

# Using Neural Networks to Achieve Optical Character Recognition

*Daniel Lages*

*15006972*

*MComp Computer Science*

*2018*

## Table of Contents

List of Figures .....	3
List of Tables .....	3
Statement of Originality .....	4
Abstract .....	5
Introduction .....	6
Project Statement .....	7
Objectives .....	7
Research .....	8
Natural Neural Networks .....	8
The Perceptron Model.....	8
Activation Functions.....	11
Forward Propagation .....	12
Back-Propagation .....	12
Optical Character Recognition .....	13
Competing Products .....	14
Methodology.....	15
Ethical Challenges .....	17
Implementation .....	17
Implementing the MNIST Dataset .....	17
Layers.....	18
Activation functions .....	19
Training with MNIST Data .....	19
Pre-Processing.....	20
Dataset Creation and Training .....	22
Classification .....	24
Results .....	24
Test Plan .....	24
Training.....	24
Classification .....	27
Conclusion.....	30
Critical Evaluation .....	31
Future Work.....	31
Bibliography .....	32
Appendix.....	34
Project Meeting Records .....	34
Gantt Chart .....	43
Ethics Checklist.....	44

## List of Figures

Figure 1- Representing the computation performed by the McCulloch – Pitts neuron	8
Figure 2: Demonstrating the computation of a perceptron processing unit	9
Figure 3: Representing the training process with two classes of data (Dunne, 2007)	10
Figure 4: The sigmoid activation function - (Stokastik, 2018)	11
Figure 5: The effect of overtraining on the decision boundary (Left) in comparison with the decision boundary generate as a result of optimal training (Right) - (Gurney, 1999)	13
Figure 6: Representing the training process to be used by the neural network model	16
Figure 7: The classification process of a character within a trained network	16
Figure 8: Representing the use of a 3D Array to determine the features of a character ( <a href="https://www.tensorflow.org/">https://www.tensorflow.org/</a> , no date)	18
Figure 9: The implementation of the hidden second layer, using the output of the first layer as input	18
Figure 10: Representing the use of the sigmoid function within the neural network	19
Figure 11: The use of the Tensorflow function in order to optimise the network based upon the margin of the error using Stochastic Gradient Decent	19
Figure 12: Representing the output of the training model	20
Figure 13: Representing the improvement of image clarity through the use of the median filter.	20
Figure 14:: Representing the effects of differing contrast levels in comparison with the original image.	21
Figure 15:: Representing the effects of thresholding on the sharpness and clarity of the image	22
Figure 16: Representing an example of the 'a' and 'b' character data points used for training	23
Figure 17: Demonstrating the function created in order to create and group the data and labels.	23
Figure 18: Representing the effects of a different number of nodes upon training accuracy.	25
Figure 19: Performance with a varying number of hidden layer nodes, a dataset of 106 data points.	25
Figure 20: Representing the effects of a different number of nodes upon training accuracy	26
Figure 21: Performance when trained with a different number of epochs over 5 tests	26
Figure 22: The effect of the ReLU and Sigmoidal units on the accuracy of the neural network model	27
Figure 23: Representing the un-successful classification of the digit '9'	28
Figure 24: Representing the successful classification of the digits '4' and '1'	28
Figure 25: Representing the successful classification of the characters 'a' and 'b'.	29

## List of Tables

Table 1: Representing the parameters used when trained with the MNIST dataset	28
Table 2: Representing the parameters used when trained with the newly created dataset	29

# Statement of Originality

## Appendix I

### **STATEMENT OF ORIGINALITY**

#### **CS3D660 Individual Project**

This is to certify that, except where specific reference is made, the work described within this project is the result of the investigation carried out by myself, and that neither this project, nor any part of it, has been submitted in candidature for any other award other than this being presently studied.

Any material taken from published texts or computerized sources have been fully referenced, and I fully realize the consequences of plagiarizing any of these sources.

Student Name (Printed) DANIEL LAGES

Student Signature 

Registered Course of Study Computer Science

Date of Signing 19/11/2017

## Abstract

Neural networks are a prominent computational concept. Used to complete a range of tasks in areas such as classification and prediction, neural networks overcome many of the challenges present in traditional computing paradigms. Components such as processing units and weighted connections enable the use of the back-propagation learning algorithm. Through the use of such components, a network model is able to train in accordance with a provided dataset, tuning weighted connections in order to obtain the desired result for each data point, with a high level of accuracy. This computational process can be applied to a range of tasks such as the classification of handwritten characters.

This project investigates the use of neural networks in order to achieve optical character recognition. A comprehensive examination of neural networks is carried out, with the aim of implementing a new model capable of classifying images of handwritten characters. Training is also explored in detail, in order to facilitate an existing dataset containing handwritten characters. This is followed by a comprehensive investigation into the creation of datasets. A range of pre-processing techniques will also be used to create a dataset, with each data point providing optimal conditions for training and classification. Finally, a range of tests are carried out, determining the effects of various training parameters on the learning and classification procedures.

At the conclusion of this project, it has been determined that the accuracy obtained by a trained model relies heavily on the optimal use of training parameters, such as the number of epochs used. The importance of optimising the neural network and the dataset used has been demonstrated, serving to reduce the risks of overtraining and misclassification. Through the use of ideal training parameters, optical character recognition can be achieved with a high degree of success.

# Introduction

Initially imagined in order to model the functionality and structure of the human brain, neural networks are now a fundamental computational paradigm. They provide a unique approach to computation that differs from the sequential processing methods implemented within traditional computers Skapura (1996). This differing approach allows neural networks to adapt to unique problems that, traditionally, cannot be solved by computers.

There are many types of neural networks, however, they share common features that serve to characterise the functionality that they employ (Dayhoff, 1996). These components include computational units, often referred to as nodes. Nodes serve as the main method of computation and can perform calculations on the input received. At a greater level of abstraction however, a node may consist of a complete neural network, increasing the level of complexity that can be computed. Connections, serve to regulate the data flow between nodes. Many neural networks are referred to as multi-layer networks, as they contain many instances of nodes and connections. These instances are known as layers. Each layer within a neural network handles a differing element of the computation problem at hand, often with a different level of abstraction from the previous layer (Fiesler, 1992).

A distinct difference between neural networks and traditional computing paradigms is the ability to learn and reactively adjust the required processes, in order to better match the expected result. The output of a neural network is based upon datasets that inform the network of the desired result corresponding to the current input. There are many methods that allow a network to better match the expected output, such as back-propagation. Back-propagation, is a learning algorithm used by many neural networks in order to calculate an error value and adjust the network accordingly. (Dayhoff, 1996).

The effectiveness of neural networks can be demonstrated by analysing the process taken by a conventional architecture to solve a computational problem. The traditional computing paradigms operate sequentially, with one process taking place after the next. This sequential method of processing limits the functionality of a computational device. For example, in order to complete a task such as optical character recognition, each letter must be processed sequentially. This behaviour defines key information, such as the edges and key details of the character that is being analysed. When each pixel has been processed, all of the gathered information is utilised in order to classify the letter (Skapura, 1996). However, neural networks distribute this image data across multiple nodes allowing for the parallel computation of each pixel.

Neural networks, can be used to solve a range of computational challenges. In fact, they can be used to solve any algorithm as a result of the described functionality that they utilise. They adhere to the divide and conquer approach, as complex problems can be separated into different instances of computation and then employed to achieve the desired outcome (Skapura, 1996).

Image recognition, and in this case handwritten character recognition, is a computational problem that is often solved through the use of neural networks. This project will approach the understanding of neural networks with the aim of achieving Optical Character Recognition. Datasets will be employed in order to present a neural network with training input and the corresponding expected outputs. There are also many pre-processing elements that will be investigated.

The pre-processing of handwritten characters may consist of a range of steps. For example, it is important that the resulting program is able to distinguish the background from the character itself. In some cases, often based on the datasets and input used, segmentation may be required in order to

separate a character from any connecting data points (Chaudhary 2011). An image of a handwritten character may also contain a large amount of visual noise that reduces the clarity of the image (Phangtriastu, Harefa and Tanoto, 2017). As a result, it may be necessary to adopt filtering techniques to recognise and eliminate the noise surrounding the processed character.

This project will investigate neural networks and how they can be used to achieve optical character recognition. This will involve a comprehensive investigation into neural networks and the functionality that allows them to learn and achieve the desired outcome, through the use of datasets. An investigation into this subject will consist of exploring the importance of different training parameters and how they are adjusted to train the network. As the project develops, more neural networking principles will be analysed, such as backpropagation and the activation functions that can be employed.

## Project Statement

The aim of this project is to demonstrate an understanding of neural networks and how they can be used in order to achieve optical character recognition. This project will explore a range of areas regarding neural networks. A program will be created that illustrates, through the use of datasets, how a neural network can be trained to recognize handwritten characters.

## Objectives

- The project must demonstrate a knowledge of neural networks and how they operate
- The project must explain and demonstrate how neural networks can be trained to improve the ability to recognize handwritten characters. This will be achieved through the facilitation of existing datasets.
- The project must demonstrate the various OCR practices that need to be implemented in order to optimise character recognition, such as pre-processing.
- The resulting program must be able to classify individual handwritten characters through the use of a neural network.
- Different neural networks are created to compare the ability to achieve optical character recognition between them.
- A dataset is created and used to train the neural network, with the aim of achieving optical character recognition.

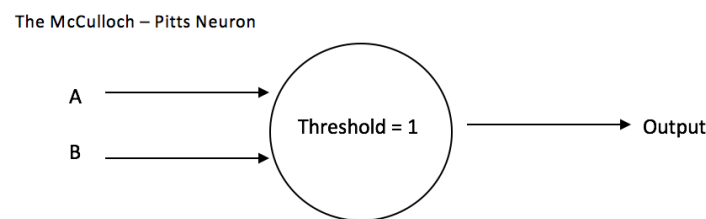
# Research

## Natural Neural Networks

Neural network architectures are modelled from the structure of the human nerve cell. Through considering the edifice of a natural neuron, one can gain an understanding of the functionality of artificial neural networks and their components (Dayhoff, 1996).

The human neuron is composed of a range of dendrite and axon fibres. The dendrite fibres receive the output provided by other neurons that are sent in the form of impulses. If generated, the impulse is sent down the axon fibres and to the dendrites of other neurons through a junction known as the synapse (Penrose, 1999). Based on the signals received by the target neuron, there will be an increased or decreased probability that the neuron's threshold will be reached and an impulse is sent down the axon to the next neuron.

In 1943 Warren McCulloch made perhaps the first attempt to envisage how a physical neuron operates within the brain (Anderson, 1995). He and Walter Pitts imagined a simple binary device that responds to it's the impulses received as inputs. In order to generate an output, the McCulloch – Pitts neuron analyses the sum of the inputs. If the sum exceeds the neuron's internal threshold value, it will fire.



*Figure 1: Representing the computation performed by the McCulloch – Pitts neuron*

Figure 1 represents the flow of data through the McCulloch – Pitts Neuron. Here the threshold is set to the value of 1. Creating the logic function 'OR'. This model for computation, although not a complete approximation to a physical neuron, was fundamental in the development of neural networks and the functionality that they provide (Anderson, 1995).

