential of the model for use in healthcare or a mobile application.

8.3.2 Limitations

8.3.2.1 Class imbalance

Weaknesses of the study include the imbalance in the dataset. This weakness is overcome by both oversampling using data augmentation techniques for the minority classes and undersampling on the majority class. Although the resulting dataset is balanced and therefore less likely to be biased, useful information from the majority class will be lost as many of these images were removed from the dataset. Although the rotated and flipped images are technically regarded as new images, they will still share many features of the original image such as colours. A more accurate model could be developed if all images were original. However, balancing the data was more important to avoid the model giving preference to the majority class and yielding unrealistically high accuracies.

8.3.2.2 Lack of computational power

The lack of access to a GPU system is a significant disadvantage to this study for many reasons. In terms of choosing a pretrained model, deeper models are not appropriate due to high memory and time demands. As deeper models generally lead to higher accuracies, perhaps a higher performing model could have been developed. However, significant research went into choosing the most suitable pre-trained CNN for use with less computational power. Xception and MobileNet were purposely designed for such situations and have comparable accuracy with other deeper models such as VGG Net. The lack of GPU and time constraints in the study also limits the number of tests that can be run on the full image set. Xception took 104 hours to train on the entire dataset. MobileNet took 12 hours.

Although MobileNet took significantly less time to train, 12 hours at a time to test numerous different hyperparameter settings is not ideal. This disadvantage is overcome by the decision to use a small subset of images and the smaller model, MobileNet, to find the optimal hyperparameters. These tests give a good indication of how the hyperparameters would perform on the entire dataset.

8.4 Further work and recommendations

To further improve the proposed model, I recommend adding more classes to expand the scope of diagnosis. This will require many professionally labelled dermoscopy images which will involve the willingness of patients to give permission for their medical images to be used. It will also demand the cooperation of healthcare professionals and researchers to take time to label and pool together these dermoscopy images. If such applications are to be used in the medical sector in the future, I propose a pilot study to first be carried out in a small number of medical clinics to determine the success of such an application. If waiting lists are handled more efficiently, diagnoses are made faster, money is saved and there is a reduction in the number of false alarms it would provide enough evidence to introduce the application on a larger scale.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean J., and Devin, M. (2016). TensorFlow: A system for Large-Scale Machine Learning. *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation*. Savannah, G.A, USA. November 2-4, 2016. 265-283.

Agarap, A.F.M. (2017). An Architecture combining Convolutional Neural Network (CNN) and Support Vector Machines (SVM) for Image Classification. *ArXiv*, abs/1712.03541

Ali, A. and Marzouqi, H. (2017). Melanoma Detection Using Regular Convolutional Neural Networks. *2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA).*

Alpaydin, E. (2009). Introduction to Machine Learning. 2nd Edition. The MIT Press. Cambridge, Massachusetts.

Barr, A. and Feigenbaun, E.A. (2014). The Handbook of Artificial Intelligence. Volume 2. William Kauffman, Inc. Los Altos, California.

Basheer, I.A. and Hajmeer, M. (2000). Artificial neural networks: fundamentals, computing, design and application. *Journal of Microbiological Methods* **43** (2000), 3-31.

Boukamp, P. (2005). Non-melanoma skin cancer: What drives tumor development and progression? *Carcinogenesis* **26**(10), 1657-1667.

Boukladida, H., Hassen, N. and Gafsi, Z. (2011). A highly time efficient digital multiplier based on the A2 binary representation. *International Journal of Engineering Science and Technology*. **3**(5), 4498-4509.

Bre, F., Gimenez, J.M. and Fachinott, V.D. (2017). Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks. Elsevier.

Buda, M., Maki, A., and Mazurowski, M.A. (2017). A systematic study of the class imbalance problem in convolutional neural networks. ArXiv: 1710.053811.

Campbell, N.A., Reece, J.B., Urry, L.A., Cain, M.L., Wasserman, S.A., Minorky, P.V. and Jackson, R.B. (2014). Biology A Global Approach, 10th Edition. Pearson Education Limited, Harlow.

Celebi, M.E., Kingravi, H.A., Uddin, B., Iyatomi, H., Aslandogan, Y.A., Stoecker, W.V. and Moss, R.H. (2007). *Computerized Medical Imaging and Graphics* **31**(2007), 362-373.

Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21-26 July 2017.

Chu, B., Madhavan, V., Beijom, O., Hoffman, J. and Darrell, T. (2016). Best Practices for Fine-Tuning Visual Classifiers to New Domains. *Computer Vision- ECCV 2016 Workshops. Lecture Notes in Computer Science* 9915.

Cieresan, D.C., Meier, U., Masci, J., Gambaradella, L.M. and Schmidhuber, J. (2011). Flexible High Performance Convolutional Neural Networks for Image Classification. *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence* 1237-1242.

Creswell, J.W. (2003). Research Design: Qualatative, Quantatative, and Mixed Methods Approaches, 2nd edition. Sage Publications, USA.

Dai, J., Li, Y., He, K. and Sun, J. (2016). R-FCN: Object Detection via Region-based Fully Convolutional Networks. *30th Conference on Neural Information Processing Systems (NIPS 2016)*, Barcelona, Spain, 1-9.

Dalia, F., Zohra, A., Reda, K. and Hocine, C. (2017). Segmentation and classification of melanoma and benign skin lesions. *Optik* **140**(2017) 749-761.

