UiA University of Agder

Master

Combating class imbalances in image classification - a deep neural network-based method for skin disease classification

EIVIND AAMODT

SUPERVISOR
Morten Goodwin

University of Agder, 2022
Faculty of Engineering and Science
Department of ICT

Obligatorisk gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

1.	Vi erklærer herved at vår besvarelse er vårt eget arbeid, og at vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.	**Ja** / Nei
2.	**Vi erklærer videre at denne besvarelsen:** • Ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. • Ikke refererer til andres arbeid uten at det er oppgitt. • Ikke refererer til eget tidligere arbeid uten at det er oppgitt. • Har alle referansene oppgitt i litteraturlisten. • Ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse.	**Ja** / Nei
3.	Vi er kjent med at brudd på ovennevnte er å betrakte som fusk og kan medføre annullering av eksamen og utestengelse fra universiteter og høgskoler i Norge, jf. Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§ 31.	**Ja** / Nei
4.	Vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert.	**Ja** / Nei
5.	Vi er kjent med at Universitetet i Agder vil behandle alle saker hvor det forligger mistanke om fusk etter høgskolens retningslinjer for behandling av saker om fusk.	**Ja** / Nei
6.	Vi har satt oss inn i regler og retningslinjer i bruk av kilder og referanser på biblioteket sine nettsider.	**Ja** / Nei
7.	Vi har i flertall blitt enige om at innsatsen innad i gruppen er merkbart forskjellig og ønsker dermed å vurderes individuelt. Ordinært vurderes alle deltakere i prosjektet samlet.	Ja / **Nei**

Publiseringsavtale

Fullmakt til elektronisk publisering av oppgaven Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven. §2).
Oppgaver som er unntatt offentlighet eller taushetsbelagt/konfidensiell vil ikke bli publisert.

Vi gir herved Universitetet i Agder en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering:	**Ja** / Nei
Er oppgaven båndlagt (konfidensiell)?	Ja / **Nei**
Er oppgaven unntatt offentlighet?	Ja / **Nei**

Abstract

Skin cancer is the most common type of cancer globally, and the current estimates say that as many as one in five Americans will develop one form of skin cancer in their lifetime. While there are many types of skin cancer, some of the more severe types can lead to life expectancies of less than five years if the cancer is left untreated. While these statistics are very severe, early detection and treatment are very prominent, reaching survival rates as high as 99% in cases where the cancer was spotted and removed early on. Unfortunately, traditional methods of diagnosis are very time-consuming and not always accurate, causing skin cancer to be one of the cancer types with the highest misdiagnosis rate.

Due to early diagnosis and treatment having such a critical role in patients' survival rate and life expectancy, together with the time-consuming and high error rate of classical diagnosis methods, a better method of diagnosing skin cancer is needed. Deep learning and convolutional neural networks have shown very promising results in image analysis in the past few years due to the ability to extract multiple features out of an image that are not recognizable to humans. Tasks that predominantly contain visual symptoms, such as skin lesions in this case, are perfect use cases where convolutional neural networks shine.

In this thesis, I have taken a deep dive into how convolutional neural networks can be used on dermoscopic images of skin cancer. To see how they performed, multiple state-of-the-art CNN models were tested, such as ResNet34, VGG16, and EfficientNet_B4. Experimentation with methods such as utilizing different types of image augmentation, oversampling, class grouping, different train/valid/test splits, and using multiple models that vote and act as a jury were conducted to combat the severe class imbalances in the dataset. The best models achieved a classification accuracy of 84.62% between the seven classes and 85.81% classification accuracy when working as an anomaly detector.

In cases where a hospital wants to reduce the workload by making the model perform all of the easy classifications, a model using confidence thresholds was made. The threshold can be changed based on the accuracy requirements of the hospital. For example, If a 90% accuracy is required, the model will achieve it while diagnosing 90% of the patients. If a 95% accuracy is required, it will achieve the results by diagnosing 70% of the patients. In cases where the network is not allowed to make any mistakes, the model managed 99.5% accuracy while still diagnosing 40% of the patients, almost cutting the hospital's workload in half.

All of my models outperformed human experts (60-76%) by a large margin. Because there is such a significant increase in performance, utilizing these new artificial intelligence models in the real world can save countless lives.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Artificial Intelligence (AI) and image classification have many different use cases in the real world. Some common examples are within the healthcare industry, where it can be used to analyze medical images to diagnose different diseases. In the automobile industry, where it is being used to train self-driving cars through cameras and ultrasonic sensors, and many other areas where giving machines the ability to make decisions based on visual inputs improves their performance and accuracy.

This master thesis will focus on the medical industry, specifically taking a deep dive into different skin diseases and how the diagnosis process can be improved through medical imaging and image classification using Convolutional Neural Networks(CNN).

Within the healthcare sector, many patients experience different kinds of skin diseases. While these skin diseases vary in severity, and some go away on their own after a while, other diseases can become very dangerous if they are simply left without getting treatment. Throughout history, patients have usually been required to visit their doctor or seek out skincare specialists such as dermatologists, which can be very time-consuming and expensive in some parts of the world. There are often long queues, and it can go weeks from when a patient asks for a consultation and when the specialists finally have time to see them to check it out.

Because these issues persist all around the world and the fact that technology has come further, many companies such as Teladoc Health and even Amazon[54] have started to focus more and more on the telehealth sector, which has turned into a multi-billion dollar industry[14]. This goes to show that the ability to conduct remote health consultations is something that the world needs and that it can improve our overall health as a society.

> "Telehealth is the use of digital information and communication technologies, such as computers and mobile devices, to access health care services remotely and manage your health care. These may be technologies you use from home or that your doctor uses to improve or support health care services"[60].

There are many diseases within the healthcare industry where the diagnosis criteria are largely based on visual inputs, so getting a convolutional neural network to do some of the classifications can improve the overall diagnostic accuracy and speed of diagnosis.

The motivation for doing this thesis is to explore different state-of-the-art networks to see how they compare when dealing with different kinds of skin diseases and to use many different techniques to try and improve the overall classification accuracy. These techniques consist of using different CNN architectures, multiple types of image augmentation to increase the number of images to train on, testing varying sampling techniques to improve the class imbalances, training multiple networks to decide by majority voting, and more.

This is not a trivial task due to the difficulty of differentiating between some diseases and the very high class imbalances in the datasets. Many skin diseases look almost identical, which can trick the neural network into focusing on the similarities rather than the tiny differences. There is also a vast range of different ways some diseases can look, where some diseases can even overlap so that the classification can be challenging even for human experts. Skin diseases and medical imaging in general also have a very skewed amount of benign photos compared to images where there are any diseases, simply because some of the diseases are very rare in the real world. When collecting images in a medical setting, the benign cases will naturally occur many more times than the dangerous cases, causing dataset imbalances that must be dealt with during the training process.

The dataset chosen in this thesis contains seven different skin diseases: Actinic keratoses and intraepithelial carcinoma / Bowen's disease (akiec), Basal cell carcinoma (bcc), benign keratosis-like lesions (bkl), Dermatofibroma (df), Melanoma (mel), Melanocytic nevi (nv) and Vascular lesions (vasc), and through using different methods to combat the class imbalances and by using multiple CNN architectures the best model achieved a classification accuracy of 85%.

1.2 Research question and hypotheses

The main goal of this master thesis is to create an artificial neural network model capable of classifying seven different types of skin diseases based on images of skin lesions.

1.2.1 Thesis goals

Goal 1: *Explore different state-of-the-art convolutional neural network architectures to figure out what the main differences are*

Goal 2: *Find out what techniques are predominantly used to combat class imbalances in datasets*

Goal 3: *Test the differences when training a neural network to be used as an abnormality detector (cancer/non-cancer) vs training a neural network to classify between seven different skin diseases*

Goal 4: *Achieve a higher classification accuracy than human experts who are working in the field of dermatology*

Goal 5: *Train multiple models which can be used together when classifying images by using a voting-based system*

Goal 6: *See how the accuracy changes when using difference confidence requirements during the classification phase, and see how many images the networks skips ("unsure, go see a doctor for further testing") when doing so.*

1.2.2 Hypotheses

Hypothesis 1: *The network will be able to achieve higher accuracy than human experts with more than ten years of experience (62.6%)*

Hypothesis 2: *Multiple neural network models working together by voting will perform better than the best network working alone*

1.2.3 Summary

The goals are based on creating a neural network using state-of-the-art architectures and ending up with a model that can perform the classifications to a satisfying degree. To achieve this, multiple models will be trained using different convolutional neural network architectures and varying image augmentation methods, which will be evaluated on a test set to find out which methods result in achieving the best performance.

If successful, the models proposed could become an incredible aid for doctors who are making multiple diagnoses each day, resulting in fewer misdiagnoses in the medical field.

1.3 Thesis outline

Chapter 2: Background, where Artificial Intelligence (2.1), Artificial Neural Networks (2.2), Deep Neural Networks (2.2.4), Convolutional Neural Networks 2.3, and the different skin diseases (2.4) is explained to gain the required background knowledge.

Chapter 3: State-of-the-art, where the best performing Convolutional Neural Networks (3.1) and the top performing skin cancer classification methods (3.2) is investigated.

Chapter 4: Methods, where the dataset (4.2), methods to combat the dataset imbalances (4.3), and building the CNN architectures (4.4) are discussed.

Chapter 5: Results, where the results on the 20valid 20test split (5.1), 80/10/10 split (5.2), binary classification between dangerous and benign (5.3), using multiple models that votes and act as a jury (5.4) and using different thresholds when making the classification (5.5) is shown.

Chapter 6: Discussions, where the results from 20valid 20test split (6.1), results from 80/10/10 split (6.2), binary classification (6.4) and using multiple models that votes (5.4) is discussed further and the results are evaluated.

Chapter 7: Conclusions, where the the results and findings are assessed against the hypoteses and goals (7.1), and future work is discussed (7.3).

Chapter 2

Background

2.1 Artificial Intelligence

Artificial Intelligence (AI) is the ability of a digital computer or robot to perform and automate difficult tasks that require complex behavior commonly associated with intelligent beings[16]. The AI field has improved greatly in the past decade, accomplishing many things that were never possible before using standard algorithms.

Most people have heard about artificial intelligence through things like Tesla, which is working on self-driving cars[65] or how a machine was able to beat the world chess champion in a game of chess for the first time in 1997, which made international news[25]. However, how does it work, and how can a machine do these challenging tasks that require critical thinking and intelligent decisions based on the information given? This chapter will discuss what AI is and how it manages to complete these complex tasks.

2.2 Artificial Neural Networks

Because Artificial Neural Networks(ANN) are largely based on how our brain functions and how neurons get activated and passes information through sending signals to other neurons, section 2.2.1 will briefly explain how the human brain neurons functions to get a better understanding of what the ANNs are modeled after.

2.2.1 Neurons in the human brain

Human brains consist of upwards of 100 billion neurons which serve as the building blocks of our nervous system[11]. These neurons are cells that are capable of communicating with other cells by sending electrical or chemical signals through a process called synapse[31]. Neurons are the fundamental components of our brain and nervous system, and the cells are responsible for receiving sensory input from the external world. These sensory inputs are "digested" and translated into your body performing actions by sending different commands to our muscles based on the sensory inputs they receive[32].

While neurons are not receiving any stimuli, they are in what is called a "resting potential." In this state, the neurons are waiting to receive stimuli, which can be everything from pressure to pain, with each neuron having different sensitivity to different stimuli.

Once the stimuli pass a threshold level for the specific neuron, showcased below in Figure 2.1, the neuron gets activated and fires, sending a signal to other neurons. Because our brains have billions upon billions of neurons, and whenever one neuron activates, it can trigger other neurons to activate, the system is incredibly complex, and the neurons can trigger in almost an infinite number of combinations.

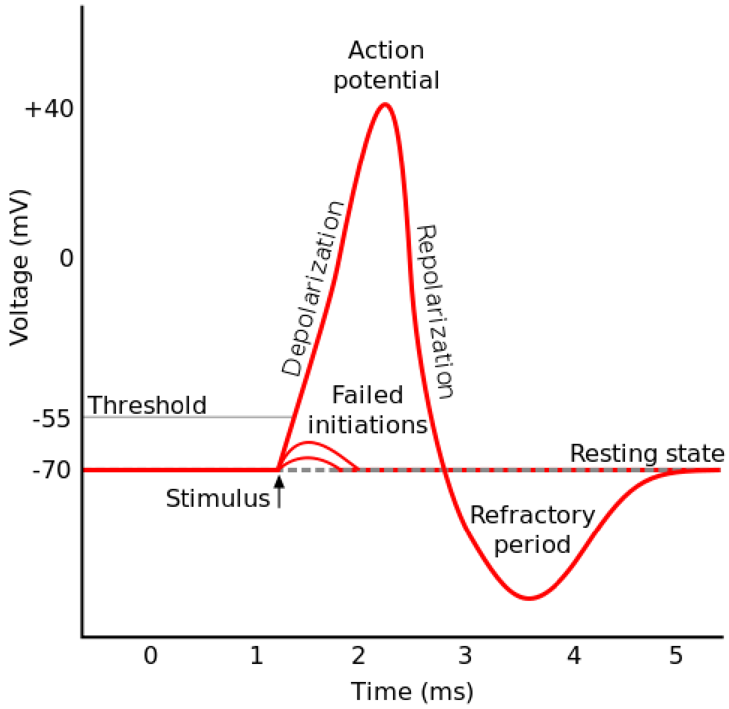


Figure 2.1: Neuron activation[47]

2.2.2 Artificial Neural Networks

Similar to how brain neurons function by receiving stimuli, then evaluating the importance of said stimuli before activating and sending signals to other neurons, ANNs are built up by a collection of nodes (often called artificial neurons) that pass information to each other.

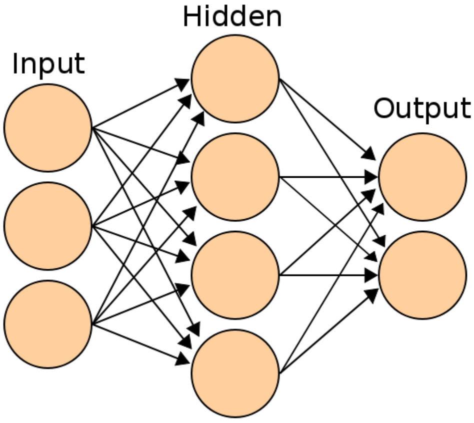


Figure 2.2: Multilayer perceptron. A common form of artificial Neural Network (ANN)[69]

The initial value of each input neuron will depend on the color of the corresponding pixel in the image. For example, if we have a grayscale image, totally white pixels can have the value of one, and black pixels can have the value 0, while the different shades of grey will have values between 0 and 1. To show what this means in practice, Figure 2.4 shows an example from the MNIST dataset.

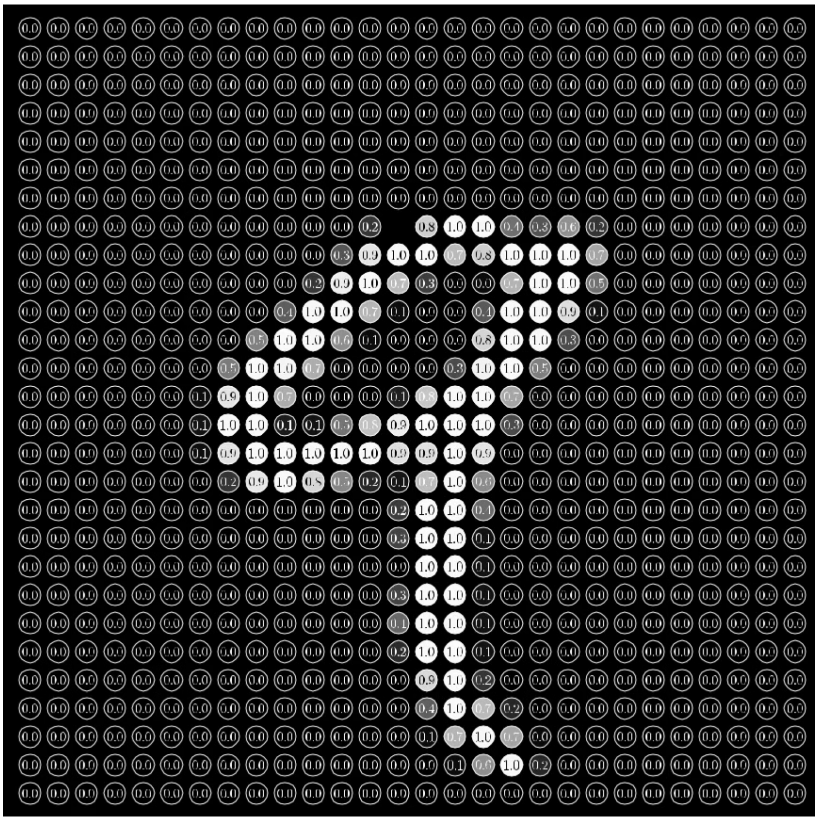


Figure 2.3: Input values from a 28 x 28 image. Black pixels have the value 0.0, white pixels have the value 1.0, and grey pixels have values ranging from 0 to 1 depending on how bright they are. [1]

After the input layer has been filled from the input image, the next hidden layer will begin to receive inputs based on the weights the network has acquired during training. The values of each neuron will depend on inputs from the previous layer, weights, biases, and the activation function used.

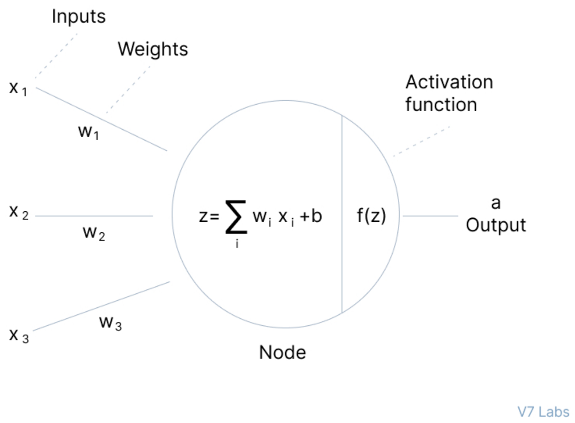
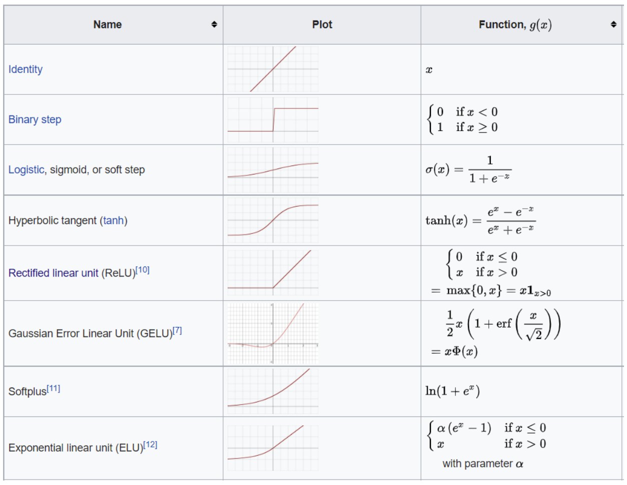


Figure 2.4: Neuron calculation based on weights, input, bias and activation function. [5]

2.2.3 Activation Functions

We have to use activation functions in deep neural networks to add non-linearity to the network. Much like when a brain neuron receives stimuli and decides whether or not the information received is important enough to activate it, neurons in ANNs also go through an activation function to determine if the neuron should become activated and what value the neuron should pass along. Without activation functions, you can have one hidden layer or a hundred hidden layers in a row, with each performing a linear function based on weights and biases. It would not change anything because the composition of multiple linear functions is a linear function itself[5]. This causes the neural network to become a linear regression model, not capable of solving more complex tasks.

There are many different activation functions, with some common examples being showcased in Figure 2.5, but ReLU is the activation function which is used the most by far[35]. This is because with ReLU, there are a lot fewer neurons that will become activated when compared to functions like sigmoid and tanh causing the computations to become a lot more efficient. The function being linear also causes the gradient descent towards the minimum of the loss function to converge quicker than for functions whose output are capped at a number like 1[5].



Name ⬍	Plot	Function, $g(x)$ ⬍
Identity		x
Binary step		$\begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$
Logistic, sigmoid, or soft step		$\sigma(x) = \frac{1}{1+e^{-x}}$
Hyperbolic tangent (tanh)		$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
Rectified linear unit (ReLU)[10]		$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ $= \max\{0, x\} = x\mathbf{1}_{x>0}$
Gaussian Error Linear Unit (GELU)[7]		$\frac{1}{2}x\left(1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right)$ $= x\Phi(x)$
Softplus[11]		$\ln(1+e^x)$
Exponential linear unit (ELU)[12]		$\begin{cases} \alpha\left(e^x - 1\right) & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ with parameter α

Figure 2.5: List of activation functions

2.2.4 Deep neural networks and hidden layers

While the image in Figure 2.2 shows an ANN with only one hidden layer, deep neural networks are built the same way with the exception of having multiple hidden layers in the middle. Having multiple layers can be a big bonus for the network because each layer can focus on different things.

Using the example with the hand drawn numbers again, the input layer will consist of the image's pixel values. For example, the first hidden layer can learn small parts of different shapes, while the second layer can learn bigger shapes like circles, moon shapes, and lines based on what shape parts exist in the first hidden layer. The output layer will then be calculated based on the last hidden layer.

- The hidden layer only found a circular shape? the number has to be a 0.
- The hidden layer only found one line? the number has to be a 1.
- Two circular shapes? the number has to be an 8.

All of these calculations are based around using weights and biases. Individual neurons connect to every single neuron in the next layer, and the value they pass through to each one depends on the weighting. Suppose we use the previous example where a neuron in the hidden layer learned to detect a circle. In that case, it will give positive weight to numbers that include a circle (0, 6, 8, 9) while giving negative weights to numbers which does not include circles (1,2,3,4,5,7). If the hidden layer detects a circular shape, this will result in the numbers that include a circle increasing in value, while the numbers that do not include a circle decrease in value. An example of what the hidden layers could be detecting can be shown below in Figure 2.6.