## The Perceptron Model

The McCulloch – Pitts neuron can be used to more clearly exemplify the functionality of a neuron. More importantly however, it establishes a basic model for the artificial neuron used in neural networks. The 1950's saw the development of an important computing model that utilised a similar functionality. According to Dayhoff (1996) the perceptron processing unit "provided an architecture that was eventually extended to the neural network learning systems of today". The perceptron model was imagined by Frank Rosenblatt who was fascinated by the cognitive learning processes of the brain. This architecture is important because of the real-world applications in which it has been implemented (Dayhoff, 1996)

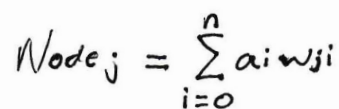
The perceptron algorithm takes a range of inputs and calculates the sum of the values in accordance with the threshold. The inputs also contain a weight value that is taken into account. The weights



assigned to the inputs are an important feature of this architecture, as it allows for the adjustment of the values received by the node. This is a powerful approach within the architecture of all neural networks. The weights of the input serve to affect the outcome of the connecting node and therefore, can serve as a template for reaching a desired target (Hu and Hwang, 2001).

The early perceptron architectures consisted of two layers. A layer within a neural network defines a set of nodes, with each layer performing a different process to solve the computational task at hand (Fiesler, 1992). For example, the input layer of a network consists of a range of nodes that receive input directly from the dataset. Conversely, the output layer contains a range of nodes that serve to generate an output value (Dayhoff, 1996). In many neural network architectures, there may be further layers that sit between the input and output layers. These nodes are used to perform calculations on the output of previous layers. This use of hidden layers allows the neural net to solve the computational problem with a greater level of abstraction (Dayhoff, 1996). A range of weighted connections serve to pass data between the layers, known as interlayer connections (Fiesler, 1992).

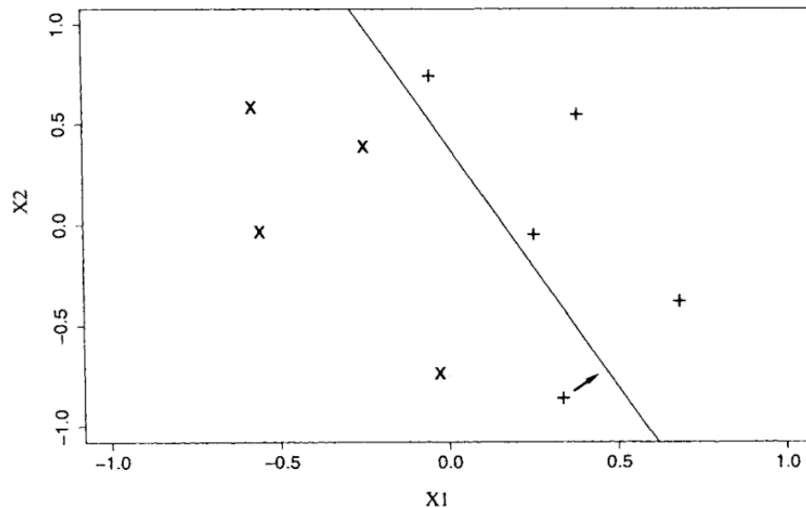
The perceptron architecture describes an input layer and an output layer, with the input layer serving only to pass the input to the following output layer. Rosenblatt envisaged the use of fixed units in order to transform the input received by the neural network. These fixed nodes can be described as bias units that act as fixed weights and corresponds to a subset of the raw input data. Dayhoff (1996) states that “the bias unit functions as a constant value”. After receiving the data from the input layer, the output layer then serves to calculate the sum of the inputs and the bias value. It is here that one can see the similarity with the McCulloch – Pitts neuron and indeed the natural neurons used by the human brain (Dunne, 2007).


$$Node_j = \sum_{i=0}^n a_i w_{ji}$$

*Figure 2: Demonstrating the computation of a perceptron processing unit*

The original perceptron paradigm contains just one layer of weights. After each computation, the perceptron architecture employs the use of a learning rule that reactively adjusts the weighted values in order to better match the desired output. This rule consists of an ‘IF’ statement that determines if the weighted sum of the inputs is greater than the node’s threshold value. Through the understanding of this rule, it can be seen why the perceptron architecture was a revolution within neural networks, as it allows for a machine to be trained in order to perform a certain task (Minsky and Papert, 1988).

The outcome of the node is therefore the result of the function performed. This function is often referred to as an activation function as it determines if the neuron fires. The training process is expressed in figure 3.



*Figure 3: Representing the training process with two classes of data (Dunne, 2007)*

The graph depicted in figure 3 details two classes of data, noted here as '+' and 'x'. For simplicity, and in accordance with the activation function denoted in figure 2, the '+' data serves to activate the threshold and the 'x' data does not. This is represented through where the data sits within the graph. The non-activating data sits below the threshold line (Dunne, 2007).

The threshold line that divides the two classes of data is often referred to as the decision boundary. In this instance, one input of class '+' sits below the decision boundary line and the network does not achieve the expected result. In order to resolve this, the network applies the learning algorithm and retroactively adjusts weights to better match the desired output. This is achieved iteratively. After multiple iterations and the tuning of weights, the network will achieve the expected result. This is detailed in figure 3 with the '+' that sits closest to the y axis. Despite this ability to learn, the simple perceptron paradigm is limited in its computational ability (Dayhoff, 1996).

Although fundamental to the development of modern neural networks, the perceptron paradigm did not provide the functionality required to complete the computational challenges solved by neural networks today. In 1969 Minsky and Papert demonstrated that the perceptron model was incapable of applying Rosenblatt's learning rule in order to overcome simple functions (Dunne, 2007). This is as a result of the number of layers used within the original perceptron model. The limited number of layers and interconnecting weights mean that the function is limited to performing 'AND' or 'OR' functions as a result of the binary output generated by each node.

In order to train a perceptron network, a data set is used. Once presented to the network, the weights are tuned. This process takes place many times in order to provide the program with the ability of determining the expected outcome based on the input (Hu and Hwang, 2001). Despite the evident increase in performance that the network gains through training, at a certain point, the increase in performance ceases. This halt in performance may be as a result of the successful or unsuccessful learning of the data provided (Dayhoff, 1996). The adjustment of the network in order to better match the desired result is known as supervised learning.

The perceptron architecture was soon advanced and saw the addition of multiple layers (Dayhoff, 1996). This type of network is often referred to as the Multi-Layer Perceptron Model or MLP, with the addition of further layers, more weights can be added. This change in structure allows for computational challenges to be solved that require a greater level of processing. Due to the limitations

of the perceptron, however, differing nodes were used that provided a greater level of granularity within their computation. Multilayer neural networks are commonly used in order to solve computational tasks such as optical character recognition (Dayhoff, 1996)

## Activation Functions

The limitations of Rosenblatt's model are largely as a result of the activation function it uses in order to calculate the weighted sum of inputs compared to the threshold value. The perceptron model adheres to a binary activation function and as a result, the weighted sum will either meet the threshold value or it will not. In this case, the neuron will not activate. In 1984, John Hopfield employed the use of the sigmoid threshold function in order to allow a computational unit to process a continuous range of values with a softer a non-linear threshold (Dayhoff, 1996). Nodes that employ the use of the sigmoid activation function are often referred to as sigmoidal units. Figure 4 represents threshold of the sigmoid activation function.

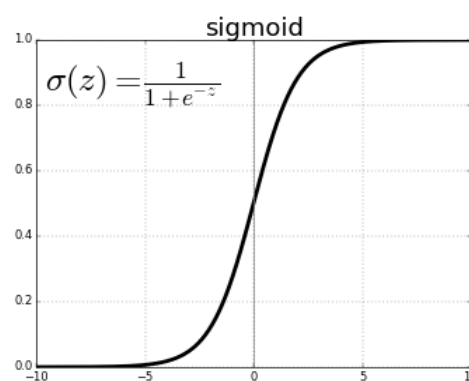


Figure 4: The sigmoid activation function - (Stokastik, 2018)

The use of the sigmoid function allows a node to operate on a soft threshold that is not binary in nature. As a result, real numbers between 1 and 0 can be employed, allowing for a more precise data input. The sigmoid activation function is often defined as an asymptotic squashing function, indicating that the effect of an input on a smaller threshold value is minimised and will not drastically effect the process of any connecting nodes in further layers (Patterson and Gibson, 2017).

One can see why this is an important advantage over the linear step function employed within the perceptron model (Anderson, 2009). The precise nature of the sigmoid activation function allows a more accurate outcome to be utilised by the network. Within this project, the neural network developed will employ the use of the sigmoid activation function in order to allow for the more specific analysis of certain character elements. Other activation functions will also be used in order to compare the effect on network optimisation.

The rectified linear unit (ReLU) can also be adopted by each layer of the neural network, serving to incorporate the activation function of the same name. The rectified linear activation function also possesses a soft threshold that produces a non - linear decision boundary. The implementation of the rectified linear activation function is a prominent within many neural network models, as a result of the threshold generated. The ReLU unit provides a sharp gradient in comparison to the sigmoid hyperplane, serving to ensure that the value remains consistent, a negative aspect of the sigmoid function can occur when a value greater than zero can be classified as '0'. (Patterson and Gibson, 2017).

## Forward Propagation

In order to imagine the flow of data through the neural network that will be created, forward propagation can be analysed. Forward propagation is an important step and concerns the processes carried by each node within the network. The first step of the forward propagation process involves providing the network with input data, serving to generate the activation values of the input nodes (Dayhoff, 1996). As stated by Volna and Kotyrba (2013), the next stage of forward propagation is to ensure that each node within the network can compute an activation function based on the inputs that it receives. In the case of a sigmoid processing unit, the sigmoid activation function will be performed. The result of the function is then set as the activation value. An activation value is sent by all nodes within the current layer to the next for the process to be repeated. Forward propagation is a fundamental functionality for neural networks. (Bishop, 1995).

## Back-Propagation

Perhaps the most important factor in the ability to learn is the back-propagation algorithm. The use of back-propagation allows the network to generate and adjust for the margin of error (Rojas, 2000). The learning procedure that is provided through back-propagation is fundamental, as neural networks are used increasingly in a range of applications (Chauvin and Rumelhart 1995). The back-propagation algorithm was envisaged first by Paul Werbos in 1972. After some iterations, it was eventually presented to a mass audience by Rumelhart and McClelland in 1986 (Dayhoff, 1996). Rumelhart and McClelland (1986) state that the back-propagation algorithm consists of two stages; forward propagation and 'a backwards pass through the network' that serves to compare the output of the processing units with that of the target data. Usually, a neural network that employs the use of back-propagation consists of three or more layers, all of which are fully interconnected.

An important feature of back-propagation is the ability to adjust the weights and parameters of a network to improve performance and better match the desired result when the network is provided with a data point. In order to train a neural network, a training set is implemented and processed in batches, allowing for optimal performance when computing a vast amount of data. A comparison of inputs against the expected outputs allows the network to be trained, obtaining the correct result with greater accuracy and frequency (Narendra and Parthasarathy, 1991). This process is often referred to as optimisation. To define the margin of error when the comparison is made, a cost function is implemented, to define the mean squared error. The effects of optimisation based upon the outcome of the cost function determines the accuracy of the network (Nielson, 2015).

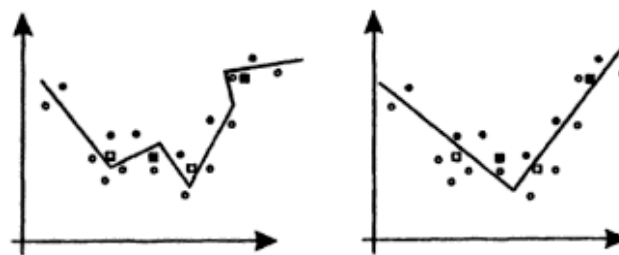
The aim of optimisation is to achieve convergence, where the results of the cost function move closer to '0', indicating that there is greater similarity between the output of the network and the expected result. The cost function is performed after each iteration (epoch) through the network. Under certain circumstances, improvement in accuracy may slow down, as the neural network finds a local minimum. In order to prevent this, a number of steps can be taken (Dayhoff, 1996). For example, increasing the number of hidden nodes allows the network to more broadly explore the search space.