Esfahani, E.N., Samavi, S., Karimi, N., Soroushmehr, S.M.R., Jafari, M.H., Ward, K. and Najarian K. (2016). *2016 38th International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC).* Orlando, FL, USA, 16-20 August 2016. 1372-1376.

Esteva, A., Kuprel, B., Novoa, R.A., Ko, J., Swetter, S.M., Blau, H.M. and Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **452**, 115-118.

Harangi, B. (2018). Skin lesion classification with ensembles of deep convolutional neural networks. Journal of Biomedical Informatics **86**(2018), 25-32.

Haykin, S. (1994). Neural Networks: A Comprehensive Foundation. Macmillan, London.

Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I. and Salahhutdinov, R.R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *ArXiv* 2012, 1-18.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyland, T., Andreetto, M. and Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv: 1704.04861

International Agency For Research on Cancer. 2019. Cancer Today. [ONLINE] Available at: https://gco.iarc.fr/today/online-analysis-multi-bars?v=2018&mode=cancer&mode_population=countries&population=900&populations=900&key=asr&sex=0&cancer=39&type=0&statistic=5&prevalence=0&population_group=0&ages_group%5B%5D=0&ages_group%5B%5D=17&nb_items=10&group_cancer=1&include_nmsc=1&include_nmsc_other=1&type_multiple=%257B%2522inc%2522%253Atrue%252C%2522mort%2522%253Afalse%252C%2522prev%2522%253Afalse%257D&orientation=horizontal&type_sort=0&type_nb_items=%257B%2522top%2522%253Atrue%252C%2522bottom%2522%253Afalse%257D&population_group_globocan_id=. [Accessed 27 July 2019].

Irish Cancer Society (ICS) (2018). About Skin Cancer. [ONLINE]. Available at: https://www.cancer.ie/reduce-your-risk/sunsmart/skin-cancer#sthash.7371ef4F.dpbs [Accessed 26 July 2019].

Irish Skin Foundation. 2019. Skin Cancer. [ONLINE] Available at: https://irishskin.ie/melanoma-skin-cancer/. [Accessed 25 July 2019].

Islam, M.T., Rahman, S., Siddique, B.M. and Jabid, T. (2018). Food Image Classification with Convolutional Neural Network. ICIIBMS 2018, Track 2: Artificial Intelligence, Robotics, and Human-Computer Interaction, Bangkok, Thailand. 257-262.

Jain, A.K., Duin, R.P.W and Mao, J. (2000). Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(1), 4-37.

Kaggle. 2018. Skin Cancer MNIST: HAM10000. [ONLINE] Available at: https://www.kaggle.com/kmader/skin-cancer-mnist-ham10000. [Accessed 27 July 2019].

Kaufman, G.J. and Breeding, K.J. (1976). The automatic recognition of human faces from profile silhouettes. *IEEE Transactions on Systems, Man and Cybernetics* **6**(2), 113-121.

Kawahara, J., BenTaieb, A. and Hamarneh, G. (2016). Deep features to classify skin lesions. *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI).* Prague, Czech Republic, 13-16 April 2016.

KD Nuggets. 2017. Understanding Deep Convolutional Neural Networks with a practical use-case in Tensorflow and Keras. [ONLINE] Available at: https://www.kdnuggets.com/2017/11/understanding-deep-convolutional-neural-networks-tensorflow-keras.html. [Accessed 27 July 2019].

Keckler, S.W., Dally, W.J., Khailany, B., Garland, M. and Glasco, D. (2011). GPUS and the Future of Parallel Computing. *IEEE Micro* **31**(5), 7-17.

Keras Documentation. 2019. Applications. [ONLINE] Available at: https://keras.io/applications/. [Accessed 27 July 2019].

Kieffer, B., Babaie, M., Kalra, S. and Tizhoosh, H.R. (2017). Convolutional Neural Networks for Histopathology Image Classification: Training vs. Using Pre-Trained Networks. 2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA).

Kingma, D.P. and Bam J. (2014). Adam: A method for stochastic optimisation. Proceedings of the 3rd International Conference of Learning Representations (ICLR) 2014.

Krizhevsy, A., Sutskever, I. and Hinton, G.E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing systems*, 1-9.

Lau, H.T. and Al-Jumaily, A. (2009). Automatically Early Detection of Skin Cancer: study based on Neural Network Classification. *2009 International Conference of Soft Computing and Pattern Recognition*, 375-380.

LeCun, Y., Bengio, Y. and Hinton, G. (2015). Deep Learning. *Nature* **521**, 437-444.

LeCun, Y., Bottou, L., Bengio, Y. and Hoffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *Proceedings of IEEE*, November 1998, 1-46.

LeCun, Y., Bottou, L., Orr, B.G. and Muller, K.R. (1998). *Lecture notes in Computer Science*, **7700**.

Liao, H. (2018). A Deep Learning Approach to Universal Skin Disease Classification.

Lin, M., Chen, Q., and Yan, S. (2014). Network In Network. ArXiv:1312.4400v3.

Long, J., Shelhamer, E. and Darrell, T. (2014). Fully Convolutional Networks for Semantic Segmentation. The IEEE Conference on Computer Vision and Pattern Recognition, 3431-3440.

Lopez, A.D., Giro-i-Nieto, X., Burdick, J. and Marques, O. (2017). Skin lesion classification from dermoscopic images using deep learning techniques. *Proceedings of the IASTED International Conference* Innsbruck, Austria. February 20-21, 2017,49-54.

Magliogiannis, I. and Doukas, C.N. (2009). Overview of Advanced Computer Vision Systems for Skin Lesions Characterisation. *IEEE Transactions on Information Technology in Biomedicine* **13**(5), 721-733.