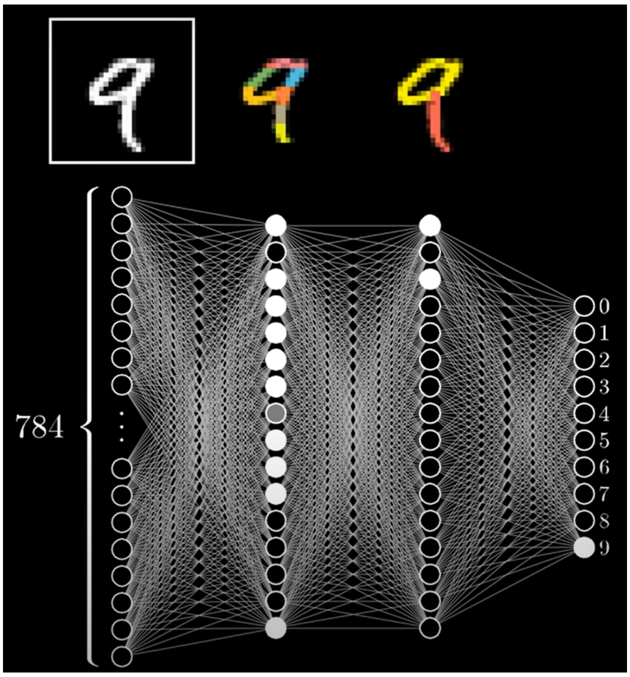


Figure 2.6: Example of what the hidden layers could be detecting[1]

Neural networks being able to learn these different features and which features correspond to which classification without human interaction is incredibly valuable because it is possible to solve machine learning tasks in different fields without being an expert at the underlying tasks. The networks can also figure out challenging patterns in complex images such as cancer images, which would be impossible to detect for humans.

While bigger image sizes give more information to the network that it can learn from, it also increases the memory requirement and computation time by a very large amount which means a middle ground has to be found. When working with images, each pixel in an image has to correlate with one neuron in the input layer of the ANN. Because of this, the input layer has to be of size width x height, which causes the network to increase in size exponentially compared to the size of an image. When working with small images such as the MNIST dataset, which is built up of hand-drawn numbers, the image size is only 28x28. Small images like this result in 784 input neurons, while increasing the image size to 112x112 results in 12544 input neurons, and an image size of 224x224 needs more than 50 thousand input neurons.

2.2.5 Loss functions and gradient decent

When an artificial network is training, it needs to evaluate how well the network is doing. To accomplish this, the network needs to have a loss function, also commonly called a cost function, which will calculate how far away a given prediction is from the truth.

> "The cost function reduces all the various good and bad aspects of a possibly complex system down to a single number, a scalar value, which allows candidate solutions to be ranked and compared[8]."

There are many loss functions commonly used, but all of them share that they want to assign the most accurate cost value to the given answer. This is required, so the network knows what it did wrong to be able to improve upon the mistakes. The average cost over all available training images indicates how well the neural network is doing. Because a singular value measures the performance of tens of thousands of inputs, weights, and biases, it is essential to use a good loss function suitable for the given task to make it accurate.

Because the loss function calculates the error when the network is trying to accomplish its task, the goal of the network should be to minimize the total loss achieved during training. The optimization process used to achieve this goal is most often gradient descent, which works by iteratively moving in the steepest direction of the loss function, which is defined by the negative of the gradient[23].

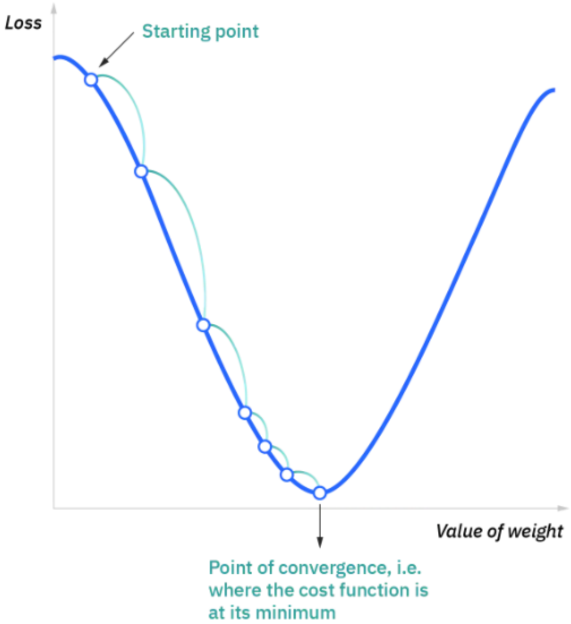


Figure 2.7: Simple gradient descent[20]

2.3 Convolutional neural networks

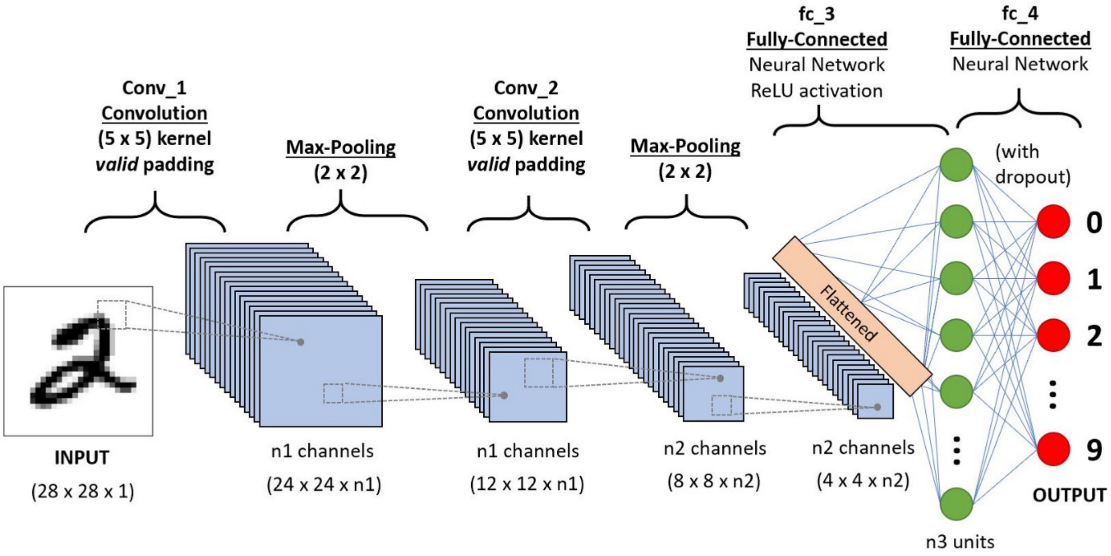


Figure 2.8: Convolutional Neural Network architecture [53]

The most common class of ANN models when it comes to image analysis is Convolutional Neural Networks (CNN)[6]. These networks are used to enable computer vision and to accomplish image analysis and classification through deep learning. The architecture consists of a multilayered neural network that specializes in detecting complex features in the data presented[53].

However, why is CNNs more used when it comes to image analysis than classical feedforward neural networks such as a multilayer perceptron (MLP) shown earlier in Figure 2.2?

While convolutional neural networks also have a fully connected neural network part at the end of the process, which can be seen in Figure 2.8, they do a lot of work on the images before they feed the inputs into the fully connected layer. The goal of the work they do is to reduce the images and turn them into a form that keeps the features that are important to distinguish between the different classes while also being much smaller and easier to process[53]. This work mainly consists of convolutions and pooling, which greatly reduces the number of inputs sent to the fully connected layer. The convolution and pooling process will be explained later in chapter 2.3.1 and 2.3.2.

This differs significantly from MLPs, where as mentioned earlier, each pixel in the images needs to correspond to one neuron in the input layer. The input space becomes huge for large images and scales very poorly. With modern cameras and even most of the newer phones being able to capture images as high as 4K resolution, meaning 3840 x 2160 pixels, feeding them into a multilayer perceptron would need more than 8.2 million input neurons! While 4K resolution images are not being used in CNNs either because they are so large, medium-sized images are also much more effective to run on CNNs than on MLPs. Feature learning and massive scalability improvements make it favorable to use.

Through training, the convolutional neural network learns what information is important and how the different characteristics relate to the different classes in the dataset. The classification works because the network assigns different weights and importance to all of the different information that it can extract out of an image. The information can consist of different shapes, patterns, color schemes, and more.

Let us use an example dataset that consists of cats and dogs. While humans find it incredibly easy to distinguish between a cat and a dog, it can be hard to describe exactly which characteristics determines what type of animal it is. Is it the size? Dogs are usually larger than cats, but some dogs are tiny, and some cats are very large. Is it the ears? There are many kinds of dogs that have pointy ears as well. Because of this, it would be a challenging task to create an algorithm to classify between them. While creating a simple algorithm such as:

$$small + pointy\ ears = cat$$

$$big + sloppy\ ears = dog$$

might be correct in some cases, but it has many flaws and will get many wrong classifications.

This is where the beauty of CNNs comes in. When working with convolutional neural networks, the network itself learns and assigns which features are present in the data and how much weighting it should give the different features. When working with challenging data, many of these features can be things we as humans do not recognize or know exists, but the computer might recognize that they are important when figuring out the result. CNNs mainly consist of three major steps, and the following chapters will delve deep into how the network operates from start to finish:

1. The convolutional layer
2. The pooling layer
3. The fully connected layer

2.3.1 Layer 1: Convolutional layer

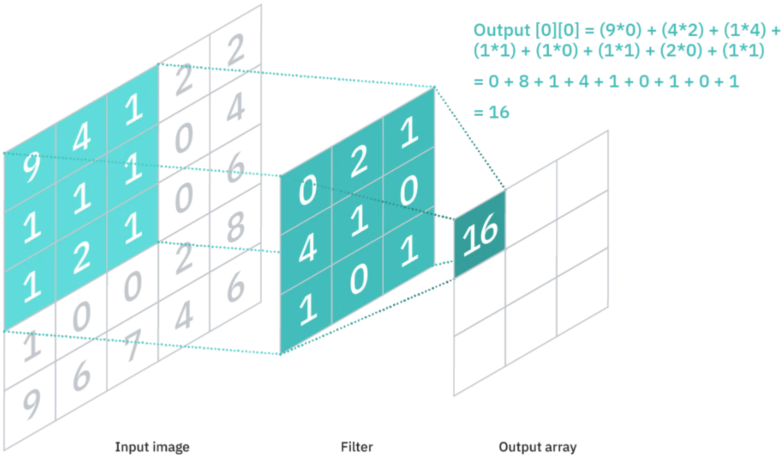
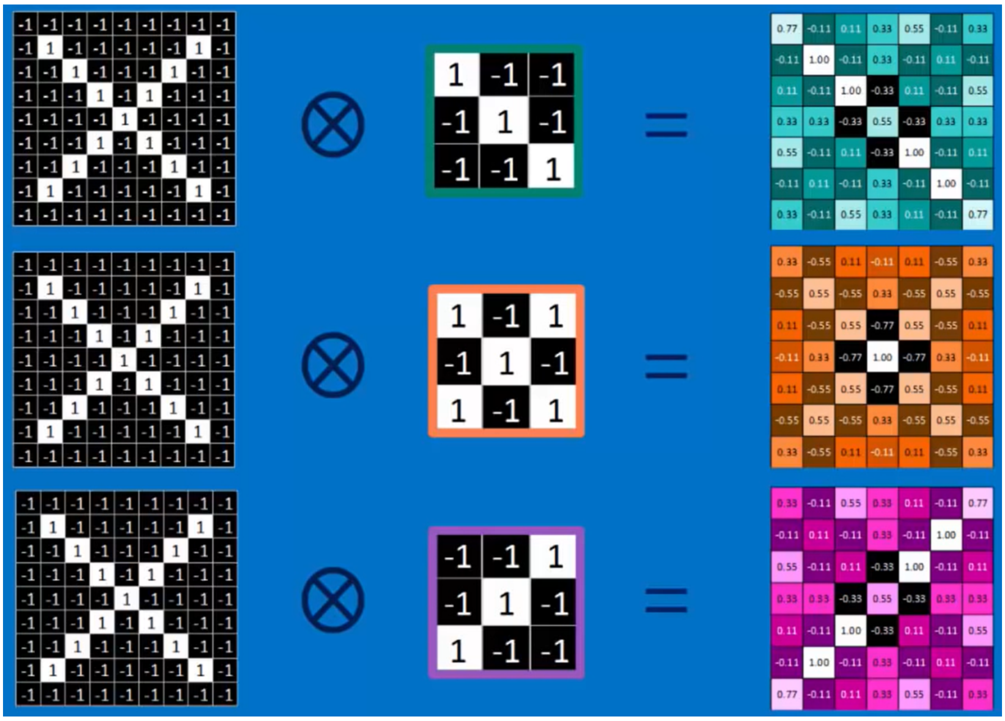


Figure 2.9: CNN filter or kernel [19]

The convolutional layer is the first layer in a convolutional neural network, and this is where most of the work is done. This is the process of finding which features are present in the image, also called feature extraction, through the use of filters (or kernels) passed along the image.

A filter is a 2D array with weights, looking for a specific pattern in the given area. Often the filter is of size 3x3, and the stride (distance it moves) is 1, but these hyperparameters can be changed to fit the desired goals[53]. The filter starts in the upper left corner and multiplies the values in the image with the values in the filter. If the output value is a large number, the specific pattern is most likely present, and if it is a low number, the specific pattern is most likely not present as the values in the image do not match with the weights in the filter. The value given from the calculation is stored in an output array shown in Figure 2.9, and then the filter moves to the side to do the calculation once again. After moving along the entire image, having checked for a given feature, the output array will act as a feature map, showing where the specific feature was found in the image.

Because each filter typically only looks for one feature, it is normal to use multiple filters in a convolutional network, looking for different features. To demonstrate this process, we can look at what filters we need to find the different features of the letter X in a given image:



-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

⊗

1	-1	-1
-1	1	-1
-1	-1	1

=

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

⊗

1	-1	1
-1	1	-1
1	-1	1

=

0.33	-0.55	0.11	-0.11	0.11	-0.55	0.33
-0.55	0.55	-0.55	0.33	-0.55	0.55	-0.55
0.11	-0.55	0.55	-0.77	0.55	-0.55	0.11
-0.11	0.33	-0.77	1.00	-0.77	0.33	-0.11
0.11	-0.55	0.55	-0.77	0.55	-0.55	0.11
-0.55	0.55	-0.55	0.33	-0.55	0.55	-0.55
0.33	-0.55	0.11	-0.11	0.11	-0.55	0.33

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

⊗

-1	-1	1
-1	1	-1
1	-1	-1

=

0.33	-0.11	0.55	0.33	0.11	-0.11	0.77
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.77	-0.11	0.11	0.33	0.55	-0.11	0.33

Figure 2.10: CNN multiple filters to find the different features of the letter X[51]

Because the letter X consists of three different features: a diagonal line to the left, a diagonal line to the right, and a section where the lines cross, we have three different filters which look for these features. The feature maps to the right show what we would expect when looking at the image. The diagonal filters show greater values by the diagonal lines and lower values elsewhere. The cross filter shows the highest value in the middle of the image, where the diagonal lines cross. Therefore, these features have been found in the image, making the classification part more straightforward than just looking at the individual pixels.

Because many of these filter calculations give values with negative numbers, the activation function ReLU is applied after each convolutional layer to introduce nonlinearity to the model. By bringing all negative numbers up to 0 instead, it makes sure that these neurons are not activated, saving computational power[19].

2.3.2 Layer 2: Pooling layer

After the convolutional layer of a CNN, we have a pooling layer which is used to reduce the amount of complexity and parameters in the input. Like how the convolutional layer sweeps a filter through the input image and uses different weights to determine if the image matches a specific pattern, the pooling layer also sweeps a filter through the finished feature map. This filter, however, does not have any weights and is not looking for any features or patterns. Instead, this filter applies an aggregate function to reduce all of the values into a single number[19].

There are mainly two different pooling used, with max-pooling being the most popular by far and average pooling being used in fewer cases. When max pooling is used, the filter chooses the highest value within the filter as an output, and when average pooling is used, the values within the filter are averaged out to give an output. An example of both pooling methods is shown in Figure 2.11.

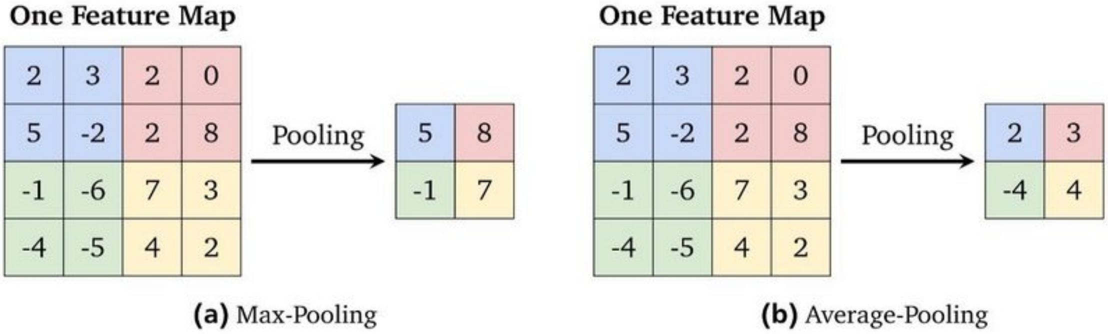


Figure 2.11: Max pooling and average pooling example[26]

Because max-pooling is often used and the pooling filter is applied on the feature map, all matching features are kept even though the output is downsampled. If a feature matched in the convolution layer and got a high value in the feature map, it will be the highest number in the pooling filter, and as such, it will be the value saved in the new layer. By keeping the matching features but removing useless outputs, the complexity will be reduced, reducing the computation requirements by a whole lot. A max-pooling filter with a size of 2x2 and a stride of 2 is very common, and it reduces the number of outputs by **75%!** A 4x4 feature map which is 16 outputs, gets turned into a 2x2 feature map which is only four outputs. This decreases the number of learnable parameters and substantially speeds up the fully connected layer part.

Because the max-pooling filter looks at multiple values within each area, it causes the feature map to become invariant to small shifts and distortions and reduces the amount of overfitting to the training data[50].

2.3.3 Layer 3: Fully connected layer

After the convolutional layer and pooling layer have found all of the features, the resulting feature maps get flattened and sent into a fully connected layer which works similarly to the architecture described in chapter 2.2.2. This time, however, the input layer does not need one neuron for all thousands of pixels in the image but rather one neuron for each feature found in the feature map. This causes the fully connected layer to work a lot faster. The network can find non-linear combinations in the high-level features presented by the outputs of the earlier layers.

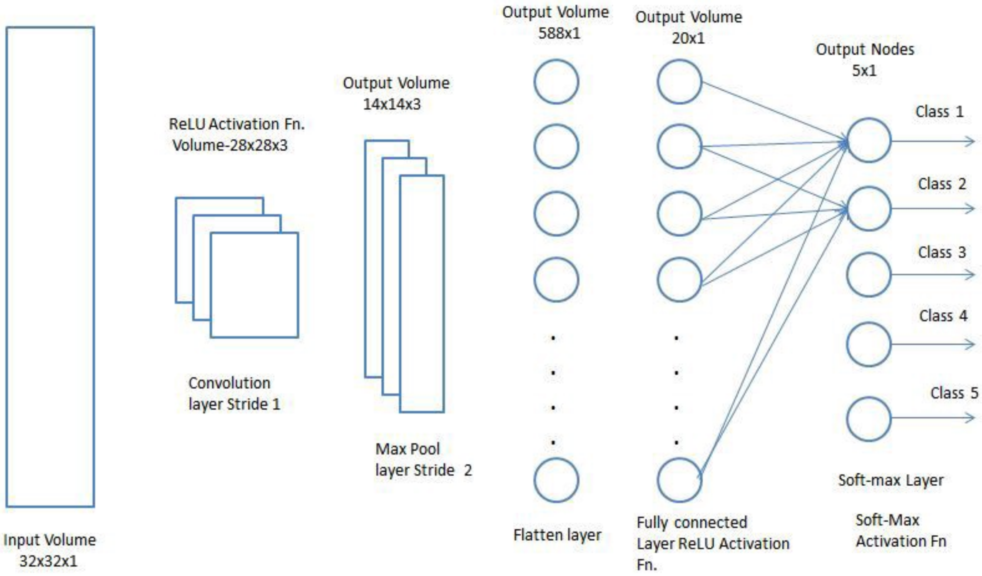


Figure 2.12: CNN architecture - the last three steps being the fully connected layer

The fully connected layer learns similarly to simple multilayer perceptrons through backpropagation. In the end, the output given by the fully connected layer determines what the given classification is.

2.3.4 Convolutional Neural Networks summary

Convolutional neural networks are extremely good at image classification, and there is no surprise that it is the most commonly used architecture for image analysis. How it works can be split into three different parts:

- 1 - Convolutional layer: uses filters to find all of the patterns and features which are present in the image and saves them in feature maps
- 2 - Pooling layer: downsamples the resulting feature maps by using pooling functions to reduce the number of useless features and to lower the number of parameters needed in the fully connected layer
- 3 - Fully connected layer: is given a map of all the given features found in the image and learns how the given features determine which class the image belongs to through a multilayer perceptron.

The convolutional and pooling layers can be stacked multiple times to find higher-level patterns. However, the resulting feature maps always go through a fully connected layer at the end to classify the given features into one of the classes in the given dataset.

2.4 Skin diseases

Because this master thesis is based on medical images and different skin diseases, this section will briefly explain how the images are obtained and what the diseases are. Skin lesions are skin areas that look different from all the surrounding skin, the most common being patches with different colors or bumps in the skin.

2.4.1 Dermatoscopic imaging

Dermoscopy is a technique that is primarily used to examine pigmented skin lesions. The process utilizes a high-quality magnifying lens with a powerful lighting system to get a better view of the structures and patterns of the skin lesion[43].

2.4.2 Skin diseases used in this project:

Actinic keratoses and intraepithelial carcinoma / Bowen's disease (akiec)

Akiec is a skin lesion that consists of an area with thick, scaly, or crusty skin[17]. The skin lesions most commonly appear in areas exposed to the sun, and they become more common for people who have fair skin and are often outside in sunny weather. The skin damage occurs from UV radiation exposure over the course of multiple decades or harsher radiation from sunbeds which are commonly used during the darker months of the year.

The skin lesions are pre-cancerous, which means that while they do not contain cancer themselves, they are associated with an increased risk of developing into cancer if they are left untreated[28]. Untreated lesions contain upwards of 20% risk of progressing into Squamous Cell Carcinoma, the second most common form of skin cancer. Akiec lesions often have a rough texture, so it is possible to notice their feel before they are visible, so early detection should be possible.

Because of the significant risk of the skin lesion developing into a more severe skin cancer, it is important to diagnose them early and intervene with different treatments before the relatively unharmful skin lesion turns into a severe medical emergency.