The most common of optimisation techniques is gradient decent. Gradient decent optimises the network through an examination of each item within the dataset, in order to converge at a global optimum. However, the use of this method soon becomes unwieldy for challenges such as image recognition in which a large amount of training data is used (Wang et al., 2017). As a result, stochastic gradient decent (SGD) is often employed, serving to analyse random batches of data.

Through the use of SGD, the network is able to make an approximation about the accuracy of the network. The less precise nature of SGD indicates that a network may see an increased accuracy at a

larger rate. As implied by Dayhoff (1996), moving the network to a different position at a random distance, allows it to escape a local minimum. With the optimum amount of training, the network will be able to find the global optima and the decision boundary will be optimised for the future classification of data that is not present within the training set.

The overtraining of the neural network model can occur in a number of ways. When training on a small dataset, the use of many training epochs can generate a decision boundary that too precisely accommodates the classification of the training set. Although accurate when provided testing values from the same set of data, the trained network model may have substantially less success when classifying new data (Gurney, 1999). The analyses of a large batch of data points within each epoch can also lead to overtraining as a substantial portion of the dataset is likely to be analysed with each iteration. Figure 5 represents the effects of overtraining on the decision boundary in comparison with the decision boundary after optimal training.



*Figure 5: The effect of overtraining on the decision boundary (Left) in comparison with the decision boundary generate as a result of optimal training (Right) - (Gurney, 1999)*

The effects of overtraining are more evident when using a dataset that contains many noisy data points that distract from the trend. This is particularly prominent within financial times series forecasting, in which variations within the data can occur as a result of a range of factors that may not determine the long-term performance of stock price. Within the area of optical character recognition, noisy data may occur if incorrect data is generated with associated labels that do not support the correct outcome.

The implementation of too many hidden nodes can also have a negative effect on over training, serving to process the data to a greater extent that is required. As a result, the trained network will be too exact and will not generalise to the data. Despite the negative effects of a dramatic increase in nodes, the network will train at a faster rate. It is important that the correct balance is found when creating a neural network model (Gurney, 1999).

## Optical Character Recognition

Optical character recognition is a prominent challenge within the computing industry, with the implementation of many techniques used in order to recognise characters. For example, postal services implement OCR in an attempt to quickly sort mail based upon a given address (Downton and Leedham, 1998). One of the earliest and most prominent example of OCR is the Optacon Device, developed by John Linvill and released in 1954. The product provided blind people with the ability to access printed material (Linvill and Bliss, 1966).

Implementing Optical Character recognition within any system requires a range of pre-processing techniques that provide the optimal conditions for classification. Perhaps the most obvious of pre-processing techniques is cropping, ensuring that every character is processed within the same area of

analysis. In a typical cropping process, the centre of the character is found, the image is scaled and the area surrounding the character is cropped (Cheng et al., 1993).

The scale and performance of the neural network is largely effected by the data that it processes. A neural network that uses whole words as data points, must be trained with a dataset of a very large size concerning all of the possible combination of letters that are available. When scaled up to include different handwriting styles, and even languages, the processing demand becomes cumbersome, compared to the logical alternative.

In order to analyse a word, a pre-processing technique known as segmentation can be implemented, serving to find and separate the individual characters within a word. This can be achieved by finding the connecting edges of two joined characters and separating them based upon the areas in which they merge. Another strategy, as adopted by Ming, Liu and Tian (2003), is to define the average length of each character as written by hand. Using this data, a computer program can depict the start and end of the letter 'a', determining the locations in which the image can be cropped and isolated. There are certain circumstances in which segmentation is simpler. For example, numbers are typically written separately and segmentation can be achieved simply through determining the white space between characters.

Noise reduction can also be implemented as a form of pre-processing, and is an important step in achieving ideal conditions for classification. One example of noise reduction serves to deplete light noise from the body of a character. To do so, a value is assigned to all white pixels within the image. Using this data, a function can be computed using the grey pixels within the surrounding in order to minimise the colour of the noisy pixel. This form of noise reduction is known as filtering. (Chaudhuri et al., 2017).

## Competing Products

There are many projects that have explored the use of neural networks to achieve object character recognition. Volna and Koytraba (2013) investigated the topic through the implementation of a vision system that recognises licence plate numbers. The paper expressed the importance of the backpropagation algorithm in order to adjust the weights of the hidden layers within the model, and thus, mitigate the error rate. Concerning a similar topic Zhai, Bensaali and Sotudeh, (2012) state that that neural networks that serve to classify digits use sigmoid nodes in order to generate an output.

Another key aspect of achieving OCR is the segmentation of the key elements within a character. (Budhi, Adipranata 2015) state that, in order to achieve segmentation, an image is divided into several regions based on their "intensity". This serves to clearly define the character that is be computed. The implementation of such techniques, will largely depend on the datasets used and the desired scope of the neural network model.

A range of factors may also hinder the visibility of a character, resulting in a "significant degrading of the feature extraction process". As a result, it may be necessary to convert an image to greyscale and, subsequently reduce noise from the input. This will serve to aid the extraction of the character from the background (Sharma, Chaudry 2013).

According to Tensmeyer, Saunders and Martinez (2017) there is an evident reaction in the success rate of the neural network dependent on the variation of the dataset used to train the network. For example, it was found that the act of lightening or darkening the text could decrease the probability of error when recognising characters.

Another important aspect of the project is establishing which datasets that will be used in order to train the neural network. Nielson (2015) states that the MNIST handwriting dataset offers a varied data sample. The MNIST data set provides a total of 60,000 training patterns to choose from and as a result, will be used in order to train the neural network model (<http://yann.lecun.com/exdb/mnist/>, no date).

## Methodology

In order to implement the neural network model, the python programming language will be used. Chosen because of its functional capabilities and high-level syntax, it also provides access to deep learning libraries such as Tensorflow (<https://www.tensorflow.org/>, no date).

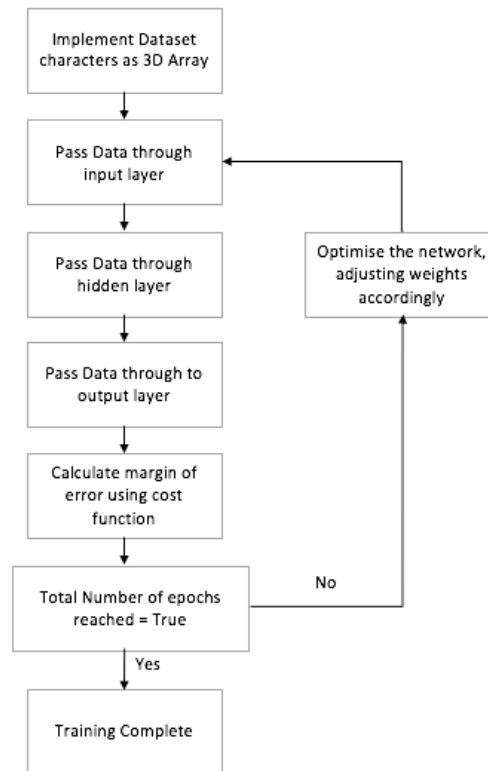
The Tensorflow library will be used within this project in order to incorporate the functionality that it provides, allowing a program to perform computation on vectors of information. This library is of use, for example, when defining the input vector read by the first layer of the neural network. This is an important stage of implementation and allows for the individual characters to be processed. The library will also be used to compute a variety of calculations within each layer and will therefore, be a fundamental component in implementing the back-propagation algorithm.

The MNIST dataset will be used in order gather a range of handwriting samples, for use when training the neural network (<http://yann.lecun.com/exdb/mnist/>, no date). Using Tensorflow, a 28x28 3D array will be created for each character to be fed into the network. Initially, a neural network will be created in order to classify the numerical characters 0-9. As the project develops, a different training set will be used, allowing the program to comprehend letters. This project will also demonstrate the creation of datasets. In order to do so, a range of practical training samples of handwritten characters will be created in a similar environment, ensuring optimal conditions for classification.

A node can consist of a one of many activation functions, generating different outcomes in relation to their threshold value. Different neural networks will be built with the aim of implementing different activation functions and comparing the resulting effect on the learning procedure.

Optimisation is a key element in the creation of a neural network. In order to train the classifier, back-propagation must be employed, allowing the network to adjust the weights and produce the desired results with greater speed and efficiency. In order to achieve the optimum convergence around a global minimum, the network will be adjusted with each epoch based upon the result of a cost function. This will ensure that accuracy is improved as the training set is processed.

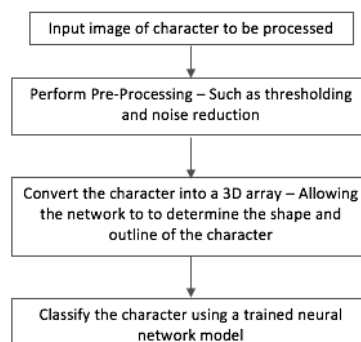
In order to facilitate this convergence, an optimisation method will be employed, such as stochastic gradient decent. The Tensorflow library provides a range of functions that represent the differing methods of optimisation. For example, the 'Adam Optimiser' training algorithm can be applied to implement SGD (<https://www.tensorflow.org/>, no date). Figure 6 describes the training process that will be used by the model.



*Figure 6: Representing the training process to be used by the neural network model*

Upon the completion of training, the neural network model will be used to classify external digits. In order to implement this feature, range of pre-processing techniques will be undertaken in order to ensure that each character can be read in the most optimal situation.

As established within the training method, the neural network recognises the shape and outline of a character through the use of a 3D array. As a result, the input also will be converted, with each element of the matrix corresponding to a position within the image. Each value will indicate the darkness of the area that is represented. The application of this feature will allow the network to determine the outline of the character from the background.



*Figure 7: The classification process of a character within a trained network*

In order to complete the project, the agile development methodology will be implemented. Certain risks may occur during development and as a result, steps must be taken to lessen them. For example,



exterior projects may conflict with the time allocated to complete a certain task. To ensure that this issue is mitigated, a Gantt chart will be created and regularly updated. This will allow for adjustments to be made to any allocated time slot, ensuring that each task is carefully planned and completed to a good standard. The agile methodology provides the best opportunity to achieve the possible and more ambitious project goals that have been set out.

## Ethical Challenges

Although outside of the project scope, training the neural network with a higher success rate, would require the implementation of a diverse dataset. As a result, a range of participants would be required, creating the need to set guidelines in relation to the data collected. The data collected will consist purely of handwriting samples and will not contain any information about the participants. As a result, any data that is not relevant to the learning process of model cannot be implemented. All data will be collected in accordance with the ethics checklist provided in the appendix.

On a broader scale, the application of neural networks in real world applications evoke many conversations concerning ethics and morality. For example, the use of neural networks within autonomous vehicles often comes under scrutiny as a result of the potential risks surrounding the decisions made. Lin, Abney, and Jenkins (2017) state that due to complexity of such systems, it is hard to achieve error-free programming, perhaps leading to incorrect decision making in situations of high risk. Although neural networks and artificial intelligence may be best suited to achieve such autonomy through the act of learning, deficiencies in the learning algorithms or datasets used may cause further ethical challenges. For example, a certain situation may occur that an autonomous vehicle has not been trained for.