Medium. 2017. CNN Architectures: LeNet, AlexNet, VGG, GoogLeNet, ResNet and more. [ONLINE] Available at: https://medium.com/@sidereal/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5. [Accessed 27 July 2019].

Medium: Towards Data Science. 2018. Review: Xception — With Depthwise Separable Convolution, Better Than Inception-v3 (Image Classification). [ONLINE] Available at: https://towardsdatascience.com/review-xception-with-depthwise-separable-convolution-better-than-inception-v3-image-dc967dd42568. [Accessed 27 July 2019].

Medium: Towards Data Science. 2019. A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. [ONLINE] Available at: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. [Accessed 27 July 2019].

Milletari, F., Navab, N. and Ahmadi, S. (2016). V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. ArXiv: 1606.04797.

Mishkin, D. , Sergievskiy, N. and Malas, J. (2016). Systematic evaluation of CNN advances on the ImageNet. arXiv:1606.0228.

Narayanan, D.L., Saladi, R.N. and Fox, J.L. (2010). *Ultraviolet radiation and skin cancer. International Journal of Dermatology* **49**, 978-986.

Negnevitsky, M. (2005). Artificial Intelligence: A Guide to Intelligent Systems. 2nd Edition. Pearson Education Limited, Harlow.

Nilsson, N.J. (2014). Principles of Artificial Intelligence. Morgan Kaufmann Publishers, Inc.

Oliver, P. (2010). The students guide to research ethics. 2nd Edition. Open University Press, England.

Owens, J.D., Houstan, M., Luebke, D., Green S., Stone, J.E. and Phillips, J.C. (2008). GPU Computing. *Proceedings of the IEEE* **96**(5), 879-899.

Pandas documentation. 2019. pandas Dataframe. [ONLINE] Available at: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html. [Accessed 27 July 2019].

Peterson, L.E. (2009). K-nearest neighbour. *Scholarpedia*, **4**(2): 1883.

Pomponiu, V., Nejati, H. and Cheung, N.M. (2016). *2016 IEEE International Conference on Image Processing (ICIP)*. Phoenix, AZ, USA. 25-28 September 2016, 2623-2627.

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

Ruder, S. (2016). An overview of gradient descent optimisation algorithm. ArXiv: 1609.04747

Sarle, W.S. (1994) Neural Networks and Statistical Models. *Proceedings of the Nineteenth Annual SAS Users Group International Conference*, April 1994. 1-13.

Saunders, M. and Tosey, P. (2012). The Layers of Research Design. Rapport (Winter), 58-59.

Saunders, M., Lewis, P. and Thronhill, A. (1997). Research Methods for Business students 5th Edition. Pearson Education Limited, Harlow.

Sheha, M.A., Mabrouk, M.S. and Sharawy, A. (2012). Automatic Detection of Melanoma Skin Cancer using Texture Analysis. *International Journal of Computer Applications* **42**(20), 22-26.

Shoieb, D.A., Youssef, S.M. and Aly, W.M. (2016). Computer Aided Model for Skin Diagnosis Using Deep Learning. *Journal of Image and Graphics* **4**(2), 122-129.

Skin Cancer Foundation. 2019. Skin Cancer: Facts and Statistics. [ONLINE] Available at: https://www.skincancer.org/skin-cancer-information/skin-cancer-facts. [Accessed 27 July 2019].

Smith, L.N. (2018). A disciplined approach to neural network hyper-parameters: Part 1—learning rate, batch size, momentum and weight decay. ArXiv: 1803.09820.

Soehnge H., Ouhtit, A. and Anathaswamy, H.N. (1997). Mechanisms of Induction of Skin Cancer by UV Radiation. *Frontiers in Bioscience* **2**, 538-551.

Spector, L. (2006). Evolution of Artificial Intelligence. *Artificial Intelligence* **170**(2006), 1251-1253.

Steeb, T., Wessely, A., Mastnik, S., Brinker, T.J., French, L.E., Niesert, A.C. and Berking, C. Patient Attitudes and Their Awareness Towards Skin Cancer Related Apps: Cross-Sectional Survey. *JMIR Mhealth Uhealth 2019*, **7**(7).

Szegedy, C., Vanhouke, V., Ioffe, S. and Shiens, J. (2016). Rethinking the Inception Architecture for Computer Vision. The IEEE Conference on Computer Vision and Pattern Recognition, 2016, 2818-2826.

Tang, Y. (2013). Deep Learning Using Linear Support Vector Machines. ArXiv: 1306.0239.

Tempere University. 2019. Neural Network Basics. [ONLINE] Available at: http://www.uta.fi/sis/tie/neuro/index/Neurocomputing2.pdf. [Accessed 27 July 2019].

VanderPlas, J. (2016). Python Data Science Handbook. O'Reilly Media Inc.

Webb, R.A. and Copsey, K.D. (2011). Statistical Pattern Recognition, 3rd Edition, Wiley, UK.

Wiens, J. and Shenoy, E.S. (2017). Machine Learning for Healthcare: On the Verge of a Major Shift in Healthcare Epidemiology. *Clinical Infectious Diseases 2018*, **66**(1), 149-153.

Witten, I.H., Frank, E., Hall, M.A. and Pal, C.J. (2017). DATA MINING: Practical Machine Learning Tools and Techniques, 4th Edition. Elsevier.