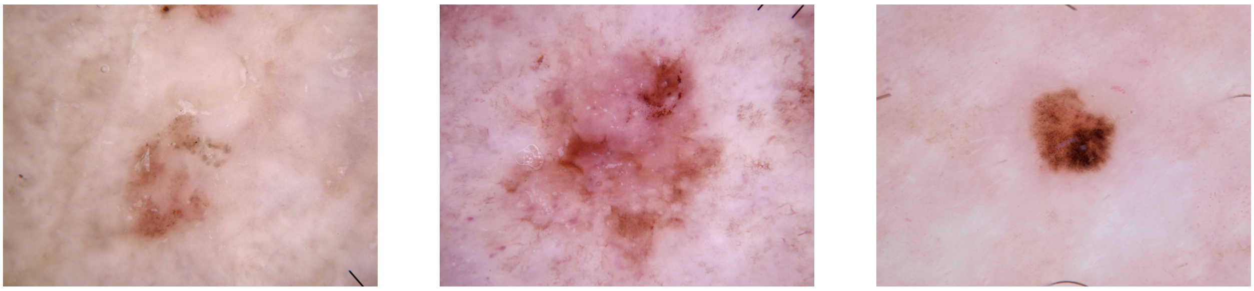

(a) Example 1 (b) Example 2 (c) Example 3

Figure 2.13: Akiec examples

Basal cell carcinoma (bcc)

Basan cell carcinoma is a cancerous skin lesion that can appear in a few different ways. They can look like open sores, shiny bumps, scars, or red patches of raised skin with blood vessels running through them[34]. While the skin lesion is commonly painless, they present a danger to the skin around because they continue to grow and destroy the skin around the lesion. The cancer cells will stay relatively local to the skin lesion and are very rarely deadly, but they will grow deeper into the skin and eventually destroy tissue and bone if left unchecked.

BCC is the most common type of skin cancer, and it accounts for more than 30% of all cancer cases in the world[70]. BCC is also caused by sun exposure and damage caused by UV radiation, and the most common treatment is to remove the affected area surgically. Early diagnosis and treatment are vital as early treatment will prevent any meaningful complications from occurring, and it will reduce the chance of the disease recurring after removal.

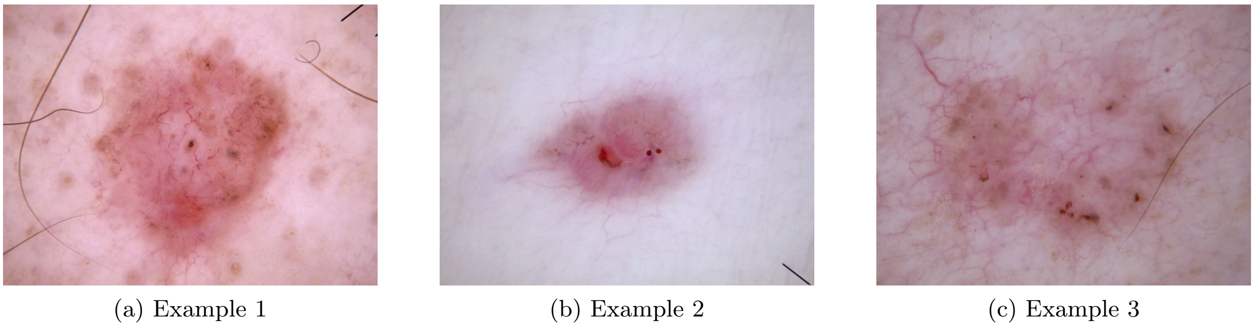

(a) Example 1 (b) Example 2 (c) Example 3

Figure 2.14: bcc examples

benign keratosis-like lesions (solar lentigines / seborrheic keratoses and lichen-planus like keratoses, bkl)

Benign keratosis is a non-cancerous skin growth that is very common to get more of as you get older. BKL is harmless and does not need treatment, but it is possible to remove them if the patient wishes to do so due to cosmetic reasons or if they become irritated from rubbing against clothing[59].

The lesions usually grow out of the skin and look like warts, which are rough bumps in the skin with a tan/brown/black color[45]. Because they are so harmless, they are often left alone as they do not grow bigger or spread.

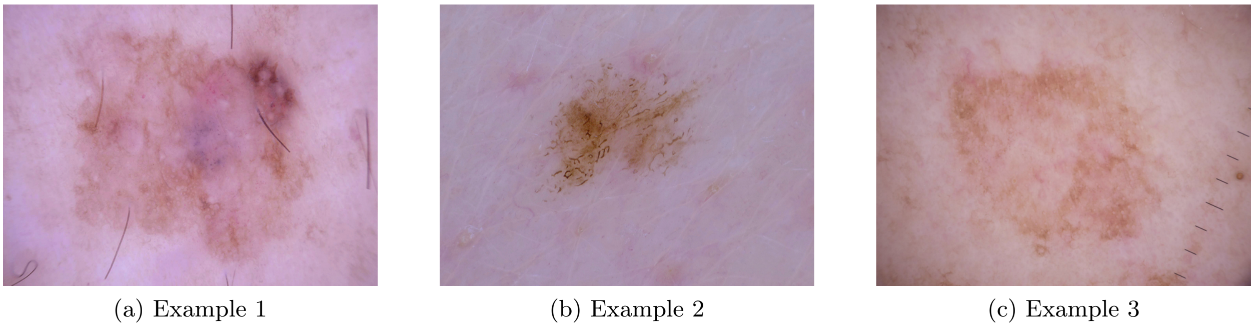
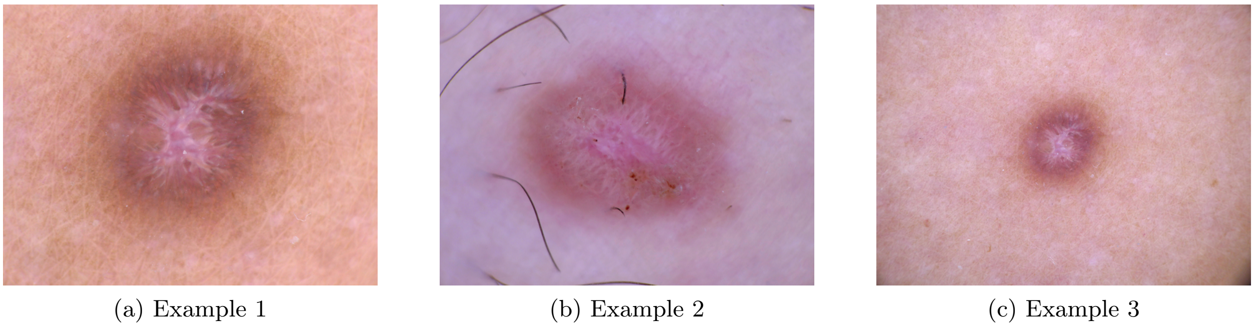

(a) Example 1 (b) Example 2 (c) Example 3

Figure 2.15: bkl examples

Dermatofibroma (df)

Dermatofibroma, just like BKL, is a harmless benign skin lesion which is brown and slightly elevated from the skin. The skin lesions are harmless and painless, and they give no increased risk of developing cancer[42]. Treatment is only considered in cases of abnormal behavior, for example, if it suddenly enlarges or if it is turning into an ulcer. DF commonly looks like normal skin moles.



(a) Example 1 (b) Example 2 (c) Example 3

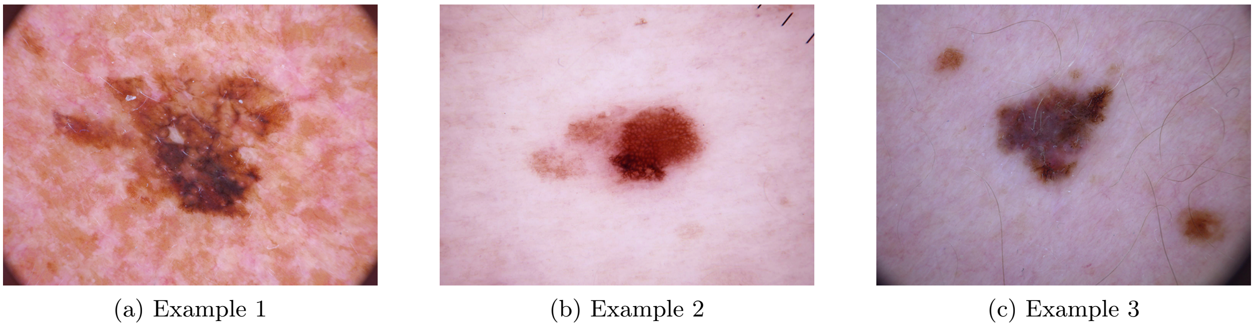
Figure 2.16: df examples

Melanoma (mel)

Melanoma is the most severe type of skin cancer we can get. Melanoma occurs when melanocytes cells, the cells that produce melanin which is what gives the skin a tan or brown color, start to grow out of control[58]. Because the infected cells still produce melanin, the affected skin tumors often appear dark brown or black.

The first sign of melanoma occurring is if an existing mole suddenly starts to grow or change in color. Early diagnosis is essential for melanoma because it spreads incredibly fast. The spread is what makes it the most deadly skin cancer, and being able to detect it and begin treatment by surgically removing it early on will increase the survival changes tenfold[39]. With early treatment and if the cancer is still local, five-year survival rates are as high as 99%. However, if the diagnosis takes too long and the cancer has a chance to spread, five-year survival rates shrink to 65% in cases where it spreads to lymph nodes and a shocking 25% survival rate with distant spread[71].

While it is less common than the other types of skin cancer, it is much more likely to grow bigger and spread around the body, making it more severe.



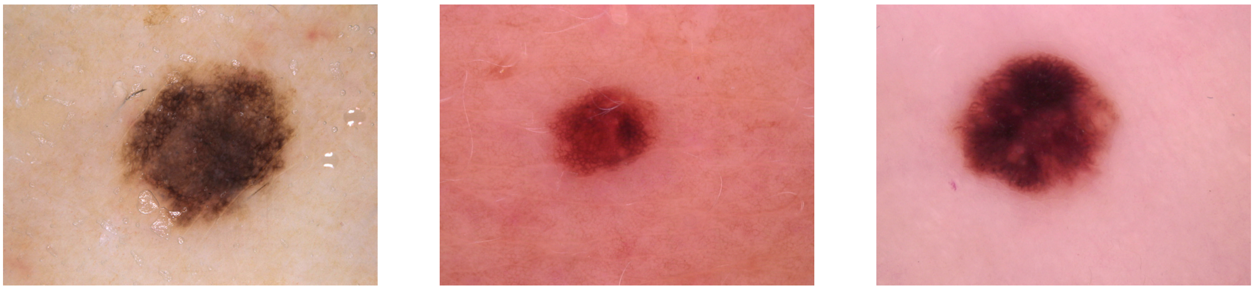
(a) Example 1 (b) Example 2 (c) Example 3

Figure 2.17: mel examples

Melanocytic nevi (nv)

Melanocytic nevi is a form of benign skin lesion which appear as moles. These are mostly formed during birth and childhood, and they appear as small bumps in the skin with a slightly darker color. Because they contain melanin, they are dark brown, black, and in some cases, red.

Nv is entirely harmless and does not cause any physical problems in most cases where it is small, but in cases where the mole is larger than usual, it can carry a small risk of developing cancer later in life[44]. Because these moles are as common and harmless, treatment is generally not advised except for when the mole is much larger than usual and is getting irritated by clothes.



(a) Example 1

(b) Example 2

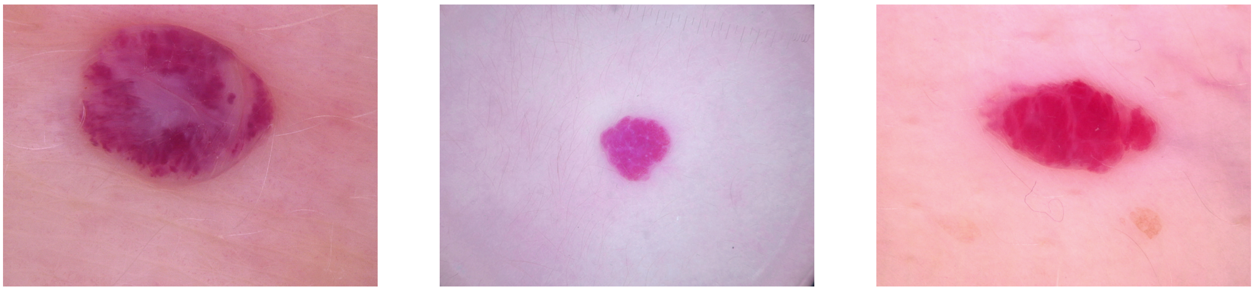
(c) Example 3

Figure 2.18: nv examples

Vascular lesions (angiomas, angiokeratomas, pyogenic granulomas and hemorrhage, vasc)

Vascular lesions are what most people commonly refer to as birthmarks. These abnormalities are caused by blood vessels that have developed in the wrong way and when there is an increased number of vessels which also happens to be enlarged[21]. Because there are different types of vascular lesions, they can vary in appearance. While some appear as tiny red skin growths(angiomas), others appear as red, blue, and purple bumps similar to scabs after injuries (angiokeratomas).

These skin lesions are also benign and are not classified as dangerous or in need of treatment.



(a) Example 1

(b) Example 2

(c) Example 3

Figure 2.19: vasc examples

2.4.3 Skin diseases summary

Because the skin diseases vary in severity, they can have different levels of importance to diagnose early on. Skin lesions such as BKL, DF, NV, and VASC are entirely harmless and do not need treatment or intervention. AKIEC lesions are pre-cancerous and increase the risk of developing cancer, while BCC and MEL have already developed into cancer and require treatment as fast as possible.

There has been a study to see how well humans do at distinguishing these images with over 500 test-takers[46]. Out of 511 people taking the test:

- 283 of 511 human readers were board-certified dermatologists
- 118 of 511 were dermatology residents
- 83 of 511 were general practitioners
- 27 of 511 were human experts with more than ten years of experience

The test takers were given a batch of 30 images to diagnose, and all the humans combined averaged 17.91 correct classifications resulting in 59.7% accuracy. The 27 experts with more than ten years of experience did a little better, averaging 18.78 correct classifications resulting in 62.6% accuracy.

Outside of this dataset human experts tend to achieve around 60-76% through the use of skin biopsy and different methods [10] [40] [46].

Chapter 3

State-of-the-art

3.1 Convolutional Neural Networks

The first modern convolutional neural network was made in the late 1990s, and the work was done by Yann LeCun, Léon Bottuo, Yosha Bengio, and Patrick Haffner. In November 1998, they released a paper named "Gradient-Based Learning Applied to Document Recognition[38]" in which they managed to build a neural network that progressively went from simple feature extraction into complicated features, which was used to classify between handwritten numbers.

The dataset with handwritten numbers is called "The MNIST database of handwritten digits[4]," and the dataset contains 60 000 training images and 10 000 test images. Some examples can be seen in Figure 3.1.

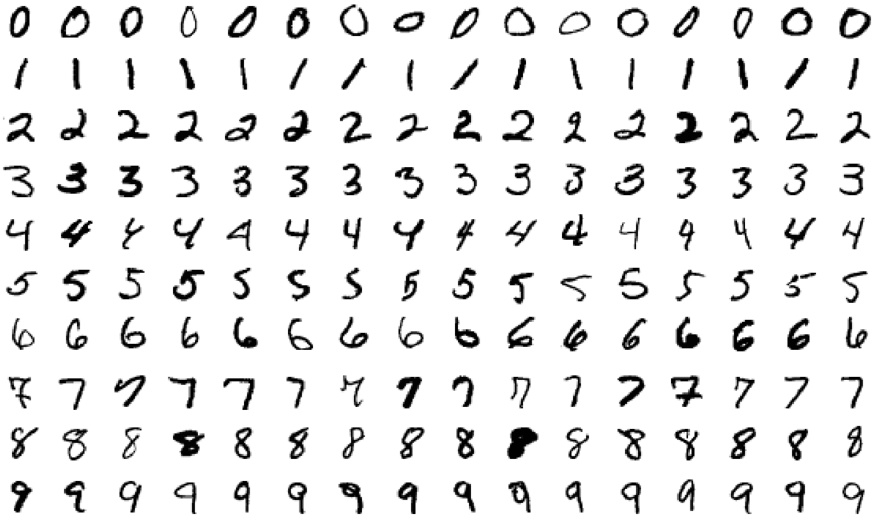


Figure 3.1: MNIST dataset [61]

After Yann LeCun et al. managed to create the first CNN, further research was done in the early 2000s. However, the popularity did not pick up as quickly until AlexNet managed to achieve state-of-the-art accuracy on the ImageNet dataset in 2012[36]. ImageNet is a dataset that is currently being used to benchmark different image classification methods due to its extreme size[29]. The dataset currently contains around 14 million images with more than 21 thousand different classes, which makes it one of the most extensive annotated datasets of images that are commonly used[7]. ImageNet has been a very important tool in advancing computer vision and deep learning research, hosting many challenges where all of the leading image classification methods in the world compete in achieving the highest accuracy[30].

Convolutional neural networks have gotten increasingly more popular over the past decade due to the significant advancements and promising results given by the current state-of-the-art techniques. While AlexNet only managed to achieve 63% accuracy on the dataset in 2012, further advancements in the convolutional neural network scene have managed to increase the accuracy substantially during the past decade[18]. ImageNet is being used as the benchmark for image classification, where 75% accuracy was achieved in 2015, 80% accuracy was achieved in 2016, 85% accuracy was achieved in 2018, and 90% accuracy was achieved in late 2020. Currently, the best performing network is CoCa which was developed in 2022, and by using 2100 million parameters, the network managed to achieve 91% accuracy.

Current CNNs are performing better than human experts, which are estimated to achieve a 5.1% error rate[52]. The current state-of-the-art top-5 accuracies have hit as high as 99%, and the first time it managed to outperform humans was in 2016 when it managed to hit 95.1% accuracy using an alternate form of ResNet.

A history of state-of-the-art models on ImageNet can be seen in Figure 3.2. During the past decade, there have been many considerable improvements. However, the current accuracy has plateaued in the past few years, where most state-of-the-art networks perform very similarly with different use cases.

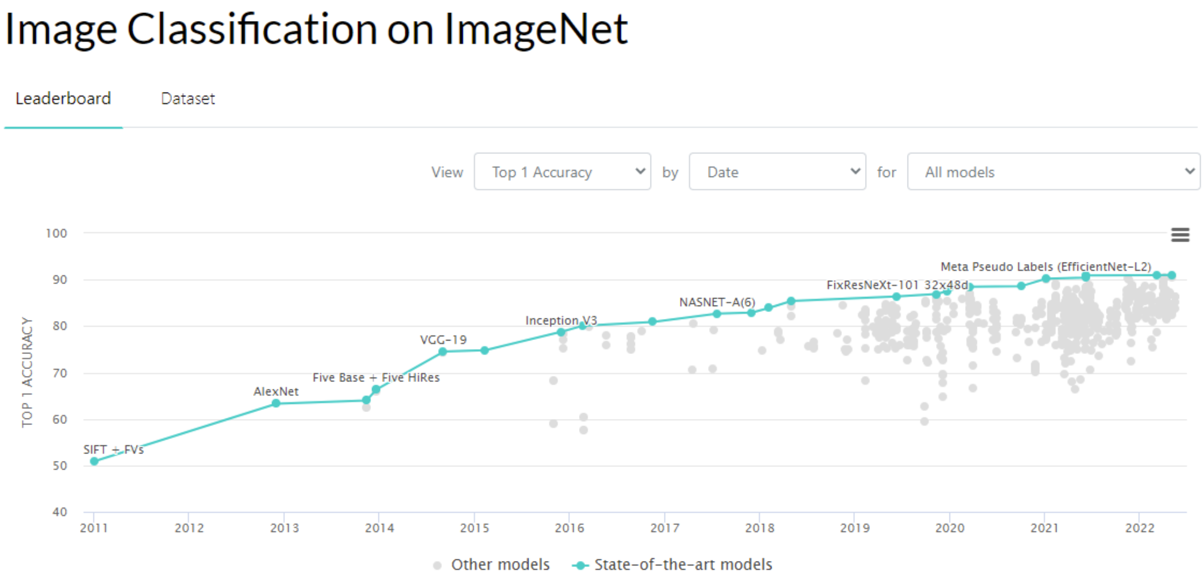


Image Classification on ImageNet

Leaderboard Dataset

View Top 1 Accuracy by Date for All models

Figure 3.2: CNN state-of-the-art accuracy on ImageNet, the current benchmark for image classification [12]

All of these CNN architectures are different. Some focus on being wide, while others are deeper with more convolution layers. Some have a large number of parameters, while others utilize a high resolution. These differences make them have different results, and some CNN networks require a lot more RAM and training time than others while still achieving around the same accuracy. During this project, I will focus on three main CNN architectures: VGG16, ResNet34, and EfficientNet_b4.

3.1.1 CNN architectures used in this thesis

VGG16

VGG16 is a CNN architecture that was published in a paper called "Very Deep Convolutional Networks for Large-Scale Image Recognition[56]," and it was the winning architecture during the 2014 ImageNet competition. The architecture is relatively simple and straightforward, but when the network was released, it managed to achieve a big jump in accuracy compared to the previous state-of-the-art models, and it is considered to be one of the excellent computer vision models to this date[66].

The VGG16 architecture had fewer hyperparameters than the previous architectures and instead focused more on having small convolutional layers with 3x3 filters, 1 stride, same padding, and a max pool layer of 2x2 with 2 stride. In comparison, AlexNet used 11x11 filters which are much larger. Using many 3x3 filters instead of fewer larger filters made VGG stand out, and they found out that two consecutive 3x3 filters provide for an effective receptive field of 5x5, and three 3x3 filters create a receptive field of 7x7[64]. Instead of having big filters, VGG leveraged the fact that multiple 3x3 fields can work together to create much larger fields.

An image of the architecture can be seen in Figure 3.3. As the name implies, the architecture is deep, containing 16 layers, causing the network to be very large and having many parameters. AlexNet only consisted of 5 convolutional layers in comparison, which is less than a third of VGG. This causes the network to use a very long time to train and requires a lot of RAM to run, and even though many newer neural networks are even deeper than VGG, containing more layers, most of them are much smaller when it comes to the number of parameters. The main reason is that VGG16 utilizes a very large number of channels that increase each time the image width and height are reduced through max pooling. The last layers reach as far as 512 channels, and the goal is for each channel to learn different feature information[49].