The designers of a neural network may also possess certain biases when creating the model. For example, training sets concerning only a select sample of data could be used to achieve expected result. Additionally, the learning algorithms may be unfairly tuned, serving to unethically encourage a certain goal.

Despite the ethical challenges that artificial intelligence and neural networks provide, Anderson, Anderson (2007) argue that training artificial intelligence to follow a moral code may result in more ethical morality than that of a human. This is as a result of certain persuasions or a possible lack of understanding that a human may possess, whereas artificial intelligence may have less difficulty in tackling such situations.

## Implementation

### Implementing the MNIST Dataset

The first step in creating a neural network is to define the model. The model uses a range of established techniques that have been exposed through the research process. The MNIST dataset is implemented in order to learn the characteristics of an individual handwritten digit. Through many iterations, this will improve the accuracy of the neural network as it is able to make more informed choices when classifying a character (<http://yann.lecun.com/exdb/mnist/>, no date).

Through the use of the Tensorflow library, each individual MNIST image is represented as a three-dimensional array (<https://www.tensorflow.org/>, no date). The adoption of this method provides a way for the neural network to distinguish the image from the background and visualize any unique features that may define the character. The background surrounding a character correlates to the

value 0, representing a white section of the data. In order to determine the shape of the character itself, the values '0.1' to '1' are employed. The higher values correspond with the darker areas of the image, as represented in figure 8. For example, the values '0.9' and '0.1' are used to determine the main body and outline of a character respectively (<https://www.tensorflow.org/>, no date).

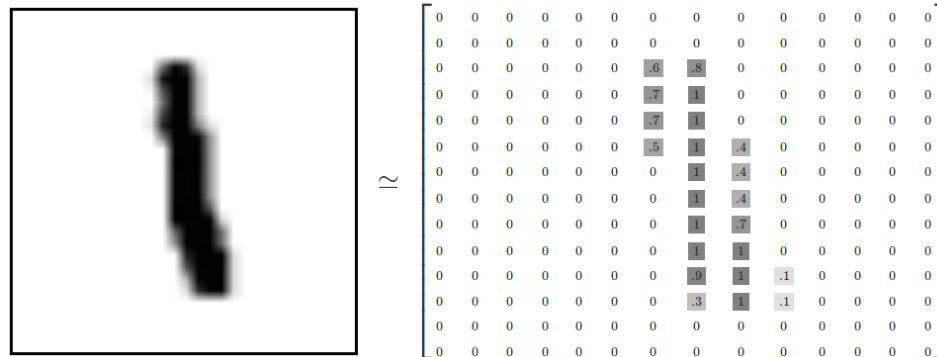


Figure 8: Representing the use of a 3D Array to determine the features of a character (<https://www.tensorflow.org/>, no date)

Each character is defined as a matrix of 28 by 28 (784 values) and is processed alongside a label, matching the character with the value that it represents. The label is used by the neural network in order to determine the margin of error between the judgement made and the expected output. The use of the MNIST dataset represents the implementation of training data, allowing the network to gradually form an understanding of the different characters specified.

## Layers

An important stage in developing a neural network is the implementation of layers, ensuring that the required nodes and weights are defined. Each layer is connected through the nodes that they possess. As a result, the data is able to pass through the network as it is processed (Rojas, 2000). As conveyed by Dayhoff (1996), networks that employ the use of a back-propagation must contain a minimum of 2 layers. As a result, the model implements four layers. The use of a multi-layered neural network allows the network to be more precise when analysing the elements of a character, thus, an improved accuracy is achieved.

Within each layer, a number of nodes are allocated, each serving to iterate upon the processes carried out within the previous layer. The first layer takes the character matrix as input whilst each of the consecutive layers use the previous layer as input. This implementation reduces the possible outcomes as a result of the activation functions used, determining if the threshold has been met and the node is fired. Through the use of this implementation, accuracy is improved with each epoch. Figure 9 represents the implementation of the second layer.

```
layer2 = {'weights': tFlow.Variable(tFlow.random_normal([nodesLayer1, nodesLayer2])),
          'biases': tFlow.Variable(tFlow.random_normal([nodesLayer2]))}
```

Figure 9: The implementation of the hidden second layer, using the output of the first layer as input

The initial weights and biases are randomly assigned a value. As the network is trained, the weights are adjusted, ensuring the margin of error is decreased. Weights are used to control the information as it is processed throughout the network (Skapura, 1996).

## Activation functions

To process the data, an activation function is performed on each node input. The Tensorflow library allows for different activation functions to be applied to the nodes within each layer. Initially, the sigmoid activation function is utilised in order to implement a softer and perhaps more precise threshold (Rojas, 2000). Within Tensorflow, an activation function is performed on the output of a layer. Figure 10 demonstrates the implementation of the sigmoid activation function.

```
firstLayer = tFlow.add(tFlow.matmul(data, layer1['weights']), layer1['biases'])
firstLayer = tFlow.nn.sigmoid(firstLayer)
```

*Figure 10: Representing the use of the sigmoid function within the neural network*

A substantial topic when researching machine learning is the influence that differing activation functions have on performance. Hannun, Mass and Ng (2013, p. 1) explore the differences when comparing the sigmoidal functions with that of other functions. Their findings suggested that sigmoid functions are less capable in areas such as image processing. Throughout the implementation of the neural network, the sigmoid and rectified linear functions will be used with the aim of comparing the corresponding effects on learning accuracy.

## Training with MNIST Data

In order to optimise a neural network for object recognition, back-propagation is implemented. Back-propagation allows for the network to adjust in response to the margin of error calculated at the conclusion of each epoch. The act of propagating through a network in this way is an important component in machine learning.

The neural network implemented within this project implements a cost function in order to calculate the margin of error. The function uses the output generated by the neural network in order to gather the predictions made for each character that has been analysed. Through the comparison of predictions made with the corresponding dataset labels, an average can be produced representing the accuracy of the output. As the network optimises (a technique often referred to as learning) the output of the network moves closer to the desired result. (Dayhoff, 1996).

Through the use of the Tensorflow library, optimisation can be condensed into a single function. After each epoch, the SGD optimiser will be applied to the weighted connections between nodes, allowing the network to adjust. Figure 11 represents the implementation of the SGD optimiser, serving to reduce the margin of error (<https://www.tensorflow.org/>, no date).

```
costOfError = tFlow.reduce_mean(tFlow.nn.softmax_cross_entropy_with_logits(logits = prediction, labels = y))
optimizeCost = tFlow.train.AdamOptimizer().minimize(costOfError) #Stochastic Gradient Decent
```

*Figure 11: The use of the Tensorflow function in order to optimise the network based upon the margin of the error using Stochastic Gradient Decent*

Within this model, the neural network has completed training when each epoch is completed. Figure 12 demonstrates the output of the training model, using MNIST as the training set. For each epoch, a margin of error is displayed. As expected, the margin of error decreases, a result of back-propagation.

Upon the completion of training, the model makes a range predictions concerning the MNIST test set and a final accuracy score for the network is shown.

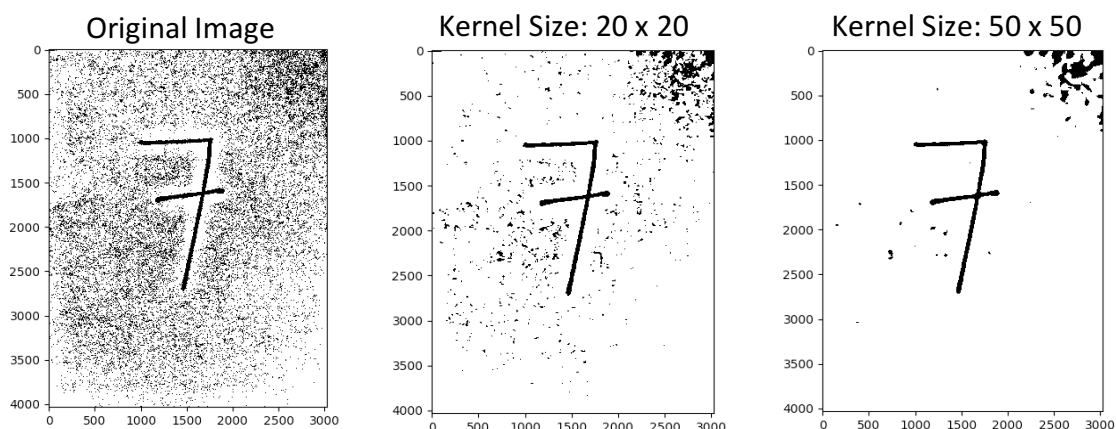
```
[(tensorflow) Daniels-MBP:~ danielldages$ cd Desktop
[(tensorflow) Daniels-MBP:Desktop danielldages$ python NN.py
Extracting MNIST_data/train-images-idx3-ubyte.gz
Extracting MNIST_data/train-labels-idx1-ubyte.gz
Extracting MNIST_data/t10k-images-idx3-ubyte.gz
Extracting MNIST_data/t10k-labels-idx1-ubyte.gz
2018-02-02 16:15:35.357050: I tensorflow/core/platform/cpu_feature_guard.cc:137
Your CPU supports instructions that this TensorFlow binary was not compiled to
use: SSE4.2 AVX AVX2 FMA
('Epoch:', 1, 'out of', 10, ' Margin of Error:', 1565900.2662658691)
('Epoch:', 2, 'out of', 10, ' Margin of Error:', 375029.9978027344)
('Epoch:', 3, 'out of', 10, ' Margin of Error:', 200936.58493447304)
('Epoch:', 4, 'out of', 10, ' Margin of Error:', 115830.83323440207)
('Epoch:', 5, 'out of', 10, ' Margin of Error:', 68563.17012703419)
('Epoch:', 6, 'out of', 10, ' Margin of Error:', 44563.17710939795)
('Epoch:', 7, 'out of', 10, ' Margin of Error:', 27902.38223014772)
('Epoch:', 8, 'out of', 10, ' Margin of Error:', 21685.714144531594)
('Epoch:', 9, 'out of', 10, ' Margin of Error:', 21218.484682902694)
('Epoch:', 10, 'out of', 10, ' Margin of Error:', 19549.50763696246)
('Accuracy:', 0.9489)
```

*Figure 12: Representing the output of the training model*

## Pre-Processing

To ensure the best conditions for classification, a range of pre-processing techniques are employed. The clarity of the image that is being classified can have a substantial effect on the result, such as an interfering pen stroke or light pollution that serves to deplete the sharpness of the character. To implement the pre-processing techniques, the scikit-image library is utilised (Scikit-image.org, 2018).