Wolner, Z.J., Yelamos, O., Liopyrisk, K., Rogers, T., Marchetti, M., and Marghoob, A. (2017). Enhancing Skin Cancer Diagnosis with Dermoscopy. *Dermatol Clin*. **35**(4), 417-437.

World Health Organisation. 2007. Definitions: emergencies. [ONLINE] Available at: https://www.who.int/hac/about/definitions/en/. [Accessed 27 July 2019].

World Health Organisation. 2019. Skin Cancers. [ONLINE] Available at: https://www.who.int/uv/faq/skincancer/en/index1.html. [Accessed 25 July 2019].

Yigit, G.O. and Ozyildirim, B.M. (2018). Comparison of Convolutional Neural Network Models for food image classification. *Journal of Information and Telecommunication* **2**(3), 347-357.

yosuke.saito. 2017. Easy Implementation of Convolution and Pooling Layer in Deep Learning. [ONLINE] Available at: https://saitoxu.io/2017/01/01/convolution-and-pooling.html. [Accessed 27 July 2019].

Yuan, X., Yang, Z. , Zourdakis, G. and Mullani, N. (2006). SVM-based Texture Classification and Application to Early Melanoma Detection. *Proceedings of the 28th IEEE EMBS Annual International Conference* New York City, USA, Aug 30- Sept 3, 2006. 4775-4778.

Zeiler, M.D. and Fergus, R. (2013). Visualizing and Understanding Convolutional Networks. *Computer Vision-ECCV 2014. Lecture Notes in Computer Science* **8689**, 818-833.

Zhang, L. M. (2018). Effective, Fast and Memory Efficient Compressed Multi-Function Convolutional Neural Networks for More Acute Medical Image Classification. *Machine Learning for Health (ML4H). Workshop at NeurIPS 2018*. 1-6.

APPENDICES

APPENDIX A: SAMPLE FROM HAM10000 CSV FILE

	A	B	C	D	E	F	G
1	lesion_id	image_id	dx	dx_type	age	sex	localization
2	HAM_0000118	ISIC_0027419	bkl	histo	80	male	scalp
3	HAM_0000118	ISIC_0025030	bkl	histo	80	male	scalp
4	HAM_0002730	ISIC_0026769	bkl	histo	80	male	scalp
5	HAM_0002730	ISIC_0025661	bkl	histo	80	male	scalp
6	HAM_0001466	ISIC_0031633	bkl	histo	75	male	ear
7	HAM_0001466	ISIC_0027850	bkl	histo	75	male	ear
8	HAM_0002761	ISIC_0029176	bkl	histo	60	male	face
9	HAM_0002761	ISIC_0029068	bkl	histo	60	male	face
10	HAM_0005132	ISIC_0025837	bkl	histo	70	female	back
11	HAM_0005132	ISIC_0025209	bkl	histo	70	female	back
12	HAM_0001396	ISIC_0025276	bkl	histo	55	female	trunk
13	HAM_0004234	ISIC_0029396	bkl	histo	85	female	chest
14	HAM_0004234	ISIC_0025984	bkl	histo	85	female	chest
15	HAM_0001949	ISIC_0025767	bkl	histo	70	male	trunk
16	HAM_0001949	ISIC_0032417	bkl	histo	70	male	trunk
17	HAM_0007207	ISIC_0031326	bkl	histo	65	male	back
18	HAM_0001601	ISIC_0025915	bkl	histo	75	male	upper extremity
19	HAM_0001601	ISIC_0031029	bkl	histo	75	male	upper extremity
20	HAM_0007571	ISIC_0029836	bkl	histo	70	male	chest
21	HAM_0007571	ISIC_0032129	bkl	histo	70	male	chest
22	HAM_0006071	ISIC_0032343	bkl	histo	70	female	face
23	HAM_0003301	ISIC_0025033	bkl	histo	60	male	back
24	HAM_0003301	ISIC_0027310	bkl	histo	60	male	back
25	HAM_0004884	ISIC_0032128	bkl	histo	75	male	upper extremity
26	HAM_0004884	ISIC_0025937	bkl	histo	75	male	upper extremity
27	HAM_0002521	ISIC_0027828	bkl	histo	40	male	upper extremity
28	HAM_0002521	ISIC_0029291	bkl	histo	40	male	upper extremity
29	HAM_0006574	ISIC_0030698	bkl	histo	40	male	back
30	HAM_0006574	ISIC_0025567	bkl	histo	40	male	back
31	HAM_0001480	ISIC_0031753	bkl	histo	70	male	abdomen
32	HAM_0001480	ISIC_0026835	bkl	histo	70	male	abdomen
33	HAM_0005772	ISIC_0031159	bkl	histo	60	female	face
34	HAM_0005772	ISIC_0031017	bkl	histo	60	female	face
35	HAM_0005612	ISIC_0024981	bkl	histo	80	male	scalp
36	HAM_0005388	ISIC_0027815	bkl	histo	80	male	chest
37	HAM_0000351	ISIC_0024324	bkl	histo	85	male	back
38	HAM_0000351	ISIC_0029559	bkl	histo	85	male	back
39	HAM_0003847	ISIC_0030661	bkl	histo	85	male	upper extremity
40	HAM_0003847	ISIC_0027053	bkl	histo	85	male	upper extremity

APPPENDIX B: LESION DESCRIPTIONS FROM HAM10000

Melanocytic nevi

Melanocytic nevi (MN) is the medical term used for pigmented moles (British Association of Dermatologists, 2019). Melanocytic refers to the melanocytes that cause the dark colour pigment. A cluster of melanocytes form a mole/nevus. MN may take a variety of colour forms depending on skin tone (BAD, 2018). Moles present at birth are referred to as congenital melanocytic nevi. MN developed during childhood and adult life are referred to as acquired MN. MN lesions are benign but should be monitored for change as some moles can develop into melanoma. MN can be surgically removed, however they are usually left untreated due to the risk of scarring.