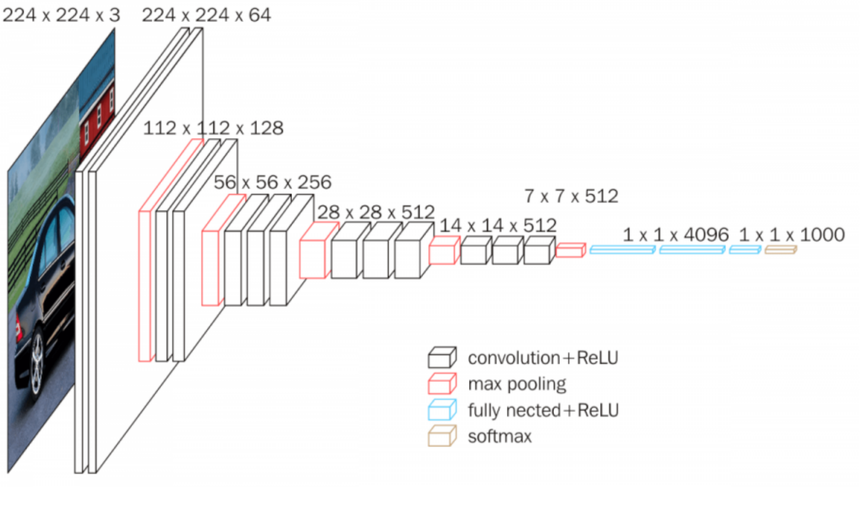


Figure 3.3: VGG16 architecture

ResNet

After VGG16 was released and won the ImageNet competition, it became much more well known that CNN networks tend to increase accuracy when the depth increases. This caused a lot of new research in the area, but many researchers quickly identified that the accuracy starts to decrease after a given amount of depth. While adding more layers is expected to help neural networks, problems such as the vanishing gradient can cause problems to arise in the backpropagation where values are so small that they effectively cause the network to stop updating the weights[72].

There was much research into methods to avoid these issues when creating increasingly deep networks, and ResNet was one of the main methods which were created to solve the problem published in the paper "Deep Residual Learning for Image Recognition [27]" released in December 2015. ResNet managed to create the first feedforward network with hundreds of layers, which is much deeper than all the previous neural networks which existed, and the results gained a lot of traction. The main part of Residual neural networks is the ability to skip layers in the convolutional neural network if the given layer is detrimental to the network's performance. An image of a residual block can be seen in Figure 3.4.

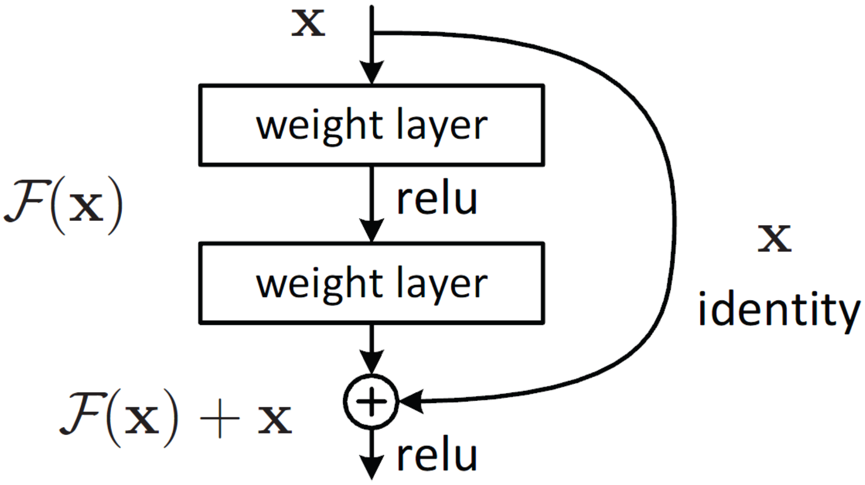


Figure 3.4: Residual learning: a building block[27]

As the image shows, the input into the first block gets sent to the output of the two layers before the second ReLU activation function is completed. Due to ReLU functions returning the same number as the input (as long as it is 0 or above), this is essentially an identity function in the cases where F(X) is close to zero, which is what happens with the vanishing gradient problem[72]. ReLU(x+F(x)) = ReLU(x+0) = ReLU(x) = x. Instead of canceling out all of the gradients just because there is a zero somewhere in the layers, the identity function maintains the signal and sends it through to the next residual block. An example of a full ResNet can be seen in Figure 3.5

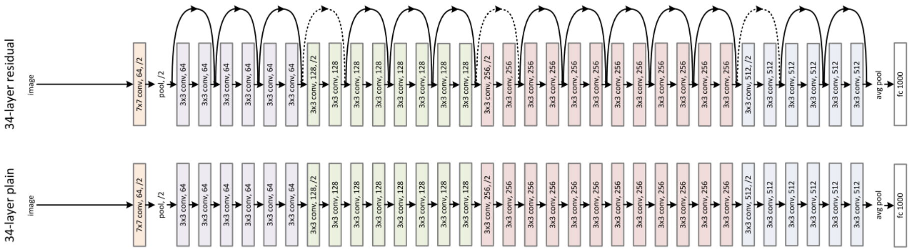


Figure 3.5: **Top:** a residual network with 34 parameter layers. **Bottom:** a plain network with 34 parameter layers [27]

EfficientNet

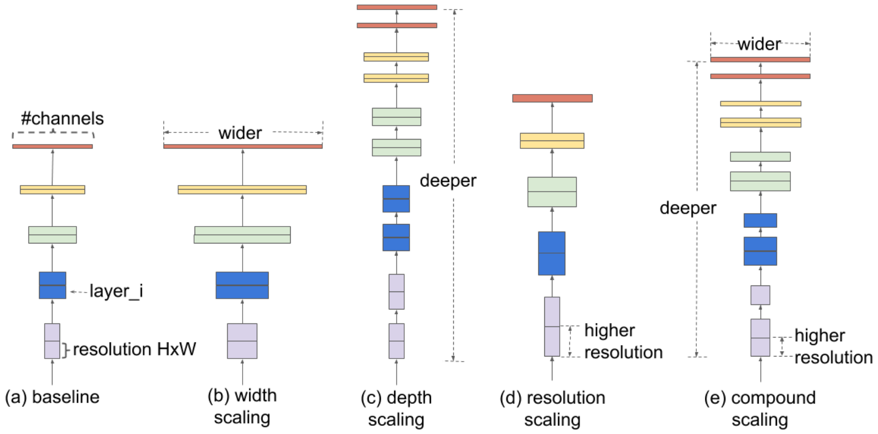


Figure 3.6: Different ways of scaling a CNN [2]

There are three main methods of scaling up a CNN to make it bigger, which are displayed in Figure 3.6. Width scaling, which means adding more feature maps to each layer. Depth scaling, which means adding more layers to the CNN architecture. And lastly, resolution scaling, which means scaling up the input image size.

While VGG16 and ResNet34 mainly focus on depth, EfficientNet tries to scale all dimensions uniformly using a compound coefficient. While it would be amazing to increase the depth, width, and resolution of a network by a substantial amount to increase accuracy, most CNN architectures are limited by how much space they can occupy. This is mainly based on the amount of RAM available, but the computing time will also get increasingly large, so EfficientNet tries to scale up all of these parameters efficiently. This way, the network will get the most performance out of a given resource budget.

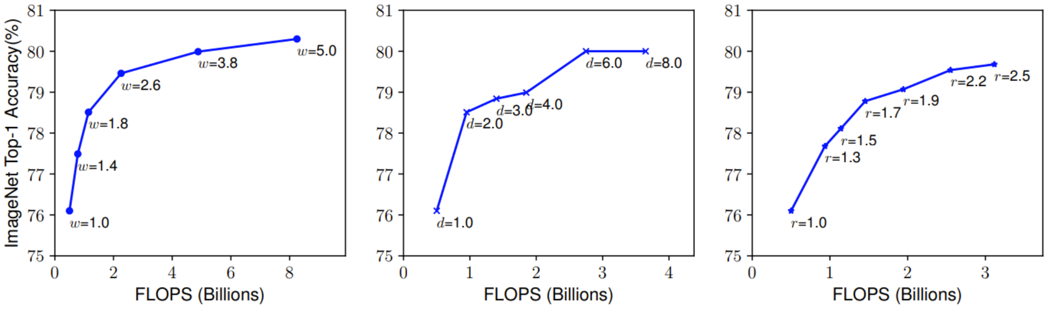


Figure 3.7: Scaling up along a single dimension. **Left:** Width (w). **Middle:** Depth (d). **Right:** Resolution (r) [63]

Currently, scaling up CNNs is not very well understood. Most network scaling is being done arbitrarily, such as randomly adding more layers, randomly editing convolution sizes, and randomly increasing the input image size required. This method of scaling networks wastes a lot of time trying to run many different experiments to see what works the best. Carefully balancing width, depth and resolution is a critical task, as it intuitively makes sense that as the input image increases, the network needs more layers and channels to capture the more fine-grained patterns on the bigger image[55]. EfficientNet managed to complete this task by finding an effective compound scaling method that enables scaling up CNNs to any target resource constraint in a much more principled way[63].

3.2 Skin cancer classification using CNNs

Using Artificial intelligence to diagnose different medical issues has been an up-and-coming focus area for the past few years. Medical diagnosis has historically had some margin of human error due to how challenging it can be to differentiate between different diseases. This is why a lot of research is trying to utilize deep learning to aid the classification process, saving many human lives. There is also a severe lack of doctors globally, which leads to very long wait times, which was highly noticeable during the Covid-19 pandemic. The pandemic opened many people's eyes to how important it is to increase our medical capacity. Finding new artificial intelligence solutions to speed up many medical processes is one way this can be achieved.

In the past few years, there have been published many papers on things like chest X-ray classification looking to find pneumonia [73], breast cancer diagnosis using convolutional networks[22], covid-19 detection[57] and many more. When it comes to skin cancer classification, some research has been done on the topic in the past few years, but one of the many challenges has been the lack of datasets containing images of skin lesions.

ISIC (The International Skin Imaging Collaboration) has worked hard to publish some datasets with dermoscopic images to combat this issue. They have been using these datasets to host challenges where people try to achieve as high accuracy as possible. They have published five different challenges: one in 2016, 2017, 2018, 2019, and 2020.

Challenge	**Number of training images**
ISIC 2016 Challenge	900
ISIC 2017 Challenge	2000
ISIC 2018 Challenge	2594
ISIC 2019 Challenge	25,331
ISIC 2020 Challenge	33,126

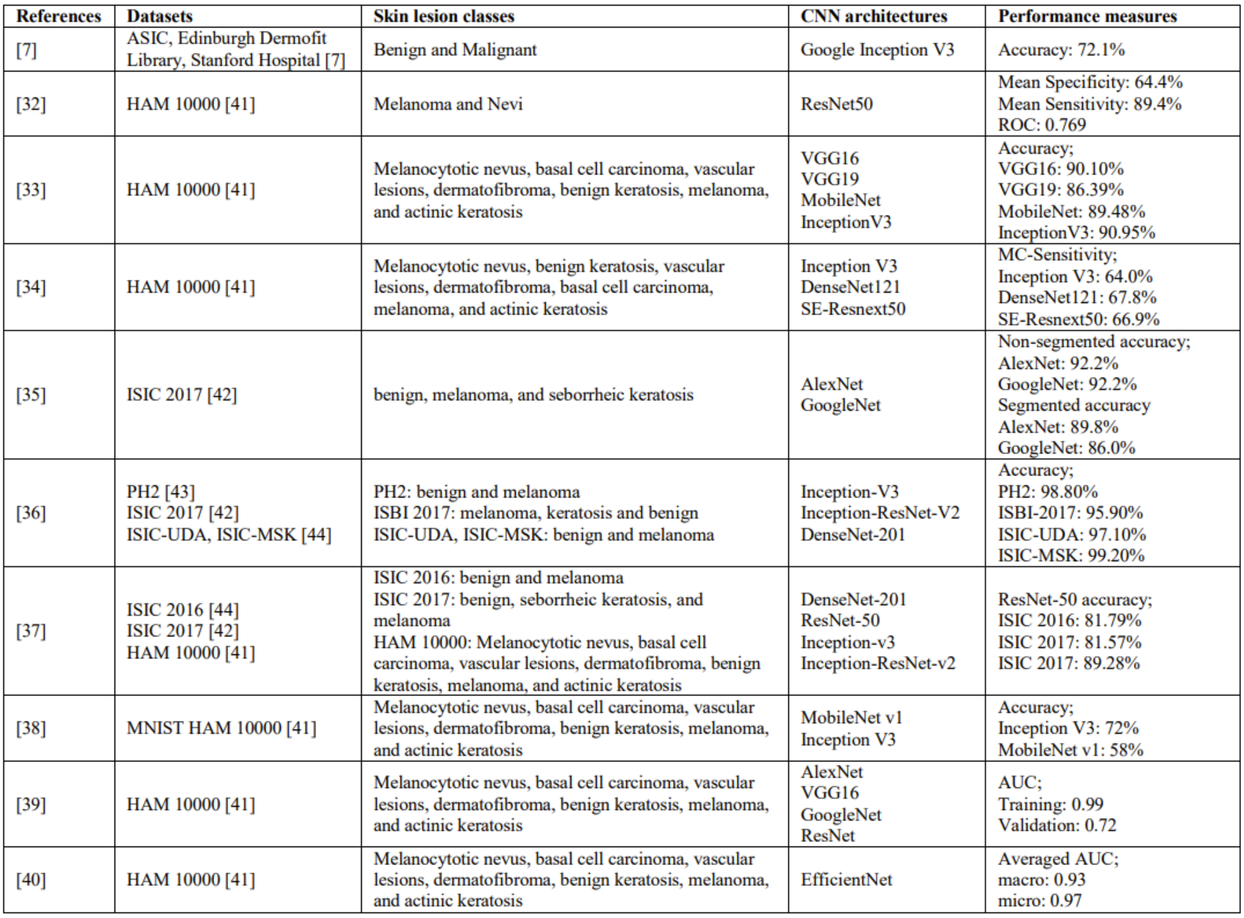
Table 3.1: ISIC challenges

Early on, the number of images was extremely low, which means that it was tough for a machine to learn all of the different features of each disease. Over the past years, they have obtained more and more images, and the last challenge they posted contained as many as 33 thousand images. This challenge is quite popular, and the last leaderboard shows that they have 64 submissions from different teams that are working on it. The best performing model has been achieved by a group of students from Hamburg University of Technology, and they managed to achieve as high as 92% accuracy[33].

Due to there already being extensive research on the ISIC dataset, I chose to focus on a different dataset that also contains images of skin cancer. The dataset chosen is HAM10000, containing 10000 images of seven different skin diseases. This dataset is also quite popular, being one of the most popular datasets for classification over at kaggle.com, a machine learning and data science community that hosts many different datasets.

State-of-the-art accuracies on the HAM10000 seem to be quite hard to find due to many of the top papers combining multiple datasets to achieve as many images to train on as possible. The most common combination is by putting HAM10000 together with the ISIC dataset, as they have similar diseases in their respective datasets. For example, Hemanth Nadipineni, in his paper "Method to Classify Skin Lesions using Dermoscopic images[41]," managed to achieve a classification accuracy of 88.6% after combining the two datasets and using image segmentation to focus on the region of interest.

Some people have done extensive literature reviews where they have compiled all of the best performing papers to this date. Marzuraikah Mohd Stofa Et al. did a lot of research finding all of the current state-of-the-art methods when it comes to classifying skin lesions in late 2021, and the results were published in a paper called "Skin Lesions Classification and Segmentation: A Review[62]." The following table showcased in Figure 3.8 is extracted from their paper, and the table shows a comparison of the recent methods for skin lesion classification using the deep CNN methods. All credit for the table goes to them.



References	Datasets	Skin lesion classes	CNN architectures	Performance measures
[7]	ASIC, Edinburgh Dermofit Library, Stanford Hospital [7]	Benign and Malignant	Google Inception V3	Accuracy: 72.1%
[32]	HAM 10000 [41]	Melanoma and Nevi	ResNet50	Mean Specificity: 64.4% Mean Sensitivity: 89.4% ROC: 0.769
[33]	HAM 10000 [41]	Melanocytotic nevus, basal cell carcinoma, vascular lesions, dermatofibroma, benign keratosis, melanoma, and actinic keratosis	VGG16 VGG19 MobileNet InceptionV3	Accuracy; VGG16: 90.10% VGG19: 86.39% MobileNet: 89.48% InceptionV3: 90.95%
[34]	HAM 10000 [41]	Melanocytotic nevus, benign keratosis, vascular lesions, dermatofibroma, basal cell carcinoma, melanoma, and actinic keratosis	Inception V3 DenseNet121 SE-Resnext50	MC-Sensitivity; Inception V3: 64.0% DenseNet121: 67.8% SE-Resnext50: 66.9%
[35]	ISIC 2017 [42]	benign, melanoma, and seborrheic keratosis	AlexNet GoogleNet	Non-segmented accuracy; AlexNet: 92.2% GoogleNet: 92.2% Segmented accuracy AlexNet: 89.8% GoogleNet: 86.0%
[36]	PH2 [43] ISIC 2017 [42] ISIC-UDA, ISIC-MSK [44]	PH2: benign and melanoma ISBI 2017: melanoma, keratosis and benign ISIC-UDA, ISIC-MSK: benign and melanoma	Inception-V3 Inception-ResNet-V2 DenseNet-201	Accuracy; PH2: 98.80% ISBI-2017: 95.90% ISIC-UDA: 97.10% ISIC-MSK: 99.20%
[37]	ISIC 2016 [44] ISIC 2017 [42] HAM 10000 [41]	ISIC 2016: benign and melanoma ISIC 2017: benign, seborrheic keratosis, and melanoma HAM 10000: Melanocytotic nevus, basal cell carcinoma, vascular lesions, dermatofibroma, benign keratosis, melanoma, and actinic keratosis	DenseNet-201 ResNet-50 Inception-v3 Inception-ResNet-v2	ResNet-50 accuracy; ISIC 2016: 81.79% ISIC 2017: 81.57% ISIC 2017: 89.28%
[38]	MNIST HAM 10000 [41]	Melanocytotic nevus, basal cell carcinoma, vascular lesions, dermatofibroma, benign keratosis, melanoma, and actinic keratosis	MobileNet v1 Inception V3	Accuracy; Inception V3: 72% MobileNet v1: 58%
[39]	HAM 10000 [41]	Melanocytotic nevus, basal cell carcinoma, vascular lesions, dermatofibroma, benign keratosis, melanoma, and actinic keratosis	AlexNet VGG16 GoogleNet ResNet	AUC; Training: 0.99 Validation: 0.72
[40]	HAM 10000 [41]	Melanocytotic nevus, basal cell carcinoma, vascular lesions, dermatofibroma, benign keratosis, melanoma, and actinic keratosis	EfficientNet	Averaged AUC; macro: 0.93 micro: 0.97

Figure 3.8: Comparison of recent CNN methods used for skin lesion detection[62]

Chapter 4

Method

4.1 The task at hand

The main goal of this thesis is to create an artificial network capable of classifying between seven different skin diseases based on images of skin lesions. To accomplish this goal, multiple convolutional networks will be trained on a dataset called "HAM10000," which contains ten thousand images of different skin diseases. These models will be trained with several different state-of-the-art architectures and different image augmentation in an attempt to find out what the best performing architecture and settings are.

The first thing I had to do when completing this task was to do a lot of research on how convolutional neural networks functions so I would gain knowledge of what the best practices are. The information I gained from my research was beneficial when I was making decisions on how to complete the task. This chapter will focus on the choices that were made when trying to accomplish the best accuracy possible.

This chapter will explain why the dataset was chosen, what steps were taken to combat the dataset's class imbalances, and why were VGG16, ResNet34, and EfficientNet_B4 were chosen as CNN architectures.

- The dataset will be discussed in chapter 4.2
- Different ways to deal with class imbalances will be discussed in chapter 4.3
- Image augmentation will be discussed in chapter 4.3.2
- The different CNN architectures used will be discussed in chapter 4.4

4.2 Dataset

When working with images and data analysis, it is essential to use good data since the data is often what determines how good the results will be. Because the network trains and learns based on the data, it naturally follows that if the data contains different biases or insufficient training data, the network will also learn these biases. Garbage in, garbage out is a common concept in computer science, as it is a good way of reminding ourselves that a computer only processes what it is given[15].

While there are many different datasets published for free on the internet, the dataset chosen for this project is the "Humans Against Machine" (HAM10000) dataset[68]. The HAM10000 dataset is a large collection of dermoscopic images of common pigmented skin lesions, and as the name suggests, the dataset consists of 10 000 images. It was initially released to be used in the ISIC 2018 challenge, which was designed to help participants develop artificial networks to automate the diagnosis of melanoma based on dermoscopic images[13].

One of the drawbacks to this dataset is that the number of images of the different skin diseases is very imbalanced. Because some skin diseases are much more common than others, there are many more examples that can be used for training compared to different diseases that are much rarer. In this example, the HAM10000 dataset is very imbalanced between the classes, where the largest class, "melanocytic nevi (nv)" consists of 6700 images. In comparison, the smallest class "dermatofibroma (df)," only consists of 115 images. A breakdown of how many images are in each class is as follows:

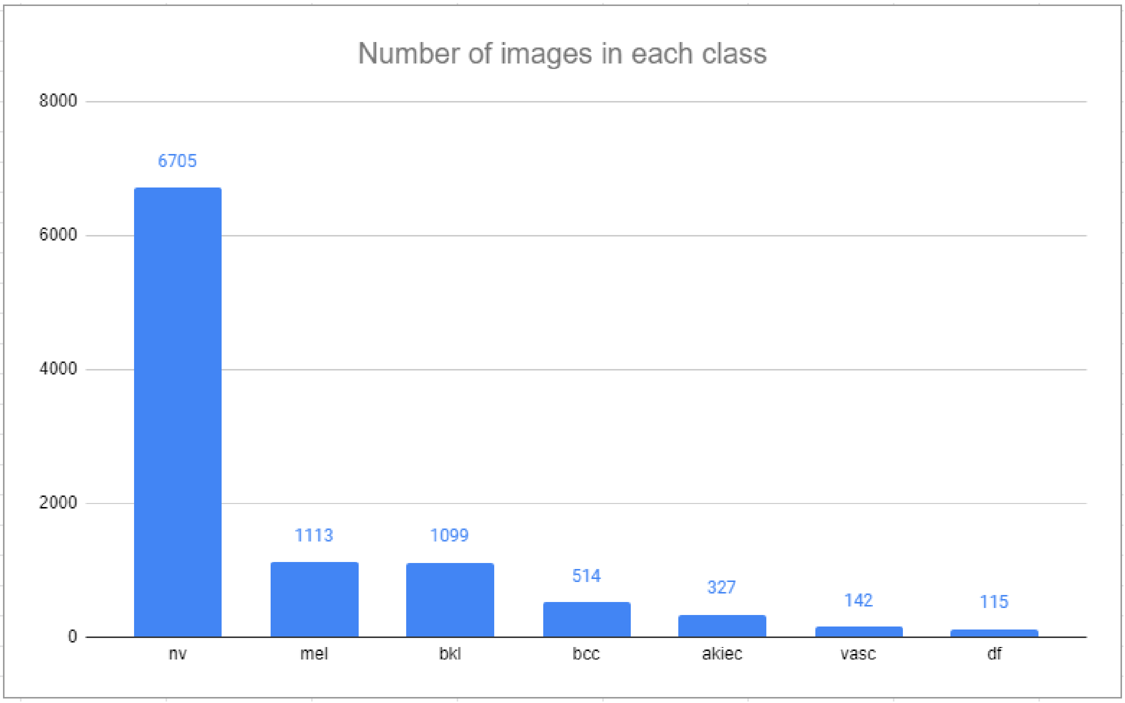


Figure 4.1: Number of images in each class

During real-world examples where the differences are this extreme, it becomes clear that getting a higher accuracy in the NV class is more important than getting a higher accuracy in the DF class when it comes to total accuracy. The network will also highly favor learning the NV characteristics during training because it has so much more data to learn from rather than the DF characteristics. There is a severe lack of images for it to learn how the symptoms can show up when it comes to the smaller classes.