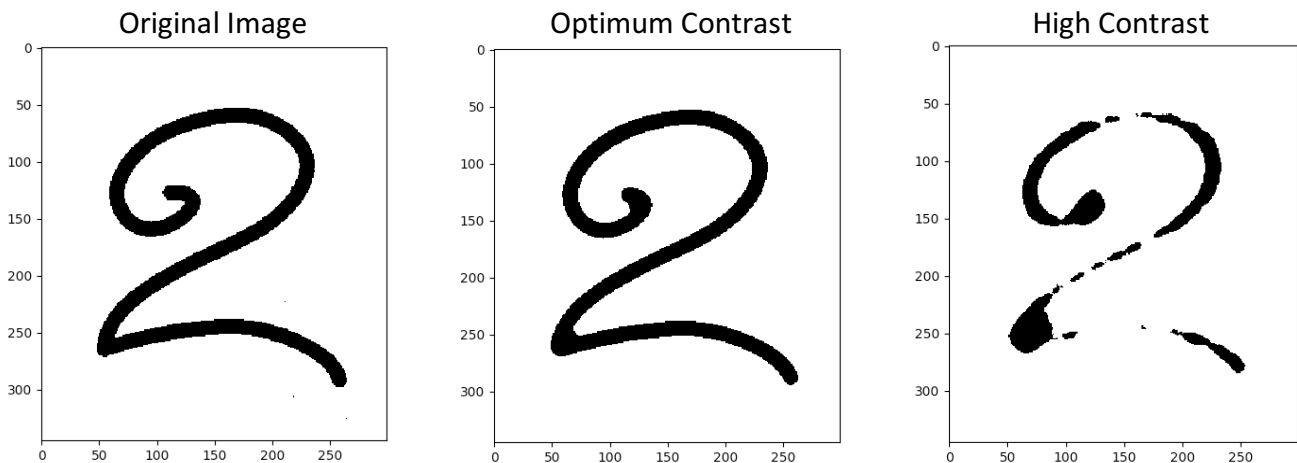
The chosen images consist of handwritten digits that have been saved as a '.jpeg' file. The script created serves to collect the images and implement a range of functions. The application of noise reduction serves to eliminate grain and pixilation from the image using a median filter. The kernel size chosen for the median filter can have a substantial effect on the outcome. If too small, there will be little changes to the image. Implementing a kernel size that is too large will serve to deplete the edges of the character, blurring the pixels with the surrounding are. Figure 13 represents the effects of the median filter in comparison with the original image.



*Figure 13: Representing the improvement of image clarity through the use of the median filter.*

As demonstrated, the use of the median filter has a substantial effect on the quality of the image, providing a clear reduction in image noise with a greater kernel size. The noise reduction function

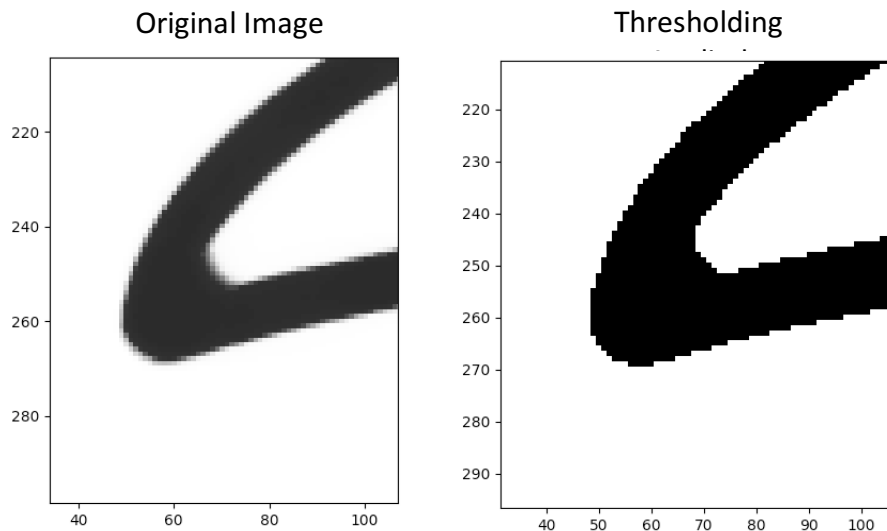
implemented can also serve to intensify the contrast of the image through increasing the richness of noise. This functionality serves to make the light and dark areas less diverse in range. This can have a positive effect on the clarity of the image, provided there is no undesirable noise within the image. However, increasing the contrast in conditions of less clarity will simply serve to magnify the noise. Figure 14 demonstrates the effect of increased contrast, in comparison with the original image. Within this example, a thresholding filter has been implemented to more clearly detail the results.



*Figure 14:: Representing the effects of differing contrast levels in comparison with the original image.*

As determined in figure 12, the contrast level can have a substantial effect on the clarity of the image and the conditions for performing OCR. There is an evident improvement when using the optimum contrast level, with more defined edges that allow for a clearer model of the character. The implementation of this level of contrast can have a negative effect during processing, drawing more attention to the undesired hook at the top of the character. The effects of a very high contrast can eliminate the vital structure of the character and provides the classifier model with little chance of accuracy. This is as a result of the greater weight applied to the white pixels.

Thresholding has also been applied to pre-processing, serving to replace any colours that are not within the true black or white range. Thresholding is applied after a greyscale filter is implemented, changing all grey values to either white or black, dependent on the pixel values. The use of thresholding has a substantial effect on the clarity of the image serving to add more weight to the structure of the character. Figure 15 represent the improvement in weight and clarity when thresholding is applied.



*Figure 15:: Representing the effects of thresholding on the sharpness and clarity of the image*

Despite the advantages of thresholding, in improving the clarity and sharpness of the image, there is a clear effect on the concavity of the character. Concavity refers to the roundness of the digit and the corresponding edges. If the threshold value is too high, the structure of the shape may become less prominent and hard to classify. The use of a practical threshold value ensures that the details are maintained whilst improving upon the clarity of the character for classification. Each character that has been created or processed within a different environment may require the application of pre-processing techniques at diverse levels of granularity.

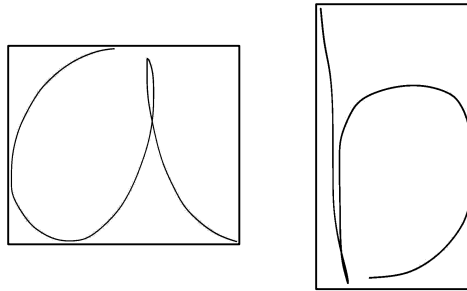
The final stage of pre-processing concerns optimising the processed image for classification. The neural network model utilises the same format used by the MNIST dataset. As this format is used for training with the dataset, the same approach must be used when processing an image for classification. In order to adhere to this blueprint, a new template image is created, with the dimensions 28 by 28 pixels. The original image is then placed within the template image, ensuring optimal conditions for classification (<http://yann.lecun.com/exdb/mnist/>, no date).

## Dataset Creation and Training

The dataset is an important aspect of the neural network paradigm, determining the type of data and the number of data points that will be used in order to train the model. A dataset contains a range of data points combined with an associated value, representing the desired result. The greater the number of combinations, the more optimal the dataset is, reducing the possibility of overtraining.

In order to implement a dataset, a range of techniques are employed, ensuring that each data point is generated within the most optimal and equal conditions. The dataset created handles a total of 106 data points concerning the characters 'a' and 'b', with 53 data points allocated to each letter. The characters, represented as images, have each been drawn within the same environment, ensuring similar conditions for pre-processing. As a result, the pre-processing script can be implemented with the same settings.

Dependant on the images used, different pre-processing techniques are required. When constructing large datasets, in which there are many sources and production environments, a greater range of differing pre-processing techniques must be employed. Figure 16 represents an example of the pre-processed handwritten characters used as data points.



*Figure 16: Representing an example of the 'a' and 'b' character data points used for training*

Following the implementation of pre-processing techniques such as noise reduction and grey scaling, the dataset is prepared using the MNIST specification of 28 by 28 pixels. This creates an input vector of 784 elements when processed by the neural network model.

A vital aspect of supervised learning is the introduction of labels that correspond to each individual data point (Dayhoff, 1996). For each character within the dataset, data is to be provided that details the value that is represented. The image data and corresponding label is defined in two separate arrays and combined as a dataset using a range of Tensorflow functions. Following the creation of the dataset, the data points are shuffled, creating more variation in the data that is analysed during each epoch. Figure 17 represents the function created in order to group the data and labels in preparation for the creation of the dataset (<https://www.tensorflow.org/>, no date).

```
def defineValidationData(imageList, labellist):
    directoryAVal = os.listdir(ValAPath) #Define route to character 'a' dataset
    directoryBVal = os.listdir(ValBPath) #Define route to character 'b' dataset
    for file in directoryBVal: #Gather character 'a' dataset data
        if file.endswith('.png'): #Avoid D.S Store and any unwanted files
            img = os.path.join(ValBPath, file) #Define image
            image = io.imread(img) #Read and append
            imageList.append(image)
            labellist.append(1) #Create desired outcome label at corresponding position
    for file in directoryAVal: #Gather character 'b' dataset data
        if file.endswith('.png'): #Avoid D.S Store and any unwanted files
            img = os.path.join(ValAPath, file)
            image = io.imread(img) #Read and append
            imageList.append(image)
            labellist.append(0) #Create desired outcome label at corresponding position

    return imageList, labellist
```

*Figure 17: Demonstrating the function created in order to create and group the data and labels.*

The neural network model uses batches in order to analyse a subset of the MNIST dataset for each epoch. The number of data points analysed for each epoch has a large effect on the efficiency and effectiveness of back-propagation. The greater the batch size, the more data points can be analysed in order to adjust the network. As a result, the trained network will be more accurate as a wider range of factors are considered.

Despite the benefits of a batch processing, incorporating a large batch size can also lead to overfitting of the training data, generating incorrect results when classifying new data. The dataset used to train the network has a small number of data points compared to the MNIST dataset. As a result, a smaller batch size is required with the aim of finding the optimum number of characters to analyse during each epoch, to increase variation and reduce overtraining.

Following the implementation of the dataset, a test set of 20 data points is also formulated, to calculate an accuracy score based on the predictions generated. As with the initial training set, the test set is randomised in order to ensure a fair environment when classifying the example data.

## Classification

Implementing the model for classification is a simple process. The model is saved upon the completion of the training process, using the Tensorflow 'saver' function. This serves to retain the weight values that have been adjusted as a result of the back-propagation algorithm. Through the implementation of the saved network, the desired character can be pre-processed and entered into the network for classification. Upon the completion of the classification procedure, the output of the trained model is presented.

## Results

In order to determine the effectiveness of neural networks used achieve optical character recognition, a range of tests can be carried out. For each test, a number of parameters will be adjusted with the aim of investigating the performance of the training and classification procedures.

### Test Plan

To determine the success of the training procedure under different constraints, a range of parameters will be adjusted such as the number of nodes used. Such components have a substantial effect on the training procedure. With various sources detailing possible generalisation or overtraining in relation to the dataset used, testing will be carried out using both MNIST dataset and the newly created dataset including the characters 'a' and 'b'.

The implementation of the sigmoid and rectified linear activation functions will also be analysed, determining the effects of each function on the decision boundary generated. There will be a focus on the ability of both models to achieve optical character recognition. Throughout testing, Stochastic Gradient Decent will be employed as the optimisation technique.

### Training

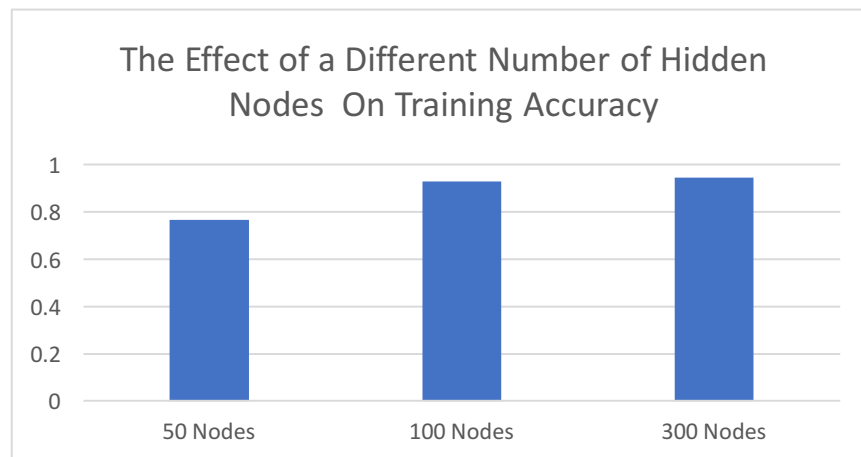
Perhaps the most important aspect of a neural network is the ability to train, serving to determine the success of the network model when classifying a character. A range of factors can serve to affect the success of the back-propagation procedure, such as the number of nodes used and the quantity of epochs performed.