Actinic Keratoses

Actinic Keratoses (AK) and Bowen's disease (BD) are common precursors of SCC (Tshandl et al. 2018). Both are characterised by a scaly, non-pigmented lesion. However, AK is usually found on the face ,and BD on other body sites (Tshandl et al. 2018). These lesions are usually a result of long term exposure to UV rays from the sun or artificial sources of UV, such as tanning beds (Callen et al. 1997). BD can also develop as a result of a human papilloma virus infection (Tshandl et al. 2018). These diseases should be treated to avoid their transition to SCC.

Benign Keratosis

Benign keratosis (BK) is a benign skin growth, similar in appearance to a wart (American Academy of Dermatology,2019). They can range in colour from white to black and can be found on any part of the body. They are usually harmless and do not require treatment (American Academy of Dermatology, 2019).

Dermatofibroma

Dermatofibroma (DF) is an overgrowth of fibrous tissue of the dermis (British Association of Dermatologists, 2019). This growth is not life threatening and will not develop into skin cancer. Lesions appear as small firm bumps on the skin with colour ranging from pink to brown (BAD). Treatments include steroid injections and cryotherapy (freezing with liquid nitrogen) and in rare cases, excision of the lesion. DF is usually caused by an inflammatory reaction to a minor injury (BAD, 2018).

Vascular lesions

This class encompases a range of lesions including cherry angiomas and angiokeratomas. An angioma is a growth made up of small blood vessels (American Osteopathic college of dermatology, 2019). The cause of most angiomas is unknown. They usually do not require treatment unless they begin to bleed. They can be treated with cryotherapy or lasers (American Osteopathic college of dermatology, 2019).

APPENDIX C: EXAMPLES OF CODE USED

```
import numpy as np
import pandas as pd
filename= '/Users/lauradempsey/documents/ham.csv'
df= pd.read_csv(filename)    #Loading the HAM10000 CSV file in Pandas

path=[]

for image in df['image_id']:
    rootpath= '/Users/lauradempsey/documents/Dissertation/data/'
    imagetype='.jpg'
    truepath= rootpath +image +imagetype
    path.append(truepath)             #Adding the column 'path'


df['path']=path


#Example of Data Augmentation in the dataset
from PIL import Image


vasc=df[df['class']=='vasc']
numbers=range(142)
for i in numbers:
    img=Image.open(vasc.path.iloc[i])
    rootpath='/Volumes/memory/data/vasc/'    #Saving original images to a labelled fol
der
    img.save(rootpath+str(i)+'.jpg')

rot_vasc=[]
numbers=list(range(142))
for i in numbers:
    img=Image.open(vasc.path.iloc[i])                      #Rotating each image in the
 vascular class by 90
    rot_img=img.rotate(90,expand=True)                     #180, 270, verical and hori
zontal flips.
    rot_imga=img.rotate(180,expand=True)                  #Putting all new images in a
 list
    rot_imgb=img.rotate(270,expand=True)
    rot_imgc = img.transpose(Image.FLIP_LEFT_RIGHT)
    rot_imgd= img.transpose(Image.FLIP_TOP_BOTTOM)
    rot_vasc.append(rot_img)
    rot_vasc.append(rot_imga)
    rot_vasc.append(rot_imgb)
    rot_vasc.append(rot_imgc)
    rot_vasc.append(rot_imgd)

numbers=range(142,852)
id_vasc=[]
for i in numbers:
    id_vasc.append(i)    #Making list of numbers that will represent the new images

id_vasc=pd.Series(id_vasc)
rot_vasc=pd.Series(rot_vasc)
vasc_df=pd.DataFrame(data={'id':id_vasc,'image': rot_vasc})  #Putting numbers and corr
esponding images into pandas dataframe

numbers=range(710)
for i in numbers:
    img=vasc_df.image.iloc[i]
    name=vasc_df.id.iloc[i]
```

```
    rootpath='/Volumes/memory/data/vasc/'     #Saving all the new images to the labell
ed folder.
    img.save(rootpath+str(name)+'.jpg')


 #Downsampling Nevus class

from random import seed
from random import sample

seed(1)

numbers = [i for i in range(6705)]

nv_subset_nums = sample(numbers, 1000)


path=[]
rootpath='/Volumes/USB DISK/data/nv/'
for i in nv_subset_nums:          #Randomly selecting 1000 images
    x=rootpath+str(i)+'.jpg'
    path.append(x)


numbers=range(1000)
a=['nv']
labels=a*1000
len(labels)     #Creating 1000 nv labels from the dataframe


labels=pd.Series(labels)
images=pd.Series(path)
df=pd.DataFrame(data={'label':labels,'image_path':images})   ##Creating the pandas dat
frame

## Rest of classes added to the dataframe as shown in example below.

path=[]
rootpath='/Volumes/USB DISK/data/bkl/'
for i in range(1099):
    x=rootpath+str(i)+'.jpg'
    path.append(x)


 a=['bkl']
 labels=a*1099
 labels=pd.Series(labels)
 images=pd.Series(path)
 df3=pd.DataFrame(data={'label':labels,'image_path':images})

 #Adding all dataframes together. Now all images including augmented images have a l
abel and corresponding path to file.
 df=df.append(df2,ignore_index=True)
 df=df.append(df3,ignore_index=True)
 df=df.append(df4,ignore_index=True)
 df=df.append(df5,ignore_index=True)
 df=df.append(df6,ignore_index=True)
 df=df.append(df7,ignore_index=True)
 df


 ##TESTS ON SUBSET OF DATA