4.2.1 Splitting the data into train/validation/test sets

One of the first things to do when working with a dataset is to split it up into three different parts: One part of the images should be used as a training set the artificial network uses to learn the different weights. Another part of the images should be used as a validation set so the artificial network can test its performance during training. The last part of the images should be used as a test set that the network tries to classify at the end, where it determines how accurate the results from all the training were. It is very important that the network never sees any of the images from the test set while it is training, as those images are supposed to be used to test how the classification will work when evaluating new images that have never been seen before.

There are multiple ways to split up a dataset, and in this project, I have mainly tested three different methods.

The first method was to split the validation and test set evenly between the different classes. This was done by randomly assigning 20 random images from each class for validation and 20 random images from each class to be used for testing. Doing so can give accurate testing to see how the network performs in the different classes. However, one of the drawbacks to splitting it up like this is that the actual proportions of the diseases will not show up, and the training data is learning different weights based on different proportions.

The second way to split the images into train, validation, and test set is to split them at a specific ratio, for example, 80% train, 10% validation, and 10% test. This is the most common method of splitting the data, as doing so keeps the proportions of each class in the validation and test set, which gives a better idea of how the classification accuracy will perform on new images. Each disease has very different rarities of occurring, so when new images are sent to the network to be classified, it is natural that the images keep the same proportions as the given dataset.

Individual users who want to see if they need to check up on their skin lesion care more about knowing if it is dangerous rather than knowing exactly which type of skin lesion it is. Because the models can be used in a telehealth way where clients take an image of a skin lesion themselves to see if they need to see a dermatologist, I also trained a network to classify between just two different classes: dangerous and benign. This was done by grouping the different classes based on their severity, with BKL, DF, NV, and VASC being classified as benign and AKIEC, BCC, and MEL as dangerous. Splitting the data into two categories helps the neural network learn more general characteristics with more data for each class. The results are still beneficial as they can tell the user if they should see a doctor or if the skin lesion is harmless and that they do not need to seek additional help.

4.3 combating the dataset imbalances

Because of the nature of different diseases having different rarities, there is no surprise that the dataset contains more images of common diseases and fewer images of rarer diseases. The uneven distribution of the classes means that the artificial network will learn much more from the big classes than they learn from smaller classes, giving it a bias that should not exist when looking to classify a single image. While it is true that having a bias towards the largest class may increase the overall performance and accuracy of a network, it should not be taken into account when looking to diagnose diseases on a case-by-case basis. Suppose an image looks like it may contain cancer cells. In that case, it must get classified as such instead of being influenced by the "non-cancer" bias of the dataset, which can lead to serious misclassifications.

Data imbalance is one of the most common problems when working with deep learning based on real-life events. The reason being is that when multiple things can happen, the distribution between them very rarely follows a uniform distribution, and oftentimes the goal of neural networks is to detect the edge case scenarios. Some examples can be in medical imaging, where the number of healthy individuals is much larger than the number of sick individuals (such as the dataset used in this thesis). In fraud detection, where the number of genuine transactions is much larger than the number of fraudulent transactions. Or when collecting images of different animal species where, depending on the method of acquiring images, some species are much more common than others[37].

4.3.1 Ways to deal with dataset imbalances

There are multiple ways to deal with imbalances in datasets. The two most common methods are by adjusting the dataset itself and the other is to adjust the learning algorithms with different methods, such as adding weights to the different classes[37].

Adjusting the data sets

When it comes to adjusting the data set, the two most common methods are called oversampling and undersampling.

- **Undersampling:** Delete random examples from the majority classes to achieve the same amount of images in each class
- **Oversampling:** Duplicate random examples from the minority classes to achieve the same amount of images in each class

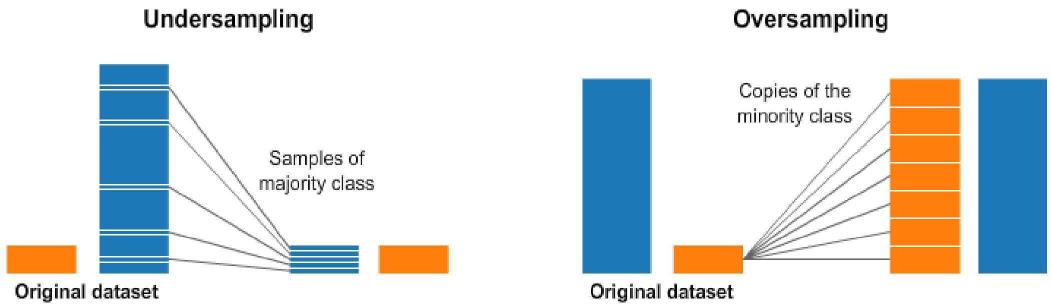


Figure 4.2: Undersampling and oversampling example[37]

Oversampling is the most used method as it keeps all of the available data in the dataset. By randomly duplicating images, the network does not add any new information, but it learns that doing misclassifications on the smaller classes is just as bad as doing misclassifications on the bigger classes due to the wrong guesses giving a higher loss value in the loss function every time the duplicated image shows up. If an image is duplicated three times and the networks always classify it as something else, it will get punished three times more in the loss function than if it just showed up once.

On the other hand, undersampling removes many images from the dataset, which can result in the network removing images that contain a lot of valuable information. Removing these images can result in the network not learning different patterns and characteristics that it would otherwise have learned from the removed images, causing the network to learn less than it would if it kept them to train. However, even though the networks have fewer images to train on, undersampling can improve performance compared to not using any data augmentation due to achieving a more balanced dataset. The balanced dataset stops the network from learning as many biases as it would with a very imbalanced dataset.

Oversampling usually results in the best accuracy at the cost of training speed, given that the network trains with many more images. With extensive datasets, undersampling can be favored if it is important to lower the training time.

Adjusting the algorithms

Another method to combat class imbalance is to adjust the learning algorithms to account for the differences in classes. One way is to modify the loss function to punish misclassifications of the smaller classes harsher than misclassifications of the bigger classes, which can make the network focus more on getting those correct. This can be done in cases where the network starts to completely ignore the smaller class if it is small enough, as the difference it makes on the total loss value by getting them wrong is minuscule if there are not enough images that punish it when getting them wrong.

It is also possible to adjust the threshold levels needed to classify different classes instead of just taking the highest value predicted by using a function like argmax. Doing so can increase the confidence needed before it gets classified as one of the majority classes while simultaneously lowering the confidence needed before it gets classified as one of the minority classes. This can help with the biases learned during training. The classifier often favors the majority classes more than the minority classes in cases where it is unsure what the classification should be.

During this thesis, I have mainly focused on using oversampling due to the advantages of achieving higher accuracy. I have also tested the usage of thresholding to achieve higher accuracy in the later stages.

4.3.2 Image augmentation

Another way to increase the number of images that can be used during training is image augmentation. Image augmentation is the act of doing minor augmentations to the images to increase the number of images. One example can be to flip all the images horizontally, which suddenly doubles the number of training images without substantially modifying the images.

Pytorch contains many common image transformations which are available in the torchvision.transforms module[48], but there are many more libraries that can do this kind of image augmentation, such as a python library called "imgaug[24]."

4.4 Building the CNN architecture

One of the first things I did was to try three different state-of-the-art CNN architectures: Resnet34, EfficientNet_b4, and VGG16.

Before creating the different CNN architectures, I figured that it would be smart to create a shell that could be used for all the different methods. The only thing that needed to be changed between them was the model and the different hyperparameters.

To do this, I started by creating a jupyter notebook that included all of the libraries needed to run convolutional networks using PyTorch, which is a machine learning framework mainly developed by Facebooks AI research lab. The setup was focused on being easily modifiable, meaning it would be possible to reuse most of the work through the different CNN architectures without recreating the entire setup. A flowchart of how the setup ended up looking can be seen in the flowchart showcased in Figure 4.3.

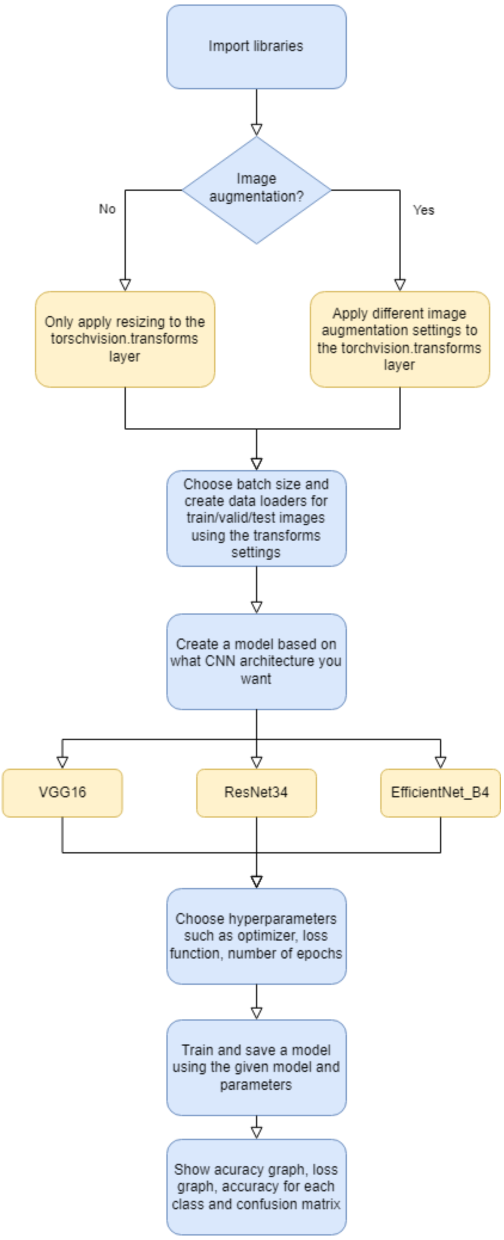

Import libraries
Image augmentation?
No
Yes
Only apply resizing to the torschvision.transforms layer
Apply different image augmentation settings to the torchvision.transforms layer
Choose batch size and create data loaders for train/valid/test images using the transforms settings
Create a model based on what CNN architecture you want
VGG16
ResNet34
EfficientNet_B4
Choose hyperparameters such as optimizer, loss function, number of epochs
Train and save a model using the given model and parameters
Show acuracy graph, loss graph, accuracy for each class and confusion matrix

Figure 4.3: Flowchart of how the models were built

4.4.1 Building the models

All of the different CNN architectures used in this project were discussed in state-of-the-art. The first architecture used was VGG16, which was explained in 3.1.1. The second architecture used was ResNet34, which was explained in 3.1.1. And lastly, EfficientNet_b4, which was explained in 3.1.1. Because they already discussed them in state-of-the-art, there is no point in repeating everything here. Building these models are quite simple because they are as popular as they are, so there are many different guides to follow. They are set up exactly as described earlier.

These architectures are all state-of-the-art and are very popular in the CNN world, but they were chosen mainly because of the available computing power and GPU memory. The machine running these tests is described in Table 4.1, and the GPU only has 3GB of memory. Newer graphic cards in the past years have increased in memory, often having 8GB or more. Bigger models such as EfficientNet_B7 would increase the performance but were not runnable on this machine. The only modification that was made was with the VGG16 network due to it being so large and taking too much GPU memory. , The default image size of 224x224 had to be turned down to 112x112, as the network could not run even with batch size set to 1. ResNet34 and EfficientNet_B4 were fine with all of the default parameters.

Type	**Model**
Processor	AMD Ryzen 5 3600X
GPU	Nvidia GTX 1060-3GB
Memory	16GB DDR4 3200MHz
Operating System	Windows 10

Table 4.1: Computer specs

Chapter 5

Results

Due to testing many different ways of splitting the dataset, the results section will be split into five main parts:

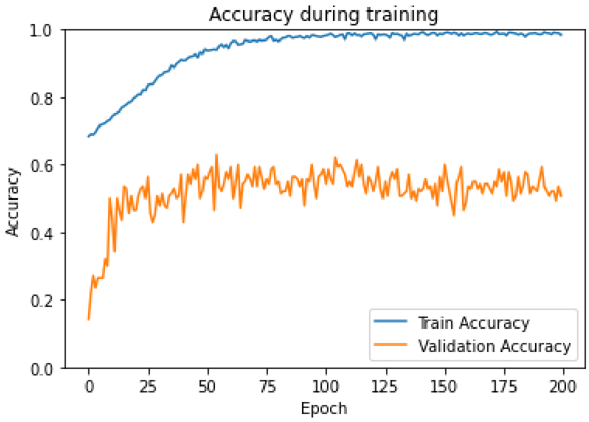
- 20 valid and 20 test images results (5.1)
- 80% train, 10% valid, and 10% test images results (5.2)
- Binary classification between dangerous and benign results (5.3)
- Voting with multiple models results (5.4)
- Using confidence thresholding when classifying (5.5)

5.1 20 valid and 20 test images results

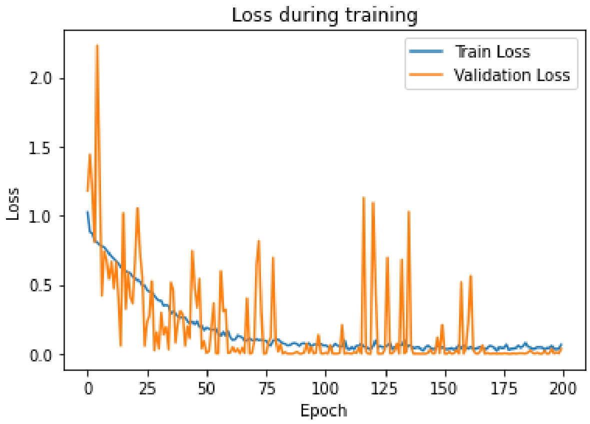
The first tests were run on the dataset, which was split with only 20 validation images and 20 test images, with the rest being used for training.

5.1.1 VGG16

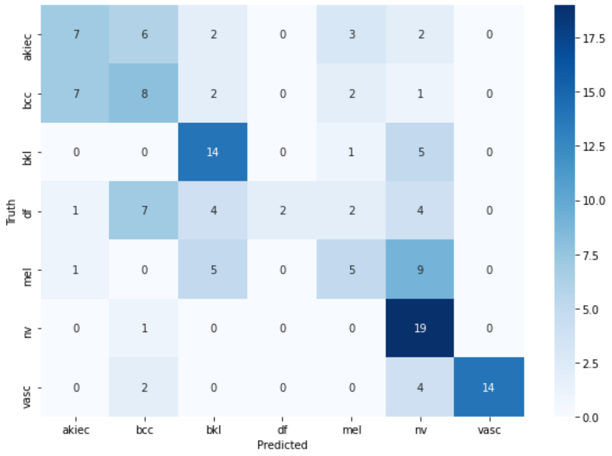
VGG16 test no image augmentation



(a) VGG16_noimgaug - accuracy over time



(b) VGG16_noimgaug - loss over time



	akiec	bcc	bkl	df	mel	nv	vasc
akiec	7	6	2	0	3	2	0
bcc	7	8	2	0	2	1	0
bkl	0	0	14	0	1	5	0
df	1	7	4	2	2	4	0
mel	1	0	5	0	5	9	0
nv	0	1	0	0	0	19	0
vasc	0	2	0	0	0	4	14

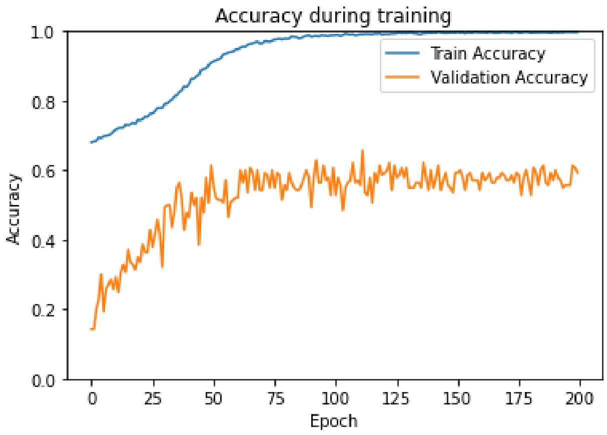
(c) VGG16_noimgaug - confusion matrix

Class	Accuracy	Count
Actinic keratoses (akiec)	35.0%	7/20
Basal cell carcinoma (bcc)	40.0%	8/20
benign keratosis-like lesions (bkl)	70.0%	14/20
Dermatofibroma (df)	10.0%	2/20
Melanoma (mel)	25.0%	5/20
Melanocytic nevi (nv)	95.0%	19/20
Vascular lesions (vasc)	70.0%	14/20
Total accuracy	**49.3%**	**69/140**

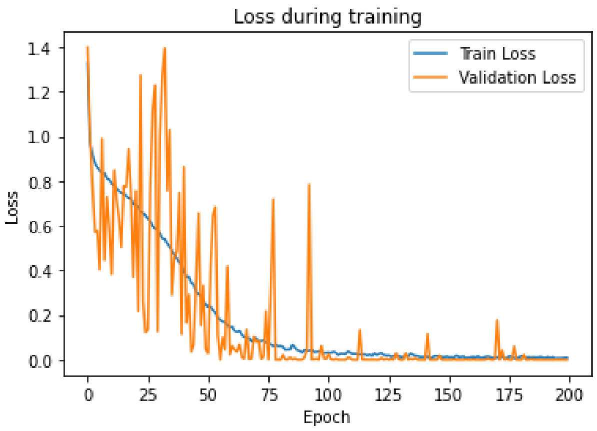
Table 5.1: Accuracy VGG16 no imgaug

5.1.2 ResNet34

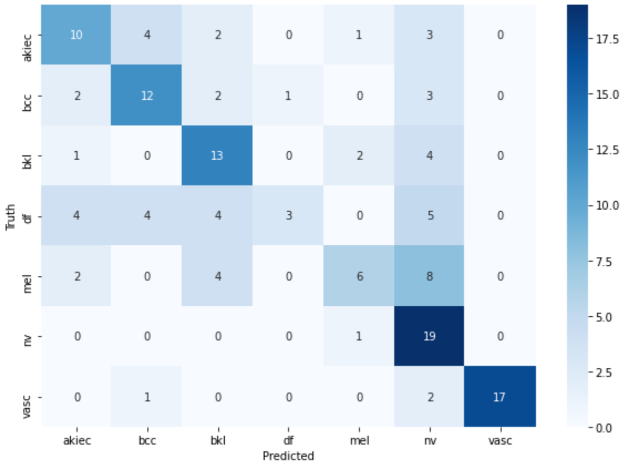
ResNet34 test no image augmentation



(a) ResNet34_noimgaug - accuracy over time



(b) ResNet34_noimgaug - loss over time



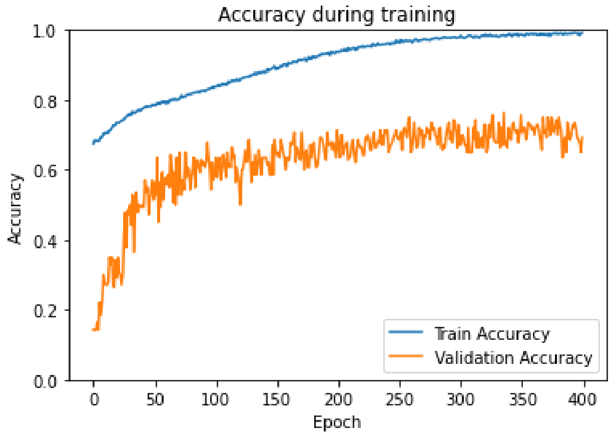
	akiec	bcc	bkl	df	mel	nv	vasc
akiec	10	4	2	0	1	3	0
bcc	2	12	2	1	0	3	0
bkl	1	0	13	0	2	4	0
df	4	4	4	3	0	5	0
mel	2	0	4	0	6	8	0
nv	0	0	0	0	1	19	0
vasc	0	1	0	0	0	2	17

(c) ResNet34_noimgaug - confusion matrix

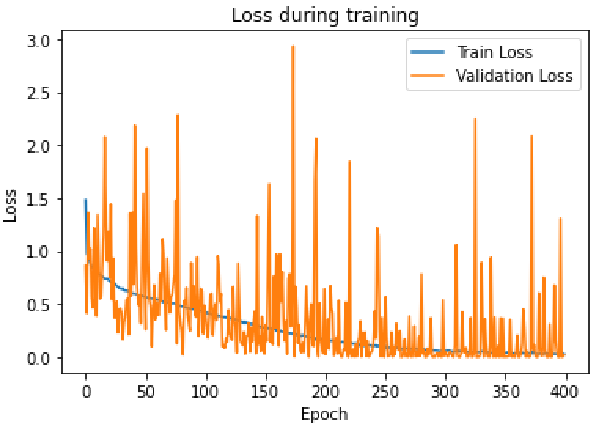
Class	**Accuracy**	**Count**
Actinic keratoses (akiec)	50.0%	10/20
Basal cell carcinoma (bcc)	60.0%	12/20
benign keratosis-like lesions (bkl)	65.0%	13/20
Dermatofibroma (df)	15.0%	3/20
Melanoma (mel)	30.0%	6/20
Melanocytic nevi (nv)	95.0%	19/20
Vascular lesions (vasc)	85.0%	17/20
Total accuracy	**57.1%**	**80/140**

Table 5.2: Accuracy ResNet34 no imgaug

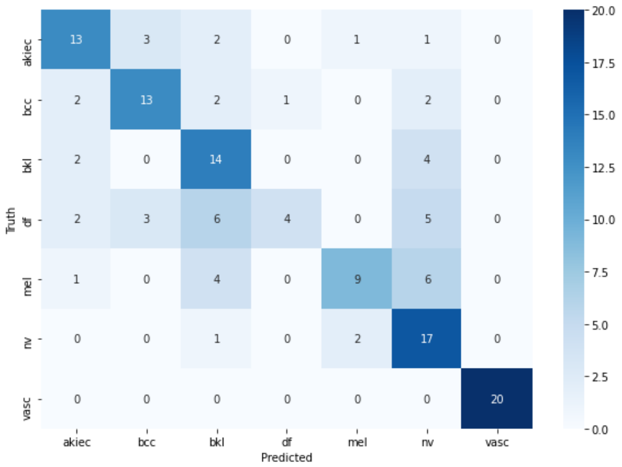
ResNet34 test with image augmentation



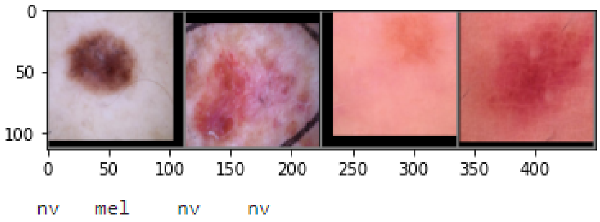
(a) ResNet34_imgaug - accuracy over time