The first test will analyse the effects of the number of nodes when training the network. A range of sources, such as Gurney (2009), suggest that the performance of the back-propagation procedure is largely dependent on the number of nodes that is available within each layer. In order to test the model, a total of 10 epochs and a batch size of 100 will be used whilst the MNIST dataset will be used to train the network. Each layer will adopt the use of the rectified linear activation function. To gather the accuracy score, the MNIST test set will be used, serving to analyse the predictions made on each of the data points.

As represented within figure 18, the number of nodes implemented has a dramatic effect on the performance of training and the subsequent accuracy of the network model. When 50 nodes are employed within each hidden layer, the accuracy of the network is clearly underwhelming. The implementation of a small number of nodes serves to hinder the number of data points that can be analysed with each epoch. As a result, other parameters such as the number of epochs and the batch size used must be increased.

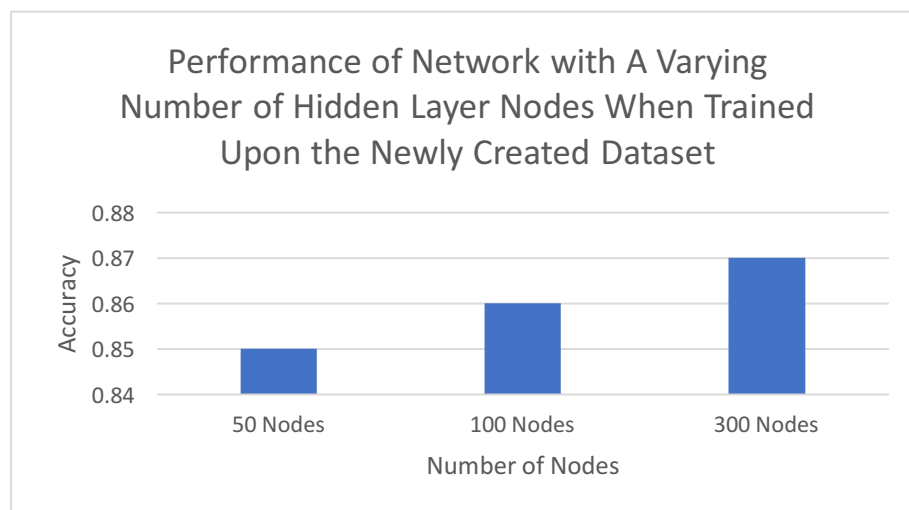


As the number of nodes increases, the accuracy of the network improves, allowing the model to analyse a larger number of data points. The analysis of a greater range of data, allows the training procedure to comprehend a large variation of hand-written digits. This results in a greater performance when classifying the characters within the test set. The improvement made to the training procedure is clearly demonstrated through the use of 100 and 300 nodes per hidden layer, with a dramatic increase in the accuracy achieved.



*Figure 18: Representing the effects of a different number of nodes upon training accuracy.*

The trend continues when implementing the smaller dataset of 106 data points, concerning the characters 'a' and 'b'. Figure 19 represents the similar results generated when implementing a different number of hidden nodes when training upon a the newly created dataset



*Figure 19: Performance with a varying number of hidden layer nodes, a dataset of 106 data points.*

The next test investigates the performance of the training procedure with a varying number of epochs. It is estimated that a larger number of epochs will see an increase in performance, as the network over-trains to a wider range of data. The the MNSIT dataset was used for this test, analysing the performance of the network with 5, 10, 20, and 50 epochs respectively.

Within each test, a batch size of 100 is used and the total number of nodes within each hidden layer is 300, the optimum found within the previous test. Figure 20 details the effect of a varying number

of epochs on the training procedure. The results represent a clear improvement in performance with the greater epoch size, allowing the network to process a wider range of data points. This is detailed in the definitive increase in accuracy when the network is verified against the test set.

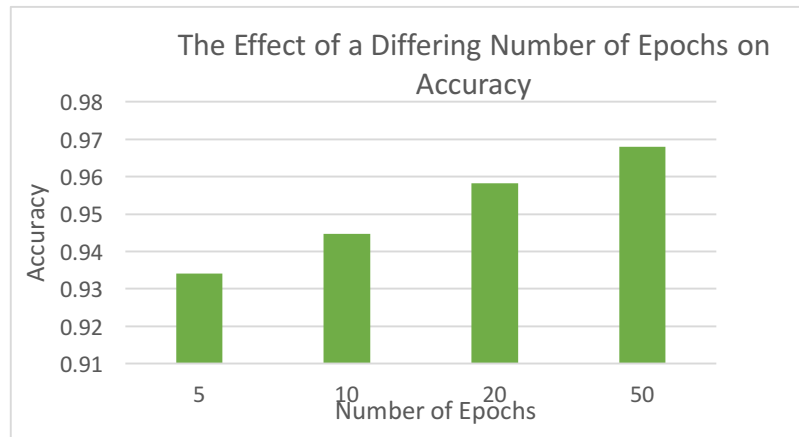


Figure 20: Representing the effects of a different number of nodes upon training accuracy

The number of epochs has a more substantial effect on the model that implements the newly created dataset of 'a' and 'b' characters. Figure 21 represents the performance of the training procedure when using 10, 50, 100, 200 and 1000 epochs. It is once again clear, that the greater number of epochs that are employed, the more accurate the network is when examined against the test data. However, the use of 500 and 1000 epochs have a very negative effect on the training procedure. It is clear that the model has over trained to the data. As a result, the network performs badly when classifying the previously unseen characters within the test set. These results demonstrate the negative impact of overtraining when attempting to obtain optimum accuracy.

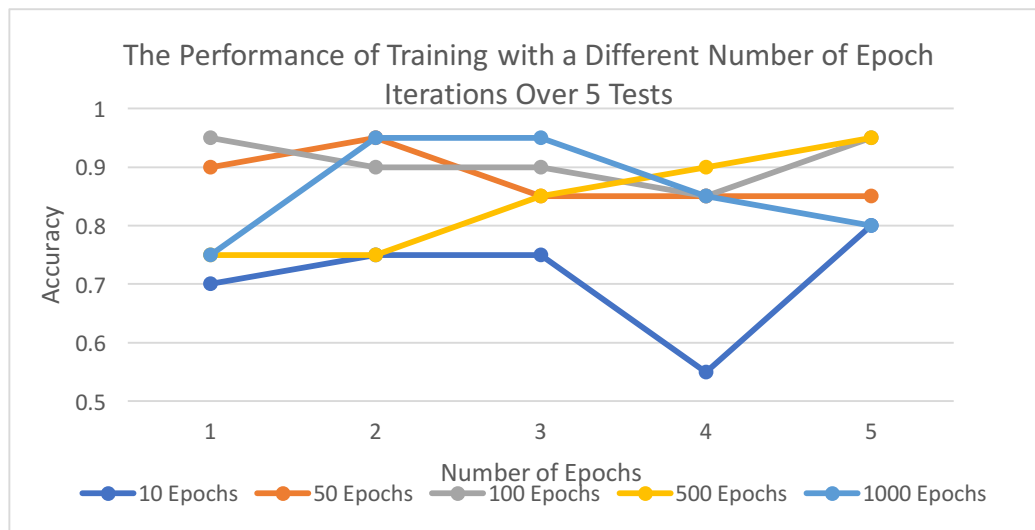


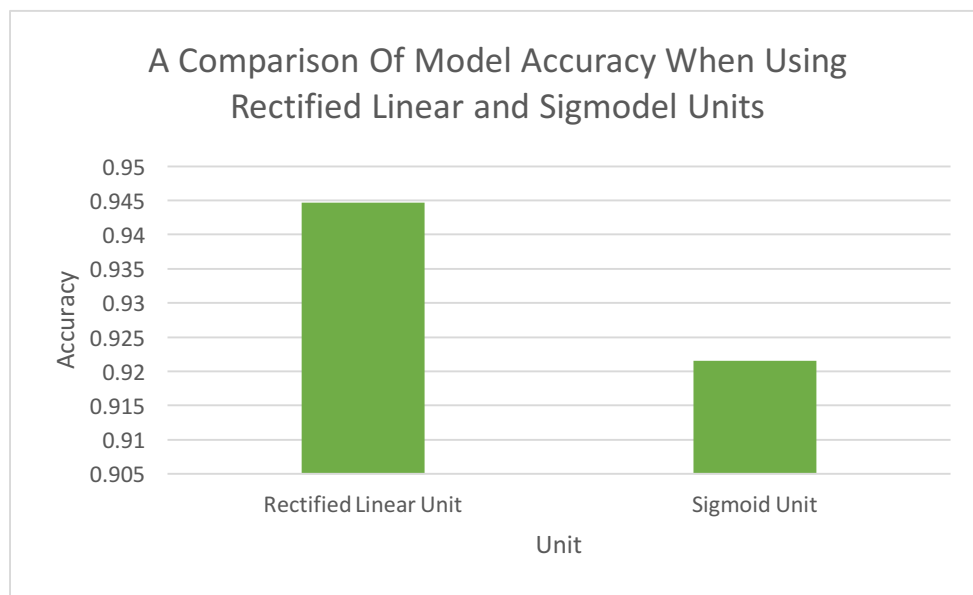
Figure 21: Performance when trained with a different number of epochs over 5 tests

The activation function employed by the network has a substantial effect on performance, as a result of the decision boundary that is generated. The sigmoid function is heavily used within the area of optical character recognition. This is a consequence of the ability to represent the values of the input vector, using a soft threshold between 0 and 1 (Dayhoff, 1996). Despite this, the use of the sigmoid unit also possesses disadvantages, with the gradient occasionally serving to minimise low inputs as the value 0. According to Patterson and Gibson (2017), the rectified linear activation units overcome

this problem through the use of a steep gradient that ensures that values above zero are set as constants.

To determine the effects of the sigmoid and Rectified Linear activation function, the sigmoid and ReLU nodes will be incorporated into the network, and the learning procedure will be monitored. Within both tests, the MNSIT dataset will be used, with a total of 10 epoch and a batch size of 100. Figure 22 represents the performance comparison of the learning procedure when incorporating ReLU and sigmoidal nodes.

The implementation of the Rectified Linear activation function clearly produces a higher training accuracy. This is largely as a result of the vanishing gradient problem, that is common when using sigmoidal units (Patterson and Gibson, 2017). The sharp gradient that is present within the rectified linear function ensure that the input is not minimised and remains constant.



*Figure 22: The effect of the ReLU and Sigmoidal units on the accuracy of the neural network model*

## Classification

The success of the classification procedure is largely dependent on the accuracy of the model and the size of the dataset used. Through incorporating the MNIST dataset and training with optimal conditions. The network is able to classify a range of handwritten digits with a high rate of success. In order to test the success of the classification procedure, a model will be trained with parameters aimed at achieving optimal accuracy. Each character that is provided to the network is optimised through the use of Pre-processing, ensuring the optimal and most equal conditions for categorisation. Table 1 represents the parameters used in order to achieve optimal accuracy, when using the MNIST dataset.