```

```
import tensorflow as tf
from tensorflow import keras as keras
from keras.preprocessing.image import ImageDataGenerator
import sklearn
from sklearn.model_selection import train_test_split
from keras.applications import MobileNet
from keras.applications.mobilenet import preprocess_input
from keras.layers import Dense,GlobalAveragePooling2D
from keras.models import Model
training,validation= train_test_split(df,test_size=0.85,random_state=42)
training,validation= train_test_split(training,test_size=0.20,random_state=42)    # Selecting only 1000 images for training and validation set.

train_datagen=ImageDataGenerator(preprocessing_function=preprocess_input)    #Initiating Keras Image Generator
validation_datagen=ImageDataGenerator(preprocessing_function=preprocess_input)


training_generate=train_datagen.flow_from_dataframe(training,directory=None,x_col='image_path',y_col='label',target_size=(224,224),batch_size=32,class_mode='categorical',shuffle=True)
validation_generate=validation_datagen.flow_from_dataframe(validation,directory=None,x_col='image_path',y_col='label',target_size=(224,224),batch_size=32,class_mode='categorical',shuffle=True)

#Generator flows the images from the dataframe, resizes and shuffles the images.


#Transfer Learning
model=MobileNet(weights='imagenet',include_top=False)
for layer in model.layers:
    layer.trainable=False   #No layers open to training

x=model.output
x=GlobalAveragePooling2D()(x)
preds=Dense(7,activation='softmax')(x)  #Seven classes in softmax function

model=Model(inputs=model.input,outputs=preds)

model.compile(optimizer='SGD',loss='categorical_crossentropy',metrics=['accuracy'])  #Compiling the model, specifying SGD as the optimiser, default learning rate and no momentum.

step_size_train=training_generate.n//training_generate.batch_size
step_size_val=validation_generate.n//validation_generate.batch_size


from keras.callbacks import EarlyStopping
es = [EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=10)] #Early stopping
X=model.fit_generator(generator=training_generate,steps_per_epoch=step_size_train,epochs=500,validation_data=validation_generate,validation_steps=step_size_val,callbacks=es) #Initiates model training, specifies the training and validation data.


#Fine Tuning
model=MobileNet(weights='imagenet',include_top=False)
for layer in model.layers:
    layer.trainable=True
for layer in model.layers[0:68]:   ##Specifies that from layer 0-68 no training will take place
        layer.trainable=False

x=model.output
x=GlobalAveragePooling2D()(x)
preds=Dense(7,activation='softmax')(x)
```

```
174.
175.  model=Model(inputs=model.input,outputs=preds)
176.
177.  #Rest of code same as above
178.
179.  #Training from scratch
180.  model=MobileNet(weights='imagenet',include_top=False)
181.  for layer in model.layers:
182.      layer.trainable=True #All layers open for training.
183.
184.  x=model.output
185.  x=GlobalAveragePooling2D()(x)
186.  preds=Dense(7,activation='softmax')(x)
187.
188.  model=Model(inputs=model.input,outputs=preds)
189.
190.  #To change optimiser settings:
191.  sgd = optimizers.SGD(lr=0.001,momentum=0.9,nesterov=True)
192.  model.compile(optimizer=sgd,loss='categorical_crossentropy',metrics=['accuracy'])
193.
194.  ##Training Full dataset: MobileNet Example
195.  training,test= train_test_split(df,test_size=0.10,random_state=42)
196.  training,validation= train_test_split(training,test_size=0.10,random_state=42)
197.
198.  train_datagen=ImageDataGenerator(preprocessing_function=preprocess_input)
199.  validation_datagen=ImageDataGenerator(preprocessing_function=preprocess_input)
200.  training_generate=train_datagen.flow_from_dataframe(training,directory=None,x_col='
    image_path',y_col='label',target_size=(224,224),batch_size=32,class_mode='categorical'
    ,shuffle=True)
201.  validation_generate=validation_datagen.flow_from_dataframe(validation,directory=Non
    e,x_col='image_path',y_col='label',target_size=(224,224),batch_size=32,class_mode='cat
    egorical',shuffle=True)
202.  from keras.callbacks import EarlyStopping
203.  es = [EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=10)]
204.
205.  model=MobileNet(weights='imagenet',include_top=False)
206.  for layer in model.layers:
207.      layer.trainable=True
208.
209.
210.  x=model.output
211.  x=GlobalAveragePooling2D()(x)
212.  preds=Dense(7,activation='softmax')(x)
213.
214.  model=Model(inputs=model.input,outputs=preds)
215.
216.  from keras import optimizers
217.  sgd = optimizers.SGD(lr=0.0005,momentum=0.9,nesterov=True)
218.
219.  step_size_train=training_generate.n//training_generate.batch_size
220.  step_size_val=validation_generate.n//validation_generate.batch_size
221.  model.compile(optimizer=sgd,loss='categorical_crossentropy',metrics=['accuracy'])
222.  L=model.fit_generator(generator=training_generate,steps_per_epoch=step_size_train,e
    pochs=500,validation_data=validation_generate,validation_steps=step_size_val,callbacks
    =es)
223.
224.  model.save("model.h5")   #Saving the model
225.
226.  #Making predictions with model
227.  test_datagen=ImageDataGenerator(preprocessing_function=preprocess_input)
228.  test_generate=test_datagen.flow_from_dataframe(test,directory=None,x_col='image_pat
    h',target_size=(224,224),batch_size=1,class_mode=None,shuffle=False)    #Generating th
    e test data
229.
230.  test_generate.reset()
```