(b) ResNet34_imgaug - loss over time



(c) ResNet34_imgaug - confusion matrix



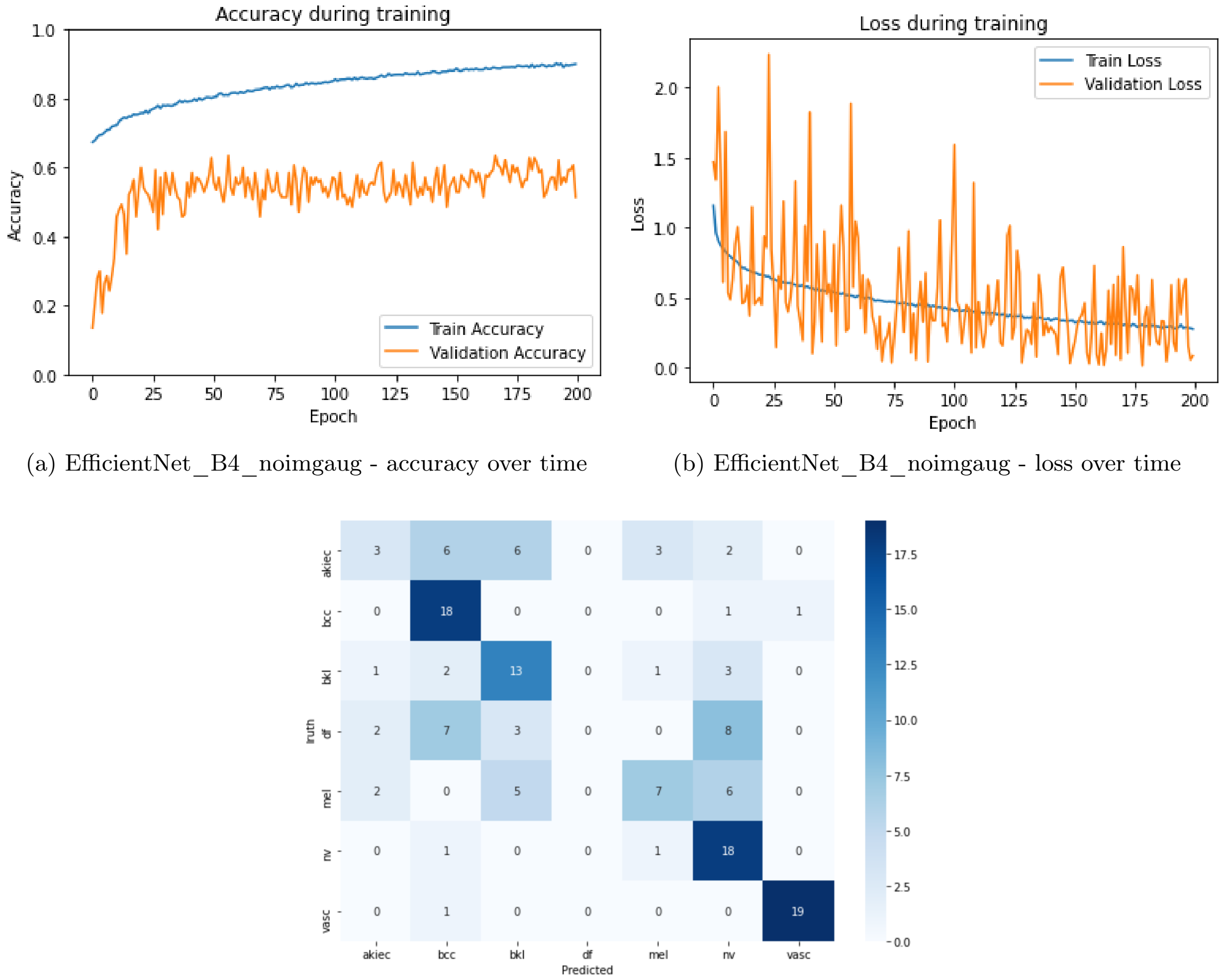
(d) Image augmentation

Class	Accuracy	Count
Actinic keratoses (akiec)	65.0%	13/20
Basal cell carcinoma (bcc)	65.0%	13/20
benign keratosis-like lesions (bkl)	70.0%	14/20
Dermatofibroma (df)	20.0%	4/20
Melanoma (mel)	45.0%	9/20
Melanocytic nevi (nv)	85.0%	17/20
Vascular lesions (vasc)	100.0%	20/20
Total accuracy	**64.3%**	**90/140**

Table 5.3: Accuracy ResNet34 with imgaug

5.1.3 EfficientNet_B4

EfficientNet_B4 test no image augmentation



Truth \ Predicted	akiec	bcc	bkl	df	mel	nv	vasc
akiec	3	6	6	0	3	2	0
bcc	0	18	0	0	0	1	1
bkl	1	2	13	0	1	3	0
df	2	7	3	0	0	8	0
mel	2	0	5	0	7	6	0
nv	0	1	0	0	1	18	0
vasc	0	1	0	0	0	0	19

(a) EfficientNet_B4_noimgaug - accuracy over time

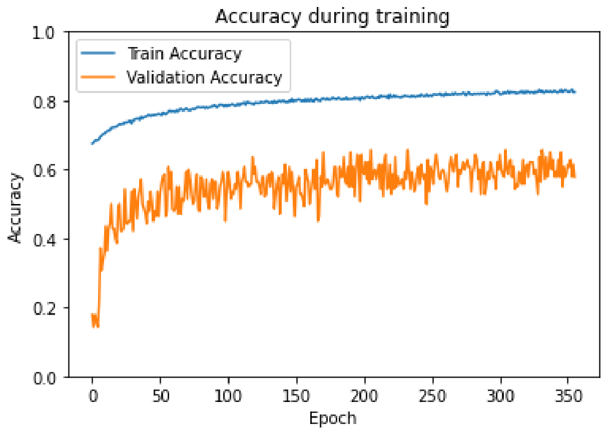
(b) EfficientNet_B4_noimgaug - loss over time

(c) EfficientNet_B4_noimgaug - confusion matrix

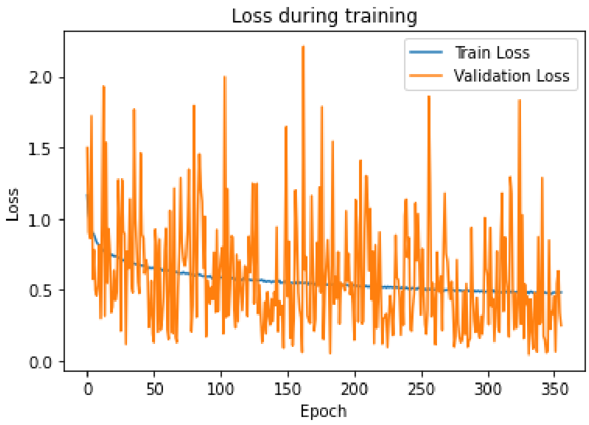
Class	Accuracy	Count
Actinic keratoses (akiec)	15.0%	3/20
Basal cell carcinoma (bcc)	90.0%	18/20
benign keratosis-like lesions (bkl)	65.0%	13/20
Dermatofibroma (df)	0.0%	0/20
Melanoma (mel)	35.0%	7/20
Melanocytic nevi (nv)	90.0%	18/20
Vascular lesions (vasc)	95.0%	19/20
Total accuracy	**55.7%**	**78/140**

Table 5.4: Accuracy EfficientNet_B4 no imgaug

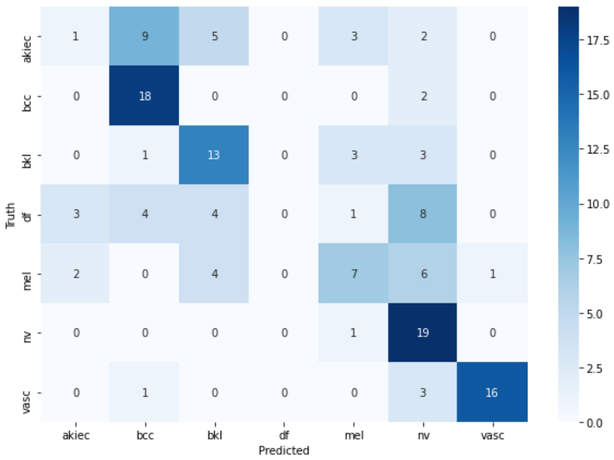
EfficientNet_B4 test with image augmentation



(a) EfficientNet_B4_imgaug - accuracy over time

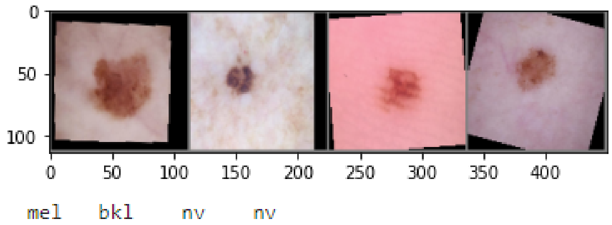


(b) EfficientNet_B4_imgaug - loss over time



Truth \ Predicted	akiec	bcc	bkl	df	mel	nv	vasc
akiec	1	9	5	0	3	2	0
bcc	0	18	0	0	0	2	0
bkl	0	1	13	0	3	3	0
df	3	4	4	0	1	8	0
mel	2	0	4	0	7	6	1
nv	0	0	0	0	1	19	0
vasc	0	1	0	0	0	3	16

(c) EfficientNet_B4_imgaug - confusion matrix



(d) Image augmentation

Class	Accuracy	Count
Actinic keratoses (akiec)	5.0%	1/20
Basal cell carcinoma (bcc)	90.0%	18/20
benign keratosis-like lesions (bkl)	65.0%	13/20
Dermatofibroma (df)	0.0%	0/20
Melanoma (mel)	35.0%	7/20
Melanocytic nevi (nv)	95.0%	19/20
Vascular lesions (vasc)	80.0%	16/20
Total accuracy	**52.9%**	**74/140**

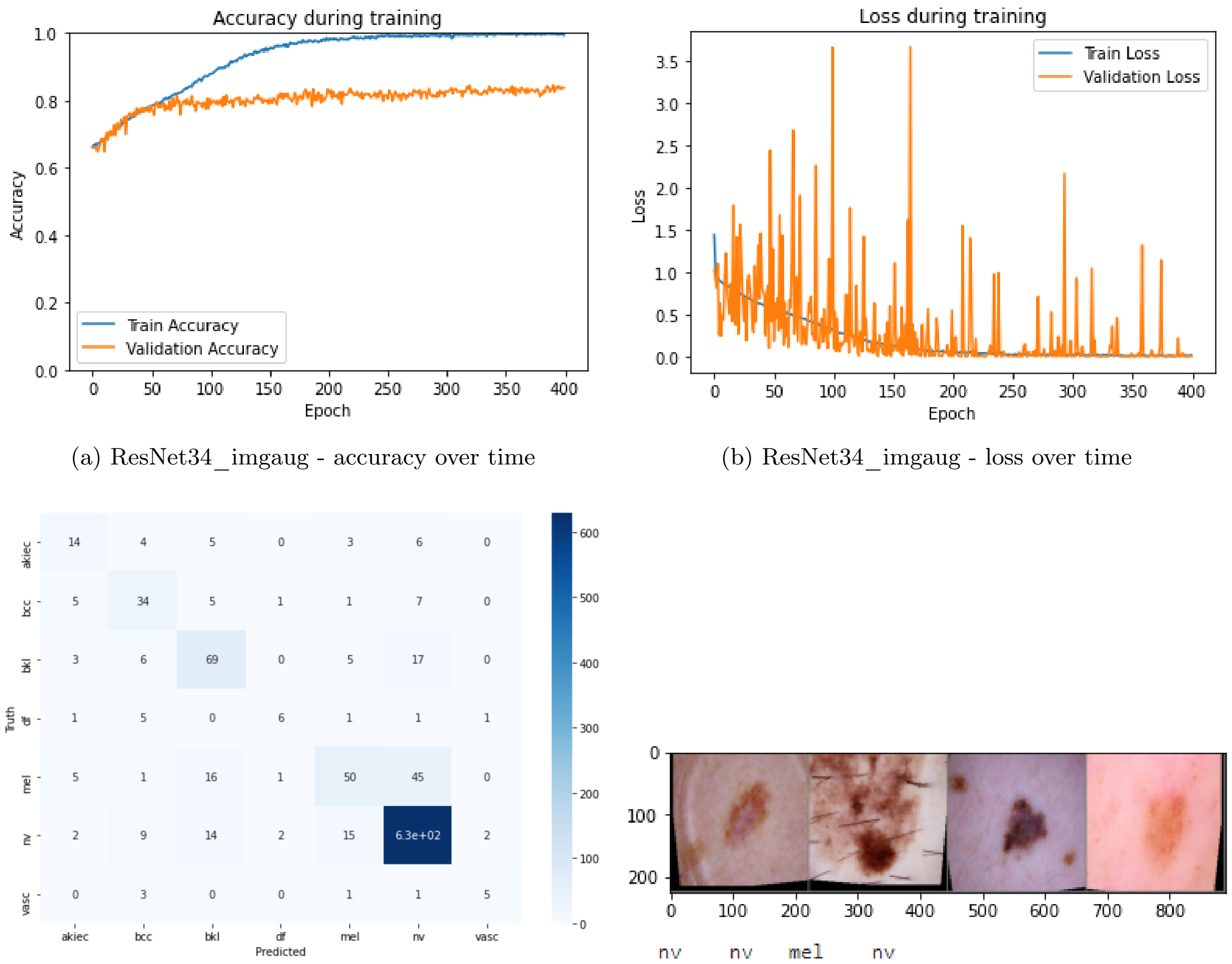
Table 5.5: Accuracy EfficientNet_B4 with imgaug

5.2 80/10/10 split

The next tests were ran on the dataset which was split with 80% of the images to train, 10% of the images used as validation and 10% of the images used for testing.

ResNet34 test with image augmentation

The first test was made with ResNet34 using simple image augmentation such as flips and small rotations.



(a) ResNet34_imgaug - accuracy over time

(b) ResNet34_imgaug - loss over time

(c) ResNet34_imgaug - confusion matrix

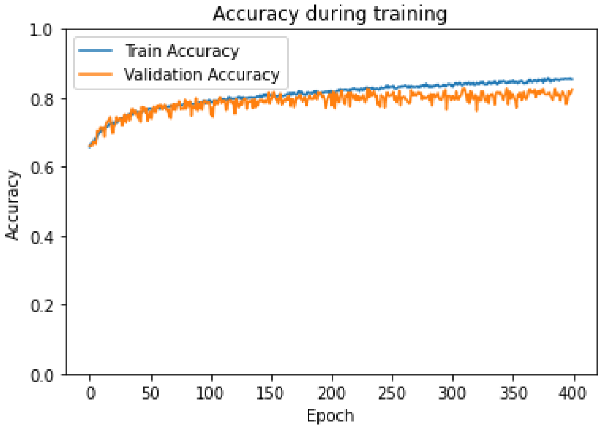
(d) Image augmentation

Class	Accuracy	Count
Actinic keratoses (akiec)	43.8%	14/32
Basal cell carcinoma (bcc)	64.2%	34/53
benign keratosis-like lesions (bkl)	69.0%	69/100
Dermatofibroma (df)	40.0%	6/15
Melanoma (mel)	42.4%	50/118
Melanocytic nevi (nv)	93.5%	629/673
Vascular lesions (vasc)	50.0%	5/10
Total accuracy	**80.62%**	**807/1001**

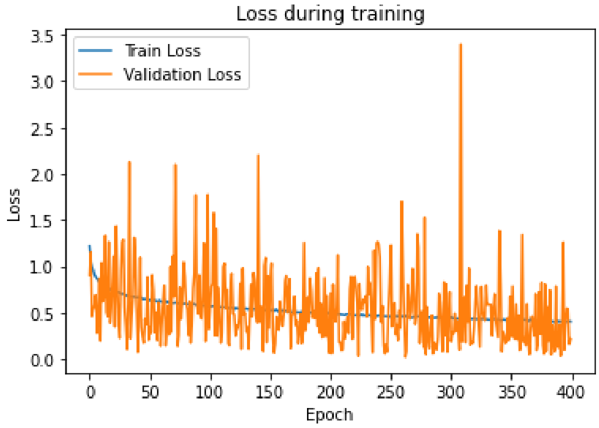
Table 5.6: Accuracy ResNet34 with imgaug

EfficientNet_B4 test with image augmentation

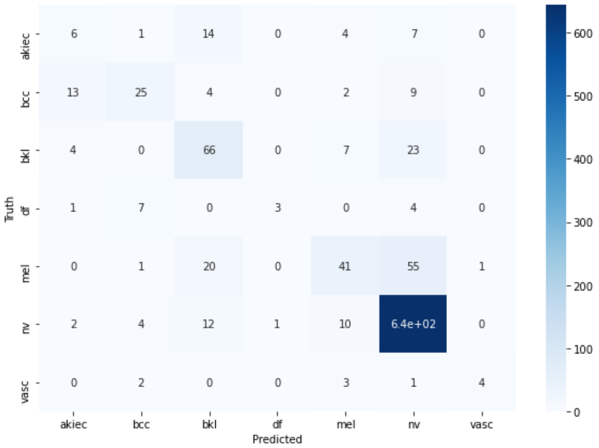
The test with EffientNet used a bit heavier image augmentation. Instead of just using flips and small rotations, it also tried to use ColorJitter and GaussianBlur. The colorjitter effect randomly changes the brightness, saturation, and other properties of an image, and the GaussianBlur randomly added some blur to reduce image noise.



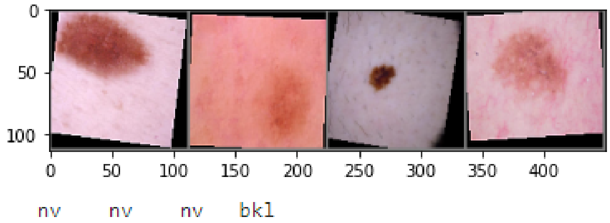
(a) EfficientNet_B4_imgaug - accuracy over time



(b) EfficientNet_B4_imgaug - loss over time



(c) EfficientNet_B4_imgaug - confusion matrix

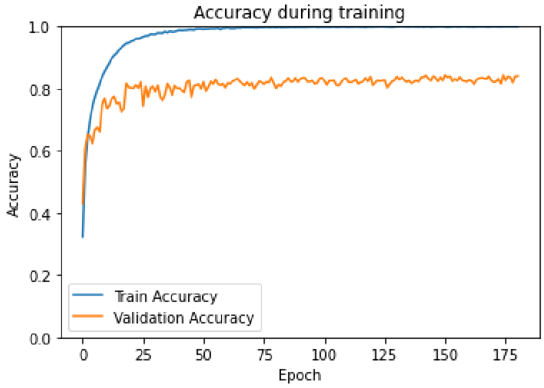


(d) Image augmentation

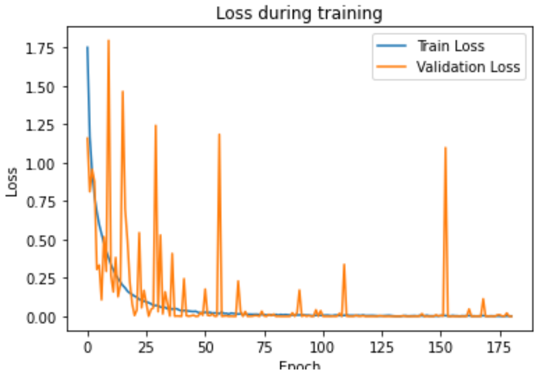
Class	Accuracy	Count
Actinic keratoses (akiec)	18.8%	6/32
Basal cell carcinoma (bcc)	27.2%	25/53
benign keratosis-like lesions (bkl)	66.0%	66/100
Dermatofibroma (df)	20.0%	3/15
Melanoma (mel)	34.7%	41/118
Melanocytic nevi (nv)	95.7%	644/673
Vascular lesions (vasc)	40.0%	4/10
Total accuracy	**78.82%**	**789/1001**

Table 5.7: Accuracy EfficientNet_B4 with imgaug

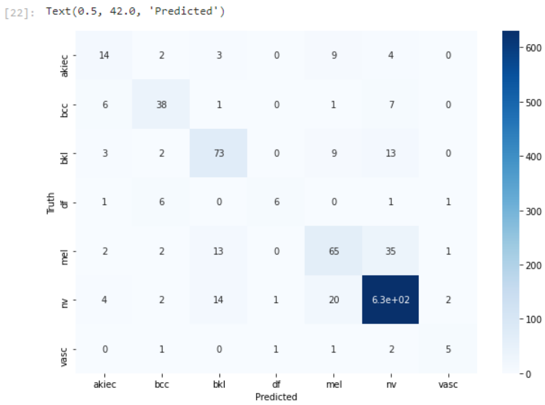
ResNet34 test with image augmentation - balanced dataset with oversampling



(a) ResNet34_imgaug_oversampling - accuracy over time



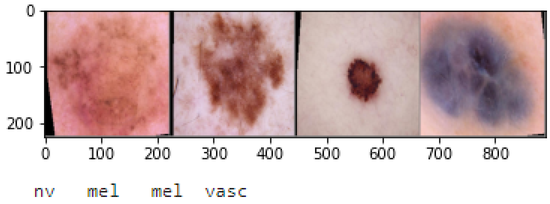
(b) ResNet34_imgaug_oversampling - loss over time



```
[22]: Text(0.5, 42.0, 'Predicted')
```

Truth \ Predicted	akiec	bcc	bkl	df	mel	nv	vasc
akiec	14	2	3	0	9	4	0
bcc	6	38	1	0	1	7	0
bkl	3	2	73	0	9	13	0
df	1	6	0	6	0	1	1
mel	2	2	13	0	65	35	1
nv	4	2	14	1	20	6.3e+02	2
vasc	0	1	0	1	1	2	5