Table 1: Representing the parameters used when trained with the MNIST dataset

Parameter	Value
Number of Layers	4
Number of Input Layer Nodes	500
Number of Node per Hidden Layer	300
Number of Epochs	10
Activation Function	Rectified Linear Function
Batch Size	1000
Accuracy Achieved	95.6%

The model had a varying rate of success when classifying handwritten digits. In many cases the data provided was incorrectly classified despite the high accuracy archived by the network when analysing the test set. This is the result of a range of factors. The volume of the MNIST dataset created difficulty when attempting to train upon the wide range of character samples available.

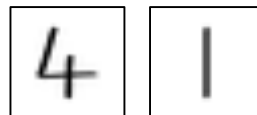
There were also apparent challenges when pre-processing the data. It was found that incorporating a noise reduction filter would often remove some necessary characterises within the main body of the digit, as a result, classification is sometime restricted. In order to overcome this challenge, the kernel size of the filter was reduced. Figure 23 represents an example of the errors made during the classification procedure. The misclassification of the digit '9' as the digit '1' can be comprehended as a result of the large stem that rises from the baseline. The similarities also extend to the narrow loop that extends from the stem, a detail that could be mistaken for the crossbar present in the predicted digit.



```
Predicticted Digit: 1  
(tensorflow) Daniels-iMac:OCR-Project daniellages$
```

Figure 23: Representing the un-successful classification of the digit '9'

Despite the challenges faced, the trained MNIST classifier is able to determine the characteristics of a range of handwritten digits between '0' and '9'. Figure 24 represents an example of the successful output achieved when classifying the characters '4' and '1'.



```
Predicticted Digit: 4  
Predicticted Digit: 1  
(tensorflow) Daniels-iMac:OCR-Project daniellages$
```

Figure 24: Representing the successful classification of the digits '4' and '1'

There are many ways in which the performance of the classification procedure can be improved. For example, through more closely matching the input images with the format of the dataset, the model

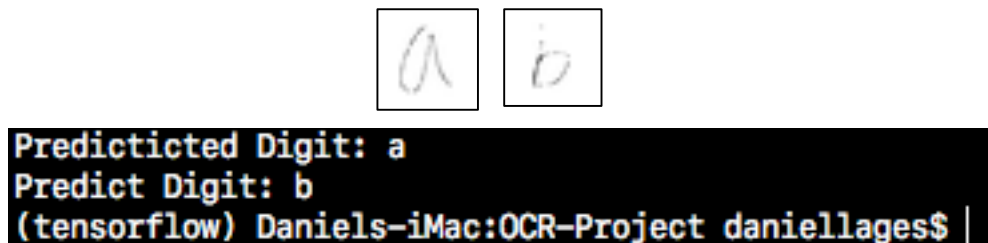
will be able to better differentiate the characteristics of differing digits. This is conveyed, through the high accuracy rating achieved when analysing the test set.

When implementing the newly created dataset concerning the characters 'a' and 'b', the model had a higher rate of success. The pre-processing procedures implemented with the training and test sets were also applied to the characters provided for classification. This ensured an optimal environment for achieving OCR, allowing for clarity when identifying the characteristics of each data point. Table 2 represents the training parameters used to achieve optimal accuracy when training upon the dataset of 'a' and 'b' values.

*Table 2: Representing the parameters used when trained with the newly created dataset*

Parameter	Value
Number of Layers	4
Number of Input Layer Nodes	500
Number of Node per Hidden Layer	300
Number of Epochs	200
Activation Function	Rectified Linear Function
Batch Size	100
Accuracy Achieved	94%

It is evident that achieving a high rate of success when classifying handwritten characters, requires a generalisation to the dataset used to train the network. Furthermore, through careful consideration of the pre-processing techniques used, the decision boundary generated by the network will be able to determine the features of a character with greater detail. Figure 25 demonstrates the successful classification of the characters 'a' and 'b'.



*Figure 25: Representing the successful classification of the characters 'a' and 'b'.*

## Conclusion

Neural networks, a prominent component within technological landscape, are a useful paradigm for classification challenges (Dayhoff, 1996). The implementation of concepts such as back-propagation allow for training procedures that possess a success rate unavailable to other computational methods. This project successfully explores how neural networks can be used in order to achieve optical character recognition.

Within the early developments made in this area, perceptron models provided a novel way to classify data. The addition of features such as input weights serve to manipulate the output of each node in order to achieve a desired outcome. The adjustability of a neural network allows for training to occur through the use of the back-propagation algorithm. This concept, a significant development in classification, serves to provide the ability to develop an understanding of a range of data points with a high level of accuracy. When trained, the model can be used in order to classify data of the same type.

This project has explored the importance of correctly implementing the training procedure, determining that a low accuracy of classification occurs when the decision boundary does not generalise to the training set. For example, when the network is over trained, the decision boundary is too exact in relation to the dataset, often leading to the miss-classification of data in the real world. Conversely, when the network is not trained to the required level, the characteristics of each data point are not considered sufficiently. Adjusting training parameters such as the number of epochs, batch size and the activation function used can serve to negate this challenge and improve the training procedure.

A range of steps can be employed in order to prepare a character for classification. Referred to as pre-processing, images of handwritten characters can be manipulated to ensure optimal clarity. Noise reduction, for example, can serve to eliminate light pollution from the characters, making the body of the character more defined. Other prominent techniques can also be implemented, such as grey scaling and thresholding, serving to better distinguish the character from the background.

Such pre-processing techniques can be applied to the individual characters within a training dataset. This project has successfully demonstrated the requirements of a dataset and how they can be implemented in order to benefit the learning procedure. Each data point consists of a character and an associated label representing the desired output. The implementation of a test set also has a large significance, serving to calculate the accuracy achieved by a trained network and allowing for the appropriate changes to be made to the learning parameters. Through the creation of a dataset, this project has aptly detailed the fundamental aspects of implementing a dataset with the aim of achieving OCR.

A neural network model has been successfully implemented that is able to process a large number of data points, such as the MNIST dataset of handwritten numbers (<http://yann.lecun.com/exdb/mnist/>, no date). The model is able to achieve a high rate of accuracy, demonstrating that the parameters required for accurate classification vary depending on the volume and complexity of the data being processed.

In order to determine the accuracy that can be generated using the training procedure, a range of neural network implementations have been compared, determining the effects of different training parameters. For example, through examining the effects of the sigmodal and ReLU units, the optimal activation function for a given classification task can be applied. As a result, the network is able to correctly the classify the individual characters that it has been provided with a high rate of success.

## Critical Evaluation

Throughout the completion of this project, a range of strategies have been employed, allowing for the best conditions at each stage of completion. In order to gain feedback on the progress made, regular meetings were held with the project supervisor. At each meeting, a progress report was provided and any suggestions for improvement were welcomed. It was of importance that the project meetings were recorded, allowing for any feedback to be carefully collected and considered. All arranged project meetings were attended, with the aim of ensuring that the project was progressing to a high standard.

In order to implement the project successfully, the agile methodology was adopted. The presence of regular meetings allowed for the implementation of new stages of development, quickly followed by relevant and consistent feedback. This permitted any relevant improvements to be made. The adoption of the agile methodology also encouraged the evaluation of work completed at every stage, serving to ensure a good level of quality. All meetings are detailed in the appendix (Lages, 2017).

Most of the aims set out have been achieved. Despite this, the presence of other assignments has led to the prioritisation of some of the more relevant and noteworthy subject areas. The implementation of a stronger focus on select topics has ensured a higher level of detail within the areas that have been realised. With more time, areas such as segmentation and post-processing would be explored at a greater length, alongside the creation of a larger dataset.

## Future Work

Through the research carried out, it has been determined that segmentation is a major area of Pre-processing. Future work could be carried out in this area, with the aim of exploring the methods that can be used to separate the letters within a word and process each character individually. The investigation into this topic could extend to the classification of a larger body of text, such as sentences and whole documents.

Finally, post-processing procedures could be implemented in order to better improve classification. For example, a dictionary system could be implemented to improve the categorisation of words and larger bodies of text.

## Bibliography

- Skapura, D. (1996) Building Neural Networks. Massachusetts: Addison Wesley.
- Fiesler, E. (1992) Neural Network Formalization. IDIAP, pp. 3-6.
- Dayhoff, J. (1996) Neural Network Architectures. Boston: International Thomson Computer Press.
- Sharma, A. and Chaudhary, R. (2013) 'Character Recognition Using Neural Network', International Journal of Engineering Trends and Technology, 4(4), pp. 662-667
- Phangtrastum M., Harefa, J. and Tanoto, D. (2017) Comparison Between Nueral Network and Support Vector Machine in Optical Character Recognition. Procedia Computer Science, 116, pp. 351-357
- Penrose, R. (1999). The emperor's new mind. Oxford: Oxford University Press.
- Stokastik. (2018). Machine Learning Interview Questions and Answers (Part I). [online] Available at: <http://www.stokastik.in/machine-learning-interview-questions-and-answers-part-i/> [Accessed 20 December. 2018].
- Anderson, J. (1995). An introduction to neural networks. New Delhi: Prentice Hall India.
- Patterson, J. and Gibson, A. (2017). Deep learning. O'Reilly Media, Inc, pp.69-70.
- Hu, Y. and Hwang, J. (2001). Handbook of neural network signal processing. Boca Raton: CRC Press.
- Dunne, R. (2007). A statistical approach to neural networks for pattern recognition. Hoboken: Wiley.
- Minsky, M. and Papert, S. (1988). Perceptions. An introduction to computational geometry. Cambridge: The MIT Press.
- McClelland, J. and Rumelhart, D. (1986). Parallel distributed processing. Cambridge, Mass: MIT Press.
- Rojas, R. (2000). Neural networks: A systematic Introduction. Berlin: Springer.
- Narendra, K. and Parthasarathy, K. (1991). Gradient methods for the optimization of dynamical systems containing neural networks. *IEEE Transactions on Neural Networks*, 2(2), pp.252-262.
- Downton, A. and Leedham, C. (1998). Pre-processing of envelope images for optical character recognition. [1988 Proceedings] 9th International Conference on Pattern Recognition.
- Linvill, J. and Bliss, J. (1966). A direct translation reading aid for the blind. *Proceedings of the IEEE*, 54(1), pp.40-51.
- Cheng, T., Khan, J., Liu, H. and Yun, D. (1993). A symbol recognition system. *Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR '93)*.
- Ming, D., Liu, J. and Tian, J. (2003). Research on Chinese financial invoice recognition technology. *Pattern Recognition Letters*, 24(1-3), pp.489-497.



Chaudhuri, A., Mandaviya, K., Badelia, P. and K Ghosh, S. (2017). Optical Character Recognition Systems for Different Languages with Soft Computing. Cham: Springer International Publishing.

Wang, L., Yang, Y., Min, R. and Chakradhar, S. (2017). Accelerating deep neural network training with inconsistent stochastic gradient descent. *Neural Networks*, 93, pp.219-229.

Volna, E. and Kotyrba, M. (2013). Vision system for license plate recognition based on neural networks. 13th International Conference on Hybrid Intelligent Systems, pp.140-143.

Zhai, X. , Bensaali, F. and Sotudeh, R. (2012). OCR-based neural network for ANPR. *IEEE International Conference on Imaging Systems and Techniques Proceedings*, pp.393-397.

Bishop, C. (1995). *Neural Networks for Pattern Recognition*. Gloucestershire: Clarendon Press.

Nielsen, M. (2015) *Using Neural Nets to Recognize Handwritten Digits*. Available at: <http://neuralnetworksanddeeplearning.com/chap1.html> (Accessed 21 October. 2017).

Nielsen, M. (2015) *Using Neural Nets to Recognize Handwritten Digits*. Available at: <http://neuralnetworksanddeeplearning.com/chap2.html> (Accessed 15 November. 2017).