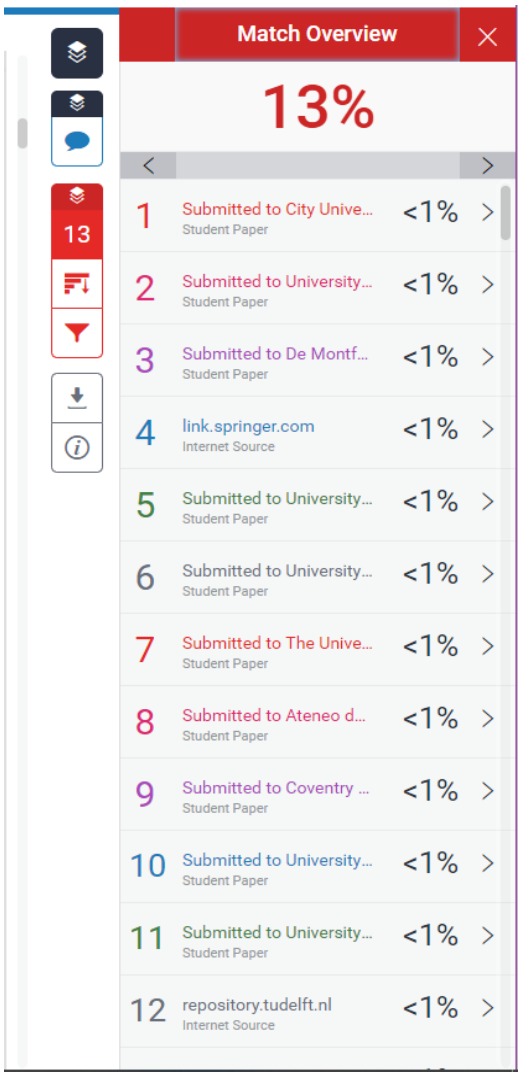
```
231.  predictions=model.predict_generator(test_generate,steps=len(test_generate.filenames
    ),verbose=1)
232.  predicted_class_indices=np.argmax(predictions,axis=1)  #Converting probabilities to
     prediction numbers
233.  labels = (training_generate.class_indices)  #associating a number with a label
234.  labels = dict((v,k) for k,v in labels.items())
235.  predictions = [labels[k] for k in predicted_class_indices] #Now have list of predic
    tions in terms of labels rather than numbers
236.  true_values=list(test['label'])
237.  predictions=pd.Series(predictions)
238.  true_values=pd.Series(true_values)
239.  Compare=pd.DataFrame(data={'predicted':predictions,'actual':true_values})  #Datafra
    me comparing the predictions to actual labels
240.  #Calculating accuracy
241.  ((len(Compare[Compare['predicted']==Compare['actual']]))/676)*100.
```