(c) ResNet34_imgaug_oversampling - confusion matrix



(d) Image augmentation

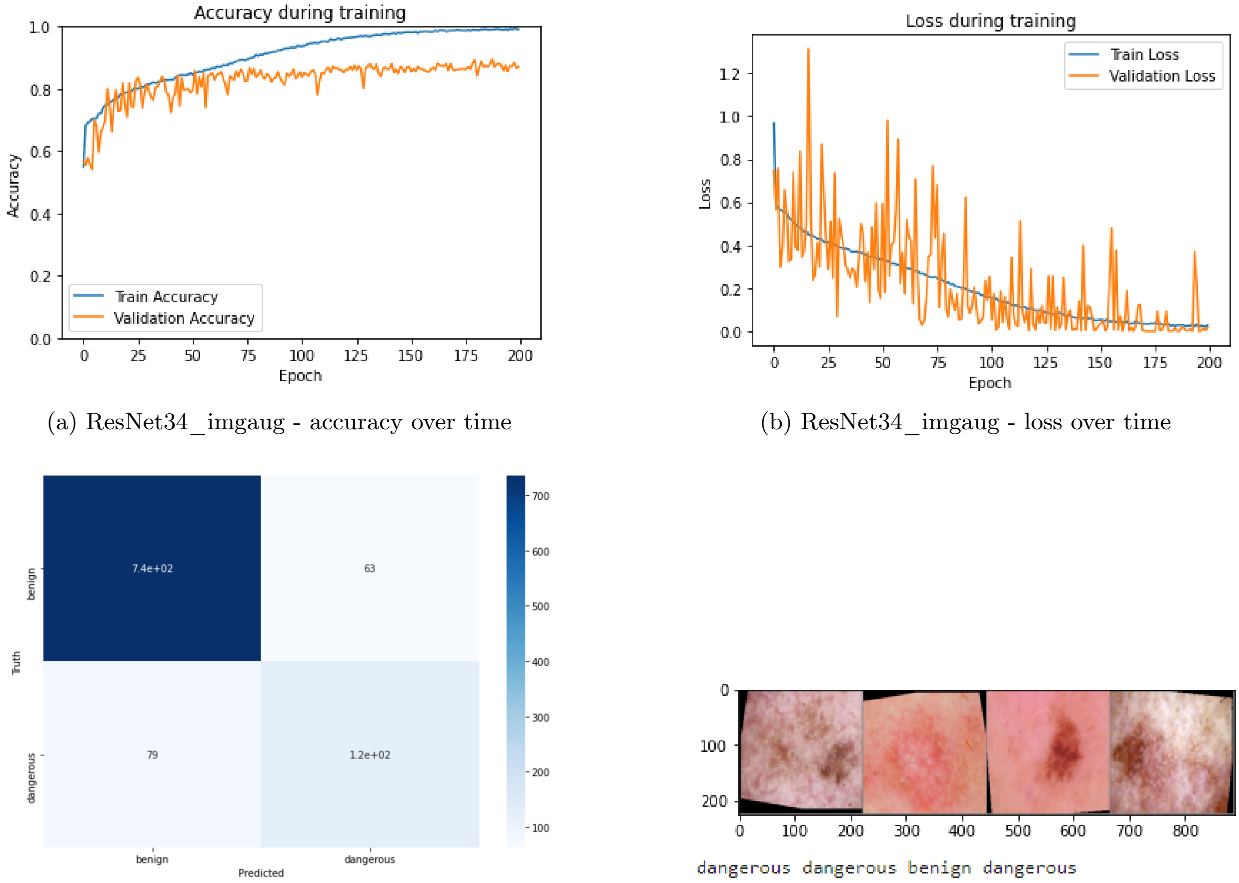
Class	Accuracy	Count
Actinic keratoses (akiec)	43.8%	14/32
Basal cell carcinoma (bcc)	71.7%	38/53
benign keratosis-like lesions (bkl)	73.0%	73/100
Dermatofibroma (df)	40.0%	6/15
Melanoma (mel)	5.1%	65/118
Melanocytic nevi (nv)	93.6%	630/673
Vascular lesions (vasc)	50.0%	5/10
Total accuracy	**83.02%**	**831/1001**

Table 5.8: Accuracy ResNet34 with imgaug and oversampling

5.3 Binary classification between dangerous and benign results

The next test was run on the dataset where the different diseases were split into two different classes based on the severity of the disease. BKL, DF, NV, and VASC were grouped up into a benign class, and AKIEC, BCC, and MEL were grouped into a dangerous class.

ResNet34 test with image augmentation - balanced dataset with oversampling



(a) ResNet34_imgaug - accuracy over time

(b) ResNet34_imgaug - loss over time

(c) ResNet34_imgaug - confusion matrix

(d) Image augmentation

Class	Accuracy	Count
Cancer/dangerous	61.1%	124/203
Benign/safe	92.1%	735/798
Total accuracy	**85.81%**	**859/1001**

Table 5.9: Accuracy ResNet34 with imgaug and oversampling binary classification

5.4 Using multiple models that votes with their prediction results

After training multiple models during the past experiments, I decided to use the four best models achieved during the previous test during the same run where they voted on the classification.

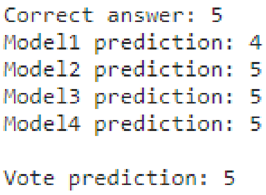
The models used to vote were:

Model	Accuracy
ResNet34 with imgaug and oversampling	83.02%
ResNet34 with image augmentation	80.62%
ResNet34 with different image augmentation	79.83%
EfficientNet_B4 with imgaug	78.82%

Table 5.10: Models used to vote

Plurality voting

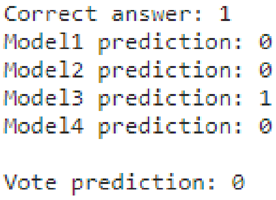
The first way to vote was by making each model predict what the image was and then vote with their best guess. Most of the time, the models predicted the same class due to training on the same dataset. However, because each model had some differences, some of the images resulted in different predictions.



```
Correct answer: 5
Model1 prediction: 4
Model2 prediction: 5
Model3 prediction: 5
Model4 prediction: 5

Vote prediction: 5
```

Figure 5.1: Models voting correct



```
Correct answer: 1
Model1 prediction: 0
Model2 prediction: 0
Model3 prediction: 1
Model4 prediction: 0

Vote prediction: 0
```

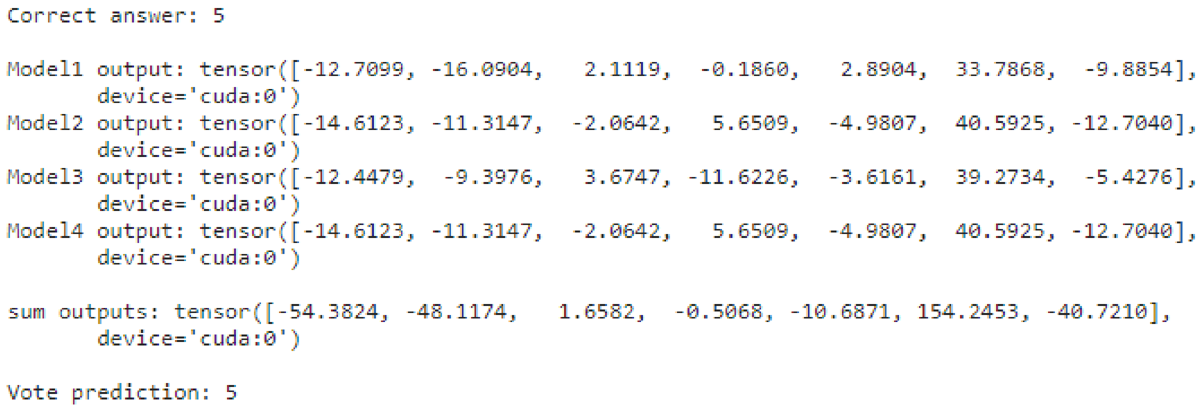
Figure 5.2: Models voting wrong

Class	Accuracy	Count
Actinic keratoses (akiec)	46.9%	15/32
Basal cell carcinoma (bcc)	75.5%	40/53
benign keratosis-like lesions (bkl)	73.0%	73/100
Dermatofibroma (df)	46.7%	7/15
Melanoma (mel)	53.4%	63/118
Melanocytic nevi (nv)	94.35%	635/673
Vascular lesions (vasc)	60.0%	6/10
Total accuracy	**83.84%**	**839/1001**

Table 5.11: Accuracy Plurality voting

Adding the outputs of the models

The second way of voting was intended to factor in the confidence of each model when voting. In the previous test, each model did the argmax function to choose their prediction, which was then used to vote. This time all of the output of the models were added together before doing the argmax function to make a prediction. This way, if one model were very confident and had a large number, it would have a more significant effect on the result than another uncertain model, sending in a low number. Instead of each model making its prediction, a prediction was made based on the sum of outputs. An image showing how this works can be seen in Figure 5.3.



```
Correct answer: 5

Model1 output: tensor([-12.7099, -16.0904,   2.1119,  -0.1860,   2.8904,  33.7868,  -9.8854],
       device='cuda:0')
Model2 output: tensor([-14.6123, -11.3147,  -2.0642,   5.6509,  -4.9807,  40.5925, -12.7040],
       device='cuda:0')
Model3 output: tensor([-12.4479,  -9.3976,   3.6747, -11.6226,  -3.6161,  39.2734,  -5.4276],
       device='cuda:0')
Model4 output: tensor([-14.6123, -11.3147,  -2.0642,   5.6509,  -4.9807,  40.5925, -12.7040],
       device='cuda:0')

sum outputs: tensor([-54.3824, -48.1174,   1.6582,  -0.5068, -10.6871, 154.2453, -40.7210],
       device='cuda:0')

Vote prediction: 5
```

Figure 5.3: Models being added up before doing argmax function to choose

Class	Accuracy	Count
Actinic keratoses (akiec)	50.0%	16/32
Basal cell carcinoma (bcc)	71.7%	38/53
benign keratosis-like lesions (bkl)	71.0%	71/100
Dermatofibroma (df)	40.0%	6/15
Melanoma (mel)	58.5%	69/118
Melanocytic nevi (nv)	95.39%	642/673
Vascular lesions (vasc)	50.0%	5/10
Total accuracy	**84.62%**	**847/1001**

Table 5.12: Accuracy adding outputs of the models

5.5 Using different thresholds when doing the classification

One of the goals is to create a model that tells a patient if they need to contact a doctor or if they are safe. One advantage when creating a model like this is that we also can say "unsure, go see a doctor for further testing" in cases where the machine is unsure, allowing it only to classify the cases where it is certain in its answer. To do this, we utilize the binary classification model we created in 5.3.

> **Goal 6:** See how the accuracy changes when using difference confidence requirements during the classification phase, and see how many images the networks skips ("unsure, go see a doctor for further testing") when doing so.

When using the softmax activation function on the output prediction, the value gets modified into the range of 0-1, where high prediction values get close to 1 (it is certain), and low values get close to 0 (it is uncertain)[9]. After the activation function, a check was made to see if the output was above or below the current threshold. If it was below the threshold, the image is skipped because the network was not certain enough for our liking, and the user gets the message: "unsure, go see a doctor for further testing." If the classification confidence is above the current threshold, the classification is done the same way as in 5.3.

A range of confidence thresholds was tested, which can be seen in this list: [0.0, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.92, 0.94, 0.95, 0.96, 0.97, 0.98, 0.99, 0.995, 0.999, 0.9999, 0.99999, 0.999999, 1]. After running the test, a chart was made where the number of images predicted and the accuracy was charted. The image can be seen in Figure 5.4

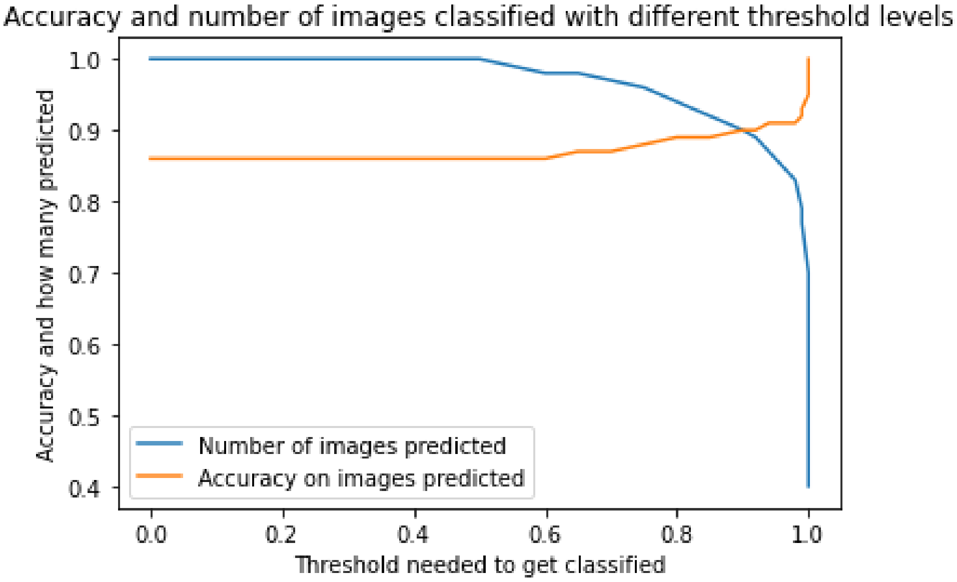


Figure 5.4: Accuracy and number of images classified with different threshold levels

A zoomed-in image of the chart can be seen on the next page in Figure 5.5, where the focus is on the higher threshold levels. A table of all the information can be seen in Table 5.13.

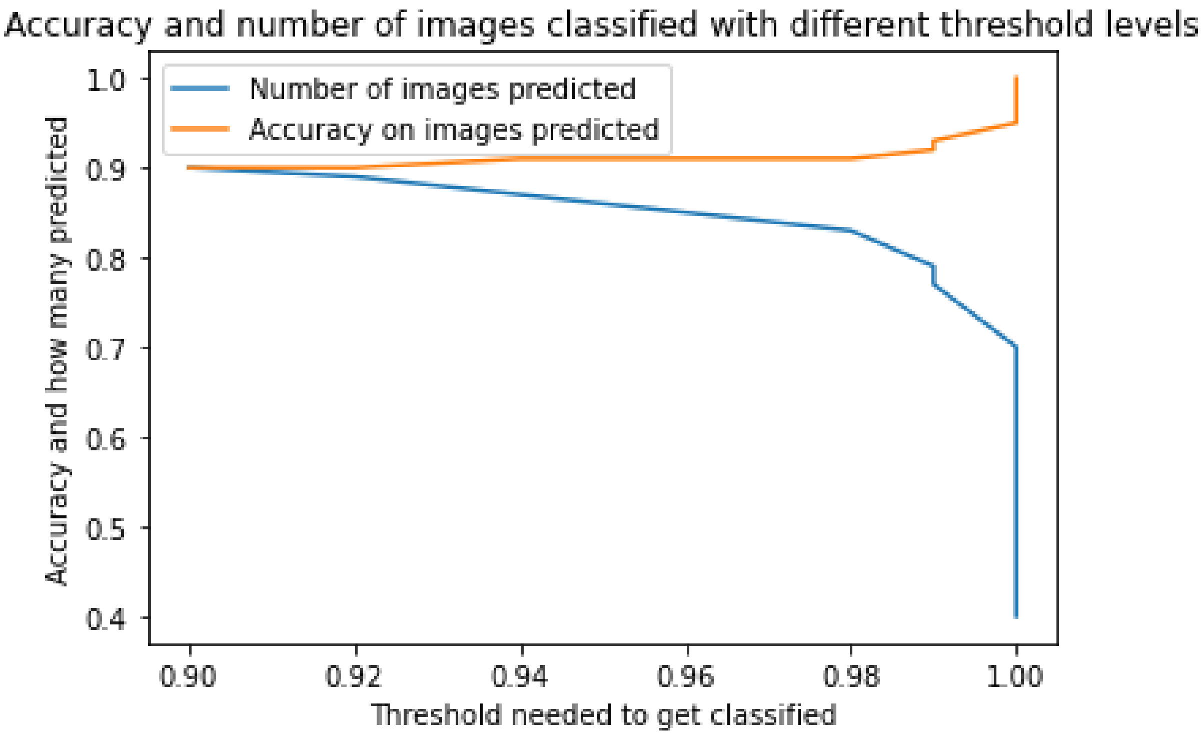


Figure 5.5: A zoomed in image with the higher classifications

Threshold	Images predicted	Total Accuracy	Accuracy benign	Accuracy dangerous
0.0	100.00% - 1001/1001	85.81%	92.1% - 735 / 798	61.1% - 124 / 203
0.5	100.00% - 1001/1001	85.81%	92.1% - 735 / 798	61.1% - 124 / 203
0.7	96.90% - 970/1001	87.11%	93.5% - 729 / 780	61.1% - 116 / 190
0.8	94.01% - 941/1001	88.63%	94.5% - 723 / 765	63.1% - 111 / 176
0.85	92.01% - 921/1001	88.71%	94.6% - 713 / 754	62.3% - 104 / 167
0.9	89.71% - 898/1001	90.09%	95.7% - 707 / 739	64.2% - 102 / 159
0.94	87.01% - 871/1001	90.70%	95.7% - 695 / 726	65.5% - 95 / 145
0.97	84.32% - 844/1001	91.11%	96.0% - 679 / 707	65.7% - 90 / 137
0.99	79.12% - 792/1001	92.30%	97.2% - 655 / 674	64.4% - 76 / 118
0.995	76.72% - 768/1001	93.10%	97.3% - 646 / 664	66.3% - 69 / 104
0.999	70.43% - 705/1001	94.89%	98.3% - 619 / 630	66.7% - 50 / 75
0.9999	61.64% - 617/1001	97.57%	99.1% - 569 / 574	76.7% - 33 / 43
0.99999	53.55% - 536/1001	98.13%	99.4% - 509 / 512	70.8% - 17 / 24
0.999999	47.75% - 478/1001	98.95%	99.6% - 463 / 465	76.9% - 10 / 13
1.0	40.36% - 404/1001	99.50%	100.0% - 400 / 400	50.0% - 2 / 4

Table 5.13: Different thresholds raw data table

Chapter 6

Discussions

This chapter will be about discussion, where I talk about the results achieved in Chapter 5 and explain what the different results show.

- Section 6.1 will discuss the results achived on the 20 valid and 20 test split
- Section 6.2 will discuss the results achived on the 80/10/10 split
- Section 6.3 will discuss the results achived when using multiple models to vote
- Section 6.4 will discuss the results achived when splitting the dataset into dangerous and benign
- Section 6.5 will discuss the results achived when utilizing different confidence thresholds

6.1 20 valid 20 test images discussion

With this split, we can understand how the different models perform without doing any work to combat the class imbalances. As expected, the accuracy for the majority classes, such as NV, performs very well in all of the different tests. In contrast, the minority classes such as DF perform very poorly in all of the different models. DF, which is the smallest class, even hit 0% accuracy in one of the models.

The total accuracy for the different runs can be seen in the table below:

Model	**Total accuracy**
VGG16 no image augmentation	49.2%
ResNet34 no image augmentation	57.14%
ResNet34 with image augmentation	64.29%
EfficientNet_B4 no image augmentation	55.71%
EfficientNet_B4 with image augmentation	52.86%

Table 6.1: Accuracies on the 20 valid 20 test images

These results are not very good, so we move on to the next steps to combat class imbalances.

6.2 80/10/10 split discussion

The following models were trained with the more common way of splitting datasets, where the classes keep their proportions in the validation and test set. This split can get a better idea of how the model will perform on totally new images, as real-life testing will come in with the same class frequency as what the dataset is made up of. This split also utilizes 1000 images as the test images, making it a lot more accurate compared to the 140 images used in the previous split. 140 images is a pretty low amount, so the accuracy given is either a bit lower or a bit higher than it would be given a bigger sample size to test on due to variance.

Splitting the dataset like this achieved a much better accuracy than the previous split, with the highest accuracy hitting 83.02% accuracy after utilizing oversampling to reduce the class imbalances. This is much better than the last split, where the best performing model hit 64.29%. Some more tests were performed using heavier image augmentation than shown here. However, the results were worse as heavy image augmentation removed critical characteristics from the different classes, such as modifying color.

Balancing dataset with oversampling

To increase the accuracy of the minority classes, I decided to use oversampling to even out the training images. Doing so increased the number of training images from 8000 to 37000 due to how extreme the class imbalances are in this dataset. Some of the smallest classes like DF and VASC had to be duplicated many times to become as big as NV. This caused the training to take more than 85 hours, but the accuracy increased up to 83.02% on the test set.

After oversampling, the model trained on a balanced dataset achieved 3% higher total accuracy than the model without a balanced dataset, but the difference in the smaller classes is much more significant. For example, BCC increased by more than 5%, and MEL increased by almost 13%, which is a huge improvement. The results of the different models can be seen in Table 6.2, and the comparison between imbalanced dataset and oversampled dataset can be seen in Table 6.3.

Model	Total accuracy
ResNet34 with image augmentation	80.62%
EfficientNet_B4 with image augmentation	78.82%
ResNet34 with image augmentation - balanced with oversampling	83.02%

Table 6.2: Accuracies on the 80/10/10 split

Class	unbalanced acc	oversampling acc
akiec	43.8%	43.8%
bcc	64.2%	71.7%
bkl	69.0%	73.0%
df	40.0%	40.0%
mel	42.4%	55.1%
nv	93.5%	93.6%
vasc	50.0%	50.0%
total acc	80.62%	83.02%

Table 6.3: Comparison between imbalanced and oversampled dataset

6.3 Using multiple models that votes with their prediction discussion

Due to different networks learning different types of features and patterns during training, with some networks being better than others at different classes, I wanted to see what would happen if multiple different models paired up to vote on which class each image belonged to. Doing so could remove some of the specific biases each network learns, and if multiple different networks vote on the same class, the confidence that it is the correct answer should increase.

Two methods were tested to achieve the voting. The first method was having each model make a prediction independently which was used to vote, and the plurality vote was chosen as the prediction. The second method was by adding all of the outputs from each model together before making the prediction based on the new output tensor to keep the confidence levels during the voting.

The first method achieved an accuracy of 83.84%, and the second method achieved an accuracy of 84.62%. Both of these results were better than what the models were able to achieve independently, which was 83.02% accuracy, and shows that working together increases the accuracy even if the different models are not as good as the best performing model. The second method, which kept the confidence levels of each network when making the predictions, performed better than just having the models vote on their top prediction.

6.4 Binary classification between dangerous and benign discussion

Due to the possibility of using the models as a binary classification of whether a patient needs to go see a doctor or if they do not need to worry about their lesion, a different model was created to do just that. The different diseases were split into two different classes based on the severity of the disease. BKL, DF, NV, and VASC were grouped up into a benign class, and AKIEC, BCC, and MEL were grouped into a dangerous class.

Grouping the classes together made the class imbalances lower than previously, with the dangerous class containing 2000 images and the benign class containing 8000 images. After oversampling to even it out, the network ended up with 12000 images to train on, and the chosen CNN architecture was ResNet34 with small image augmentation such as flips and rotation. The network managed to achieve 85.81% accuracy after training for 200 epochs.

6.5 Using different thresholds when doing the classification discussion

As a follow-up to Section 6.4, I tested a method where the network only classifies the images where it was certain and said "unsure, go see a doctor for further testing" in the cases where it was uncertain. This could be utilized for a hospital where they want to reduce the workload by making the model perform all of the "easy" classifications instead of having the model take over all of the cases. By doing it like this, the hospital can perform extensive testing on the difficult cases where the diagnosis is not as trivial. The results from the different

threshold levels can be seen in 5.5, where I utilized many different threshold levels to classify the images. The different thresholds can depend on how certain you want the network to be before the results are good enough and a doctor visit is not needed. The results were very good, and a table with the most promising thresholds can be seen below:

Threshold	**Classification accuracy**	**Number of patients diagnosed**
0.0	85.81%	100.00% - 1001/1001
0.9	90.09%	89.71% - 898/1001
0.999	94.89%	70.43% - 705/1001
0.9999	97.57%	61.64% - 617/1001
1.0	99.50%	40.36% - 404/1001

Table 6.4: Different threshold levels depending on accuracy requirement

With no threshold, the results are exactly the same as in 6.4 as the network will diagnose every person as usual. As the threshold level increases, the classification accuracy increases while the number of diagnoses goes down. This is because the network skips all of the hard cases and sends them to the hospital for a thorough check while it still gives a diagnosis in the easy cases. Based on accuracy requirements, if a 90% accuracy is required, it will reduce the workload of the hospital by 90%. If a 95% accuracy is required, it will reduce the workload of the hospital by 70%. In cases where the network is not allowed to make any mistakes, the model managed 99.5% accuracy by still diagnosing 40% of the patients!

Chapter 7

Conclusions

7.1 Hypoteses and goals

Hypothesis 1: *The network will be able to achieve higher accuracy than human experts with more than ten years of experience (62.6%)*

> The work done in this thesis shows that multiple models can achieve higher accuracy than what human experts can achieve. Human experts range from 59 to 63% accuracy, while the models in this thesis achieved as high as 84.62% accuracy when classifying between the seven different skin diseases and 85.81% accuracy when classifying between benign/dangerous.

Hypothesis 2: *Multiple neural network models working together by voting will perform better than the best network working alone*

> As shown in Section 5.4, multiple models were successfully combined into working together, and they were able to achieve higher accuracy than the models were able to achieve by themselves. When pairing four networks together (83.02%, 78.92%, 80.12%, 80.22%), the total accuracy went up to 84.62% accuracy. This is an increase of over 1% above the best performing singular model, which proves that the models learn different things to help each other.

Goal 1: *Explore different state-of-the-art convolutional neural network architectures to figure out what the main differences are*

> State-of-the-art convolutional neural network architectures were outlined in Section 3.1. Here I briefly discussed the history of CNNs, and how the state-of-the-art models have evolved during the past decade. Most CNN architectures are benchmarked on the ImageNet dataset, and in the following chapters, I went into detail about how three different state-of-the-art architectures functions: VGG16, ResNet34, and EfficientNet.

Goal 2: *Find out what techniques are predominantly used to combat class imbalances in datasets*

> Class imbalances in datasets are one of the main challenges people face when working with real-world data. There are multiple ways to combat these imbalances: adjusting the dataset through undersampling or oversampling, adjusting the algorithms by modifying the loss function and using different weights, and using image augmentation to increase the number of images the network has to train on. The different methods were further discussed in Section 4.3

Goal 3: *Test the differences when training a neural network to be used as an abnormality detector (cancer/non-cancer) vs training a neural network to classify between seven different skin diseases*

> This thesis's main focus was on classifying all seven types of skin diseases. However, some experiments were also conducted where the different diseases were grouped together into dangerous/benign based on their severity. Classifying between more classes is harder for the models because there are more choices to choose from, and the results also show this. The best models achieved 84.62% accuracy when classifying between the seven different skin diseases, while the accuracy increased to 85.81% when classifying between benign/dangerous. These models can have different use cases. While doctors might prefer knowing exactly which disease they are dealing with, patients may wish to focus on knowing if they have to seek help or not.

Goal 4: *Achieve a higher classification accuracy than human experts who are working in the field of dermatology*

> How human experts manage to perform when dealing with this dataset is shown in Section 2.4.3, and they manage to achieve around 62% accuracy. The models shown in Section 5.2 perform much better than the human experts, achieving accuracies in the mid 80s% range.

Goal 5: *Train multiple models which can be used together when classifying images by using a voting-based system*

> All of the models that were trained in Section 6.2 were combined to work together in Section 6.3. The results when working together outperformed all of the models working independently, which means that the method achieved great results.

Goal 6: *See how the accuracy changes when using difference confidence requirements during the classification phase, and see how many images the networks skips ("unsure, go see a doctor for further testing") when doing so.*

> As the results in Section 5.5 and discussion in Section 6.5 show, using confidence thresholds managed to improve the accuracy drastically at the cost of only skipping a low amount of images. if a 90% accuracy is required, it will reduce the workload of the hospital by 90%. If a 95% accuracy is required, it will reduce the workload of the hospital by 70%. In cases where the network is not allowed to make any mistakes, the model managed 99.5% accuracy by still diagnosing 40% of the patients. These results are very promising, and it shows that there is only a small amount of highly uncertain images that lowers the total accuracy by a huge margin.

7.2 Conclusion summary

All of the goals that were chosen to be in the scope of this thesis were accomplished, and the different models were able to classify the seven different skin diseases to a satisfying degree. Different methods to combat the class imbalances have been tested, and oversampling together with image augmentation and using multiple models to vote performed the best. The best model to classify all the diseases achieved 84.62% accuracy through the usage of voting, while the best model to classify dangerous/benign achieved 85.81% accuracy. In cases where confidence thresholding was used, a 90% accuracy was achieved by classifying 90% of the images, 95% accuracy achieved by classifying 70% of the images, and 99.5% accuracy achieved by classifying 40% of the images.

7.3 Future Work

As for the future work, multiple new things would be possible to try out given more time. This section explains some of the areas I would suggest for further research.

7.3.1 Using bigger and more advanced CNN architectures

The CNN architectures used in this thesis are all state-of-the-art architectures, but the models were limited by the computing power and GPU memory of my computer, which is listed below in Table 7.1. Given more computing power, I would have done further testing using bigger and better architectures such as EfficientNet_B7 instead of EfficientNet_B4, and also some of the highest performing networks on ImageNet such as CoCa. These networks would most likely improve the performance, at the cost of needing a better computer to train on. This could be done using cloud computing such as Google Cloud. I started setting up and did a few small tests with google cloud, but I did not have the time to complete them. As an example, the differences between smaller and bigger networks in EfficientNet can be seen in Figure 7.2, and the biggest network my computer was able to run was EfficientNet_b4.

Type	Model
Processor	AMD Ryzen 5 3600X
GPU	Nvidia GTX 1060-3GB
Memory	16GB DDR4 3200MHz
Operating System	Windows 10

Table 7.1: Computer specs

Name	Parameters	Top-1 accuracy
efficientnet-b0	5.3m	76.3
efficientnet-b1	7.8m	78.8
efficientnet-b2	9.2m	79.8
efficientnet-b3	12m	81.1
efficientnet-b4	19m	82.6
efficientnet-b5	30m	83.3
efficientnet-b6	43m	84.0
efficientnet-b7	66m	84.4

Table 7.2: Size and accuracy of the different EfficientNets. The bigger architectures require more parameters and ram, but they aquire higher accuracy as a result[3].

7.3.2 Utilizing more datasets

As mentioned in Chapter 3.2, many of the current state-of-the-art solutions utilize multiple datasets combined to achieve more images to train on. The most common datasets to group together are the ISIC dataset and HAM10000. In this thesis, I wanted to focus on just one dataset and try different methods to combat the class imbalances and improve the performance. However, if the goal were to achieve the highest accuracy possible, combining the datasets to achieve more training images would help. The training times would take longer, but I would do this to achieve a better-performing model.

7.3.3 Utilizing image segmentation

One interesting method for classifying images is to use image segmentation in combination with standard classification. The image segmentation can be combined with the training images to show the network where it needs to focus, so it can learn all of the skin lesion features while completely ignoring the background around the lesions. There have been some experiments on combining these, such as in [41], but more experimentation would be preferred. I briefly tested image segmentation using DDANet [67], but I did not have enough time to achieve results worth putting in the paper.

Bibliography

[1] Grant Sanderson 3Blue1Brown. *But what is a neural network? | Chapter 1, Deep learning.* URL: https://www.youtube.com/watch?v=aircAruvnKk. (accessed: 18.04.2022).

[2] Google AI - Hongkun Yu et al. *EfficientNet: Improving Accuracy and Efficiency through AutoML and Model Scaling.* URL: https://ai.googleblog.com/2019/05/efficientnet-improving-accuracy-and.html. (accessed: 26.05.2022).

[3] Github - Luke Melas-Kyriazi Et al. *EfficientNet PyTorch.* URL: https://github.com/lukemelas/EfficientNet-PyTorch. (accessed: 30.05.2022).

[4] Yann LeCun et al. *THE MNIST DATABASE of handwritten digits.* URL: http://yann.lecun.com/exdb/mnist/index.html. (accessed: 14.04.2022).

[5] Pragati Baheti. *12 Types of Neural Network Activation Functions: How to Choose?* URL: https://www.v7labs.com/blog/neural-networks-activation-functions. (accessed: 07.05.2022).

[6] Gaudenz Boesch. *A Complete Guide to Image Classification in 2022.* URL: https://viso.ai/computer-vision/image-classification/. (accessed: 09.05.2022).

[7] Jason Brownlee. *A Gentle Introduction to the ImageNet Challenge (ILSVRC).* URL: https://machinelearningmastery.com/introduction-to-the-imagenet-large-scale-visual-recognition-challenge-ilsvrc/. (accessed: 14.05.2022).

[8] Jason Brownlee. *Loss and Loss Functions for Training Deep Learning Neural Networks.* URL: https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/. (accessed: 19.04.2022).

[9] Jason Brownlee. *Softmax Activation Function with Python.* URL: https://machinelearningmastery.com/softmax-activation-function-with-python/. (accessed: 29.05.2022).

[10] A W Kopf Et al. C M Grin 1. "Accuracy in the Clinical Diagnosis of Malignant Melanoma." In: *Archives of Dermatology* 126.6 (June 1990), pp. 763–766. ISSN: 0003-987X. DOI: 10.1001/archderm.1990.01670300063008. eprint: https://jamanetwork.com/journals/jamadermatology/articlepdf/551839/archderm_126_6_008.pdf. URL: https://doi.org/10.1001/archderm.1990.01670300063008.

[11] Kendra Cherry. *How Many Neurons Are in the Brain?* URL: https://www.verywellmind.com/how-many-neurons-are-in-the-brain-2794889. (accessed: 11.04.2022).

[12] Papers With Code. *Image Classification on ImageNet.* URL: https://paperswithcode.com/sota/image-classification-on-imagenet. (accessed: 18.05.2022).

[13] Noel Codella et al. *Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC).* 2019. arXiv: 1902.03368 [cs.CV].

[14] CompaniesMarketCap. *Largest telehealth companies by market cap.* URL: https://companiesmarketcap.com/telehealth/largest-companies-by-market-cap/. (accessed: 09.02.2022).

[15] TechTarget Contributor. *garbage in, garbage out (GIGO).* URL: https://www.techtarget.com/searchsoftwarequality/definition/garbage-in-garbage-out. (accessed: 16.05.2022).

[16] B.J. Copeland. *artificial intelligence.* URL: https://www.britannica.com/technology/artificial-intelligence. (accessed: 30.01.2022).

[17] dermoscopedia. *Actinic keratosis / Bowen's disease / keratoacanthoma / squamous cell carcinoma*. URL: https://dermoscopedia.org/Actinic_keratosis_/_Bowen%5C%27s_disease_/_keratoacanthoma_/_squamous_cell_carcinoma. (accessed: 9.04.2022).

[18] Rachel Draelos. *The History of Convolutional Neural Networks*. URL: https://glassboxmedicine.com/2019/04/13/a-short-history-of-convolutional-neural-networks/. (accessed: 14.05.2022).

[19] IBM Cloud Education. *Convolutional Neural Networks*. URL: https://www.ibm.com/cloud/learn/convolutional-neural-networks. (accessed: 09.05.2022).

[20] IBM Cloud Education. *Gradient Descent*. URL: https://www.ibm.com/cloud/learn/gradient-descent. (accessed: 10.04.2022).

[21] MARK H. LOWITT FERN A. WIRTH. *Diagnosis and Treatment of Cutaneous Vascular Lesions*. URL: https://www.aafp.org/afp/1998/0215/p765.html. (accessed: 10.04.2022).

[22] Fei Gao et al. "SD-CNN: A shallow-deep CNN for improved breast cancer diagnosis." In: *Computerized Medical Imaging and Graphics* 70 (2018), pp. 53–62. ISSN: 0895-6111. DOI: https://doi.org/10.1016/j.compmedimag.2018.09.004. URL: https://www.sciencedirect.com/science/article/pii/S0895611118302349.

[23] ml-glossary github. *Gradient Descent*. URL: https://ml-cheatsheet.readthedocs.io/en/latest/gradient_descent.html. (accessed: 08.05.2022).

[24] Alexander Jung (aleju on GitHub). *imgaug*. URL: https://github.com/aleju/imgaug. (accessed: 24.05.2022).

[25] Joanna Goodrich. *How IBM's Deep Blue Beat World Champion Chess Player Garry Kasparov*. URL: https://spectrum.ieee.org/how-ibms-deep-blue-beat-world-champion-chess-player-garry-kasparov. (accessed: 30.01.2022).

[26] Alla Eddine Guissous. *Max-pooling and average-pooling example image*. URL: https://www.researchgate.net/figure/Example-for-the-max-pooling-and-the-average-pooling-with-a-filter-size-of-22-and-a_fig15_337336341. (accessed: 10.05.2022).

[27] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. DOI: 10.48550/ARXIV.1512.03385. URL: https://arxiv.org/abs/1512.03385.

[28] NHS health. *Actinic keratoses (solar keratoses)*. URL: https://www.nhs.uk/conditions/actinic-keratoses/. (accessed: 9.04.2022).

[29] ImageNet. *About ImageNet*. URL: https://image-net.org/about.php. (accessed: 14.05.2022).

[30] ImageNet. *ImageNet Large Scale Visual Recognition Challenge (ILSVRC)*. URL: https://image-net.org/challenges/LSVRC/index.php. (accessed: 14.05.2022).

[31] Queensland Brain Institute. *Action potentials and synapses*. URL: https://qbi.uq.edu.au/brain-basics/brain/brain-physiology/action-potentials-and-synapses. (accessed: 11.04.2022).

[32] Queensland Brain Institute. *What is a neuron?* URL: https://qbi.uq.edu.au/brain/brain-anatomy/what-neuron. (accessed: 11.04.2022).

[33] ISIC. *ISIC 2019 Leaderboards*. URL: https://challenge.isic-archive.com/leaderboards/2019/. (accessed: 17.03.2022).

[34] Ronald L. Moy Julie K. Karen. *Basal Cell Carcinoma Overview*. URL: https://www.skincancer.org/skin-cancer-information/basal-cell-carcinoma/. (accessed: 9.04.2022).

[35] Bharath K. *Understanding ReLU: The Most Popular Activation Function in 5 Minutes!* URL: https://towardsdatascience.com/understanding-relu-the-most-popular-activation-function-in-5-minutes-459e3a2124f. (accessed: 08.05.2022).

[36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks.” In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., 2012, pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[37] abhishek kushwaha. *Solving Class Imbalance problem in CNN*. URL: https://medium.com/x8-the-ai-community/solving-class-imbalance-problem-in-cnn-9c7a5231c478. (accessed: 16.05.2022).

[38] Y. Lecun et al. “Gradient-based learning applied to document recognition.” In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[39] The American Cancer Society medical and editorial content team. *What Is Melanoma Skin Cancer?* URL: https://www.cancer.org/cancer/melanoma-skin-cancer/about/what-is-melanoma.html. (accessed: 10.04.2022).

[40] James Muir. *Re.: Accuracy in skin cancer diagnosis: a retrospective study of an Australian public hospital dermatology department*. URL: https://pubmed.ncbi.nlm.nih.gov/22881468/. (accessed: 29.05.2022).

[41] Hemanth Nadipineni. “Method to Classify Skin Lesions using Dermoscopic images.” In: *CoRR* abs/2008.09418 (2020). arXiv: 2008.09418. URL: https://arxiv.org/abs/2008.09418.

[42] Dr Amanda Oakley. *Dermatofibroma*. URL: https://dermnetnz.org/topics/dermatofibroma. (accessed: 10.04.2022).

[43] Dr Amanda Oakley. *Dermoscopy*. URL: https://dermnetnz.org/topics/dermoscopy. (accessed: 31.01.2022).

[44] Dr Amanda Oakley. *Melanocytic naevus*. URL: https://dermnetnz.org/topics/melanocytic-naevus. (accessed: 10.04.2022).

[45] Dr Amanda Oakley. *Seborrhoeic keratosis*. URL: https://dermnetnz.org/topics/seborrhoeic-keratosis. (accessed: 9.04.2022).

[46] Philipp Tschandl, Noel Codella, et al. *Comparison of the accuracy of human readers versus machine-learning algorithms for pigmented skin lesion classification: an open, web-based, international, diagnostic study*. https://pubmed.ncbi.nlm.nih.gov/31201137/. 2019. (accessed: 14.04.2022).

[47] Lumen Learning - Boundless Psychology. *Neurons*. URL: https://courses.lumenlearning.com/boundless-psychology/chapter/neurons/. (accessed: 11.04.2022).

[48] PyTorch. *TRANSFORMING AND AUGMENTING IMAGES*. URL: https://pytorch.org/vision/stable/transforms.html. (accessed: 26.05.2022).

[49] Sebastian Raschka. *L14.3.1.1 VGG16 Overview*. URL: https://www.youtube.com/watch?v=YcmNIOyfdZQ. (accessed: 19.05.2022).

[50] Richard Kinh Gian Do Kaori Togashi Rikiya Yamashita Mizuho Nishio. *Convolutional neural networks: an overview and application in radiology*. URL: https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9. (accessed: 11.05.2022).

[51] Brandon Rohrer. *How Convolutional Neural Networks work*. URL: https://www.youtube.com/watch?v=FmpDIaiMIeA. (accessed: 10.05.2022).

[52] Olga Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge.” In: (2014). cite arxiv:1409.0575Comment: 43 pages, 16 figures. v3 includes additional comparisons with PASCAL VOC (per-category comparisons in Table 3, distribution of localization difficulty in Fig 16), a list of queries used for obtaining object detection images (Appendix C), and some additional references. URL: http://arxiv.org/abs/1409.0575.

[53] Sumit Saha. *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way*. URL: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. (accessed: 09.05.2022).

[54] Dave Sebastian Sebastian Herrera and Sarah Krouse. *Amazon to Offer Telehealth Service to Other U.S. Firms This Summer.* URL: https://www.wsj.com/articles/amazon-to-offer-telehealth-service-to-other-companies-11615997996. (accessed: 08.02.2022).

[55] Connor Shorten. *EfficientNet Explained!* URL: https://www.youtube.com/watch?v=3svIm5UC94I. (accessed: 26.05.2022).

[56] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." In: *arXiv preprint arXiv:1409.1556* (2014).

[57] C J Prabhakar Sneha Kugunavar. *Convolutional neural networks for the diagnosis and prognosis of the coronavirus disease pandemic.* URL: https://pubmed.ncbi.nlm.nih.gov/33950399/. (accessed: 24.05.2022).

[58] Mayo Clinic Staff. *Melanoma.* URL: https://www.mayoclinic.org/diseases-conditions/melanoma/symptoms-causes/syc-20374884. (accessed: 10.04.2022).

[59] Mayo Clinic Staff. *Seborrheic keratosis.* URL: https://www.mayoclinic.org/diseases-conditions/seborrheic-keratosis/symptoms-causes/syc-20353878. (accessed: 9.04.2022).

[60] Mayo Clinic Staff. *Telehealth: Technology meets health care.* URL: https://www.mayoclinic.org/healthy-lifestyle/consumer-health/in-depth/telehealth/art-20044878. (accessed: 08.02.2022).

[61] Josef Steppan. *$MNIST_{e}xamples_{i}mage$.* URL: https://en.wikipedia.org/wiki/MNIST_database#/media/File:MnistExamples.png. (accessed: 14.04.2022).

[62] Marzuraikah Mohd Stofa, Mohd Asyraf Zulkifley, and Muhammad Ammirrul Atiqi Mohd Zainuri. "Skin Lesions Classification and Segmentation: A Review." In: *International Journal of Advanced Computer Science and Applications* 12.10 (2021). DOI: 10.14569/IJACSA.2021.0121060. URL: http://dx.doi.org/10.14569/IJACSA.2021.0121060.

[63] Mingxing Tan and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks." In: (2019). DOI: 10.48550/ARXIV.1905.11946. URL: https://arxiv.org/abs/1905.11946.

[64] Great Learning Team. *Introduction to VGG16 | What is VGG16?* URL: https://www.mygreatlearning.com/blog/introduction-to-vgg16/. (accessed: 19.05.2022).

[65] Tesla. *Autopilot.* URL: https://www.tesla.com/autopilot. (accessed: 30.01.2022).

[66] Rohit Thakur. *Step by step VGG16 implementation in Keras for beginners.* URL: https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c. (accessed: 19.05.2022).

[67] Nikhil Kumar Tomar et al. *DDANet: Dual Decoder Attention Network for Automatic Polyp Segmentation.* 2020. DOI: 10.48550/ARXIV.2012.15245. URL: https://arxiv.org/abs/2012.15245.

[68] Philipp Tschandl. *The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions.* URL: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/DBW86T. (accessed: 31.01.2022).

[69] Wikipedia. *Artificial neural network.* URL: https://no.wikipedia.org/wiki/Fil:Artificial_neural_network.svg. (accessed: 11.04.2022).

[70] Wikipedia. *Basal-cell carcinoma.* URL: https://en.wikipedia.org/wiki/Basal-cell_carcinoma. (accessed: 9.04.2022).

[71] Wikipedia. *Melanoma.* URL: https://en.wikipedia.org/wiki/Melanoma. (accessed: 10.04.2022).

[72] Wikipedia. *Vanishing gradient problem.* URL: https://en.wikipedia.org/wiki/Vanishing_gradient_problem. (accessed: 4.03.2022).

[73] Samir S. Yadav and Shivajirao Manikrao Jadhav. "Deep convolutional neural network based medical image classification for disease diagnosis." In: *Journal of Big Data* 6 (2019), pp. 1–18.