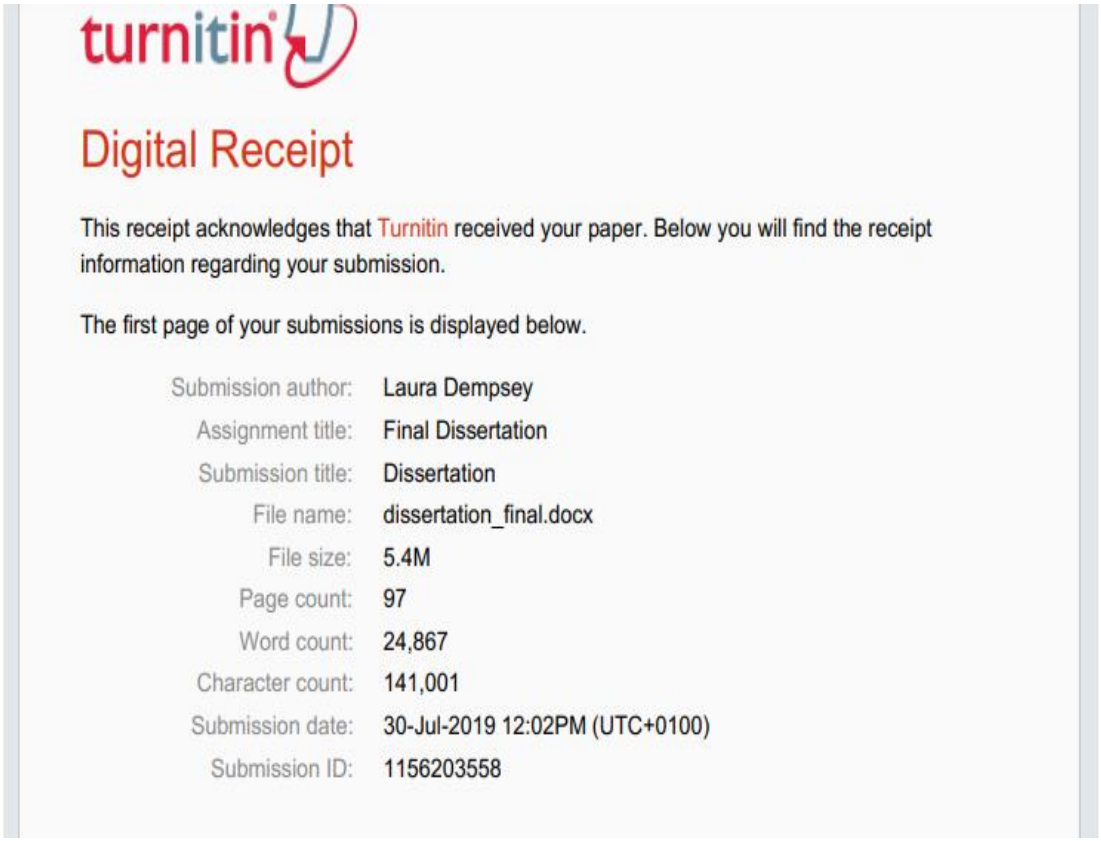
APPENDIX D: FLASK APP

```
from flask import Flask,render_template,request

from werkzeug.utils import secure_filename

from keras.models import load_model


from keras.preprocessing import image

from keras.applications.mobilenet import preprocess_input


import numpy as np


model=load_model("/Users/lauradempsey/model.h5")

model._make_predict_function()

labels={'akiec': 0, 'bcc': 1, 'bkl': 2, 'df': 3, 'mel': 4, 'nv': 5, 'vasc': 6}

labels = dict((v,k) for k,v in labels.items())


app= Flask(__name__)


@app.route("/")

def default():

    return render_template("homepage.html")


@app.route("/predictions")

def insetimage():

    return render_template("insertimage.html",message="")


@app.route("/predictions", methods=['POST'])

def form_handler():

    pic=request.files['image']

    pic.save(pic.filename)

    img_path='/Users/lauradempsey/Documents/flaskapp/' +pic.filename

    img = image.load_img(img_path, target_size=(224, 224))   ##need html to accept users image path and call it image path
```

```
    x = image.img_to_array(img)

    x = np.expand_dims(x, axis=0)

    x = preprocess_input(x)

    predicted_class_indices=np.argmax(model.predict(x),axis=1)

    guess=[labels[k] for k in predicted_class_indices]

    if guess==['mel']:

       message='This lesion may be melanoma. It is advised to see a doctor as soon as possible. For more   information please follow this link: https://www.skincancer.org/skin-cancer-information/melanoma'

    elif guess==['akiec']:

        message='This lesion looks like Actinic Keratosis. This is a potential precancer. It is advised to arrange an appointment with your GP. For more information see: https://www.skincancer.org/skin-cancer-information/actinic-keratosis'

    elif guess==['bcc']:

        message='This lesion may be Basal Cell Carcinoma. It is advised to see a doctor as soon as possible. For more information see: https://www.skincancer.org/skin-cancer-information/basal-cell-carcinoma'

    elif guess==['bkl']:

        message='This looks like a benign keratosis-like lesion, which is not life threatening. However, always consult with your GP.'

    elif guess==['df']:

        message='This looks like dermatofibroma. This is usually harmless and will not develop into cancer. Always consult with your GP for an absolute diagnosis. For further information see: http://www.bad.org.uk/ResourceListing_NonResponsive.aspx?sitesectionid=2&id=439&showmore=1&returnlink=http%3A%2F%2Fwww.bad.org.uk%2FResourceListing_NonResponsive.aspx%3Fsitesectionid%3D159%26sitesectiontitle%3DPatient%2BInformation%2BLeaflets%2B(PILs)%26originalpath%3Dfor-the-public%252fpatient-information-leaflets#.XSG41C8ZNp8'

    elif guess==['nv']:

        message='This lesion may be a melanocytic nevus (pigmented mole). These are very common but have the potential to develop into melanoma. Always consult with your GP for an absolute diagnosis. For further information see:https://www.bad.org.uk/shared/get-file.ashx?id=100&itemtype=document'

    elif guess==['vasc']:

        message='This looks like a vascuclar lesion. Please contact your GP for more information and an absolute diagnosis. For more info see: https://www.aafp.org/afp/1998/0215/p765.html'

    return render_template("insertimage.html",message=message)



if __name__=="__main__":

    app.run(debug==True)
```

APPENDIX E: TURNITIN MATCH REPORT AND DIGITAL RECEIPT





turnitin

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	Laura Dempsey
Assignment title:	Final Dissertation
Submission title:	Dissertation
File name:	dissertation_final.docx
File size:	5.4M
Page count:	97
Word count:	24,867
Character count:	141,001
Submission date:	30-Jul-2019 12:02PM (UTC+0100)
Submission ID:	1156203558

LIFELONG LEARNING CENTRE

INSTITUTE of TECHNOLOGY CARLOW

Work submitted for assessment which does not include this declaration will not be assessed.

DECLARATION

*I declare that all material in this submission e.g. thesis/essay/project/assignment is entirely my/our own work except where duly acknowledged.

*I have cited the sources of all quotations, paraphrases, summaries of information, tables, diagrams or other material; including software and other electronic media in which intellectual property rights may reside.

*I have provided a complete bibliography of all works and sources used in the preparation of this submission.

*I understand that failure to comply with the Institute's regulations governing plagiarism constitutes a serious offence.

Student Name: (Printed) _____

Student Number(s): _____

Programme Title & Yr: _____

Module : _____

Signature(s): _____

Date: _____

Please note:

a) Individual declaration is required by each student for joint projects.

b) Where projects are submitted electronically, students are required to type their name under signature.

c) The Institute regulations on plagiarism are set out in Section 10 of Examination and Assessment Regulations published each year in the Student Handbook