Gurney, K. (1999). *An introduction to neural networks*. London: Routledge, p.82.

LeCun, Y., Cortes, C. and J.C. Burges, C. MNIST handwritten digit database <http://yann.lecun.com/exdb/mnist/> (no date) (Accessed 20 October. 2017).

Tensmeyer, C. Saunders, D. and Martinez, T. (2017). *Convolutional Neural Networks for Font Classification*. Computing Research Repository.

<https://www.tensorflow.org/> (no date) (Accessed: 15 October.2017).

Chauvin, Y. (ed.), Rumelhart, D. (ed.) *Backpropagation: Theory, Architectures and Applications*. Psychology Press

Lin, P., Abney, K. and Jenkins, R. (2017). *Robot Ethics 2.0*. Oxford: Oxford University Press.

Anderson, M., Anderson, S. L. (2007). *Minds and Machines*, 17(1), pp.1-10

Maas, A.L., Hannun, A.Y., Ng, A.Y. (2013) 'Rectifier nonlinearities improve neural network acoustic models', *Proceedings of ICML*, vol. 30, p.1

Scikit-image.org. (2018). scikit-image: Image processing in Python — scikit-image. [online] Available at: <http://scikit-image.org/> (Accessed 9 Mar. 2018).

Lages, D. (2017). *Using Neural Networks for Handwritten Character Recognition - Project Blog*. [online] *Using Neural Networks for Handwritten Character Recognition*. Available at: <https://danlagesocr.wordpress.com/> [Accessed 4 Oct. 2017].

# Appendix

## Project Meeting Records

All meetings are present at <https://danlagesocr.wordpress.com/> (Lages, 2017).

### **Meeting 1**

Date: 4/10/2017

Meeting Time: 1hr

The first meeting with my supervisor took place on the 4th October 2017. During the meeting, we had discussed a range of topics.

We initially discussed how we would track my progress during the project. I found that the best way to track the progress made would be to keep a log. This would also apply to keeping track of any changes to the project as progress is made. The supervisor suggested that a blog should be used in order to achieve this. I agreed with this, as a result, I set a blog later that day with the aim of creating a platform for tracking my progress throughout the project.

We also talked about the objectives that I needed to create for the project. I wanted to know if the objectives needed to be a broader view of the topic or should be more focused on the program I was hoping to create.

Following our discussion, I learned that the objectives should cover a broad view of the topic whilst maintaining a focus on the elements relevant to implementing a program that achieves character recognition. I feel that this approach also best reflects the project title.

I also asked the supervisor if he had any recommendations on what programming language he suggested I use in order to implement the project. He recommended that I use Python. This was in line with the research that I have completed so far.

This concluded the meeting.

### **Meeting 2**

Date: 11/10/2017

Meeting Time: 45 Minuets

During my second meeting with my supervisor, I talked about the research I had carried out so far in relation to neural networks. I mentioned that I had looked at the basic structure of neural networks and the 'perceptron nodes' in order to gain a basic understanding of how they work and are useful in relation to neural Networks.

Alongside this, I had also mentioned that I had created my blog and reached a final decision on the objectives that will serve to define the scope of my project.

As a result of listening to another student as he talked to the supervisor, I realised the importance of using a relevant programming library in order to build my neural network. As a result, I determined that this would be a focus when researching.

This concluded the meeting.

### **Meeting 3**

Date: 18/10/2017

Time: 13:00

Location: TRJ237

General Points:

- The Supervisor provided examples of other blogs that have been used in order to track the progress of a project. He explained that in order to ensure clarity, a more structured layout should be implemented when writing the blog posts.
- The supervisor also stressed the importance of the looking out for LSEPI (Legal, Social, Ethical and Personal Issues). This will feature within my final report in order to illustrate the ethical practices undertaken.

Progress Made this week:

- This week I have focused heavily on research. I have gained a greater understanding of neural networks and how they function. This included gaining a further understanding of weights and biases and how they are used to tune the outcome of a neural network
- The main focus of my research involved determining which programming language and libraries I will be using for this project. As mentioned to my supervisor, I have decided to use the Python programming language and the Tensorflow library (<https://www.tensorflow.org/>, no date).
- The supervisor agreed that Tensorflow is a good library for this project.
- I have also determined that I will be using the MNIST dataset of handwritten characters in order to train the Neural network.

Plans for the near future:

- As mentioned to my supervisor, I will complete the installation of the Tensorflow library in order to begin a practical look into neural networks.
- I also aim to carry out some more research with my focus being on papers that could provide more insight into the implementation of the Neural network.

- As Milestone 1 is coming up in the next month, I will begin planning and implementing in order to complete the document as required.

## Meeting 4

Date: 25/10/2017

Time: 13:00

General Points:

- The supervisor suggested that any programming that is required for the project should be implemented as I progress. This is to ensure that there is not too much programming to carry out towards the end. I agreed and, as a result my gantt chart reflects this.
- The supervisor mentioned that if at any point, licenced software be required for our project, the university may be able to contact the creators of the software for use. This may be of note during my assignment.
- The supervisor again recommended that we look at the LSEPI rules and how they may apply to our project. As a result, I have set as an aim for completion this week.

Progress Made This Week:

- One of the main tasks for me over the last week was to install Tensorflow (<https://www.tensorflow.org/>, no date). A programming library that will allow me to implement the functionality of a neural network.
- I have also started writing the Milestone One document that will serve to illustrate my progress on the project.
- I have carried out a range of research in order to get a better understanding of neural networks, the research carried out this week was an important step in determining what I needed to focus on in the future.
- I also implemented a gantt chart that will allow me to plan and track my progress as the project develops

Plans for the near future:

- The next stages of the project are to look more closely at some important parts of neural network functionality. This includes developing an understanding of activation functions, backpropagation and how this can be implemented using python and the Tensorflow library.
- I will look at the LSEPI rules and develop an understanding of how they apply to my project.

## Meeting 5

Date: 1/11/2017

Time: 13:00

### General Points:

- This meeting focused heavily on the implementation of Milestone 1 in order to prepare for the upcoming deadline
- The supervisor mentioned that the 1st milestone will form the final report but should also be tweaked using any feedback that we have received.
- In order to properly structure our report, it was suggested that a range of subheadings are implemented listing the various research areas that will be covered
- The supervisor also stated that various sections can be added to our report in order to facilitate further research findings that aid the main body. Such as a glossary.
- I learned that it is useful to have an introduction and a summary for each chapter of the report in order to properly structure my findings.
- I asked about the referencing required when using a source that gains insight from another paper for example. The supervisor suggested that if the information is sufficient. The original source should be referenced.

### Progress Made This Week:

- Progress has been made on research into back-propagation and activation functions
- The main focus for this week was to start implementing the millstone 1 document, a heavy focus of this was implementing my findings into the report in order to properly define what has been learnt.

### Plans for the near future:

- Back-propagation networks will be looked at in more detail in order to gain a greater understanding of how they learn.
- Milestone 1 will be further implemented.
- The implementation of the fundamentals of Neural networks will be researched in order to determine how the program will be created. This will be an important step in starting to create the program.

## Meeting 6

Date: 8/11/2017

Time: 13:00

### General Points:

- With the deadline for the 1st milestone soon approaching, the document was again the main focus of the meeting.
- The supervisor stated that, although it may be tweaked, the 1st milestone will form some elements of our report. I decided that this was a practical approach to the document, and as a result proceeded to structure it as if the start of my report.
- In order to properly structure our report, it was suggested that, for each chapter there should be an introductory and ending section to provide a clear layout for the reader of the project.
- The supervisor also suggested that we may want to use referencing tools that may benefit when citing articles, papers and books.
- It was made clear that the Gantt chart should be included within the document in order to outline the progress made and the plan for the future.
- 

### Progress Made This Week:

- Progress has been made on the milestone document including the formulation of many of my findings as background research.
- The back-propagation learning algorithm was researched further in order to better understand how it can be used to train neural networks.
- In order to gain a greater understanding how neural networks can be implemented, a range of research was carried out into the differing elements that can be put into code, such as the sigmoid activation function.

### Plans for the near future:

- The first Milestone document will be the main focus of this week and is important for formulating the background research that has been completed.

## **Meeting 7**

Date: 15/11/2017

Time: 13:00

General Points:

- This meeting was again focused around the development of the first milestone document
- The supervisor stated that, although the oldest literature should be reviewed first, there are many ways that a review of competing products can be structured.
- The supervisor stated that the millstone document should be around 12 – 20 pages.
- The supervisor also stated the LSEPI ethical rules should be discussed within the milestone, including the checklists of the rules as provided.
- The LSEPI rules may cover the larger area of Artificial intelligence, not just those immediately related to my project.

Progress Made This Week:

- The main focus of the week was to, again, continue the development of the milestone. This includes the background research as this was perhaps the most fundamental part of the first milestone in relation to presenting the progress made.

Plans for the near future:

- To iterate on the work carried out within the milestone 1 document and make note of any queries I may have, to be discussed at the next meeting.
- To research the ethical issues that may occur within my project and within the broader scope of artificial intelligence.

## **Meeting 8**

Date: 22/11/2017

Time: 13:00

General Points:

- This meeting was again focused around the development of the first milestone document, that was due 2 days after the meeting
- The supervisor stated that we should reference as we go in order to avoid doing more work at a later stage
- The supervisor made several points about the milestone and what he expects to be included within the document:

- Design decisions that have been implemented so far
- A section that explains the objectives in detail, for me this is detailed within my introduction.
- List the ethical issues within milestone one, for my project, this would include the broader scope of artificial intelligence in general, whilst also taking a more specific view.

#### Progress Made This Week:

- Alongside carrying out further research in to neural networks, such as error mitigation, the main focus was to continue the development of the first milestone document.
- The main sections of milestone 1 were completed, with competing products left to be written up, this would be achieved before the deadline on Friday.

#### Plans for the near future:

- To complete the Competing Products section of milestone 1 and submit the document.

### Meeting 9

Date: 24/01/2018

Time: 13:00

#### General Points:

- The supervisor stated that we should now have moved beyond background research in order to focus on implementation.
- The Milestone 2 document should focus upon implementation. As a result, the supervisor has stated that we should focus on the prototype or initial realisation of the project.
- I detailed the progress I have made.

#### Progress made:

- Significant progress has been made on the second milestone document, with the deadline on 2nd February.
- Background Research has been completed, aside from future adjustments.
- Significant progress has been made on the implementation of a neural network that can be trained with the MNIST dataset.
- The implementation of the network has been detailed within the Milestone 2 document.

#### Plans for the near future:

- To complete the Milestone 2 document.



- To continue the implementation of the neural network model using the Tensorflow (<https://www.tensorflow.org/>, no date) library. The aim of this part of the implementation is to implement the learning process and gain a good accuracy score.

## **Meeting 10**

Date: 7/02/2018

Time: 13:00

General Points:

- The Milestone 2 document has been completed.
- The supervisor will be providing feedback concerning the second milestone document.

Progress made:

- The neural network model has been programmed that uses the MNIST dataset in order to gain an accurate understanding of characters. The next stage of this process is to achieve OCR.
- The Milestone 2 document has been completed with detailed information about the implementation of the learning neural network model that has been implemented.

Plans for the near future:

- To investigate the process of using the learning neural network model in order to achieve OCR when given a character to classify. For example, a range of image processing libraries will be investigated for key components.

## Meeting 11

Date: 21/02/ 2018

Time: 13:00

General Points:

- The supervisor stated that having briefly looked at the Milestone 2 documents, they look to a good standard. With more feedback to come.
- I discussed the progress I have made

Progress made:

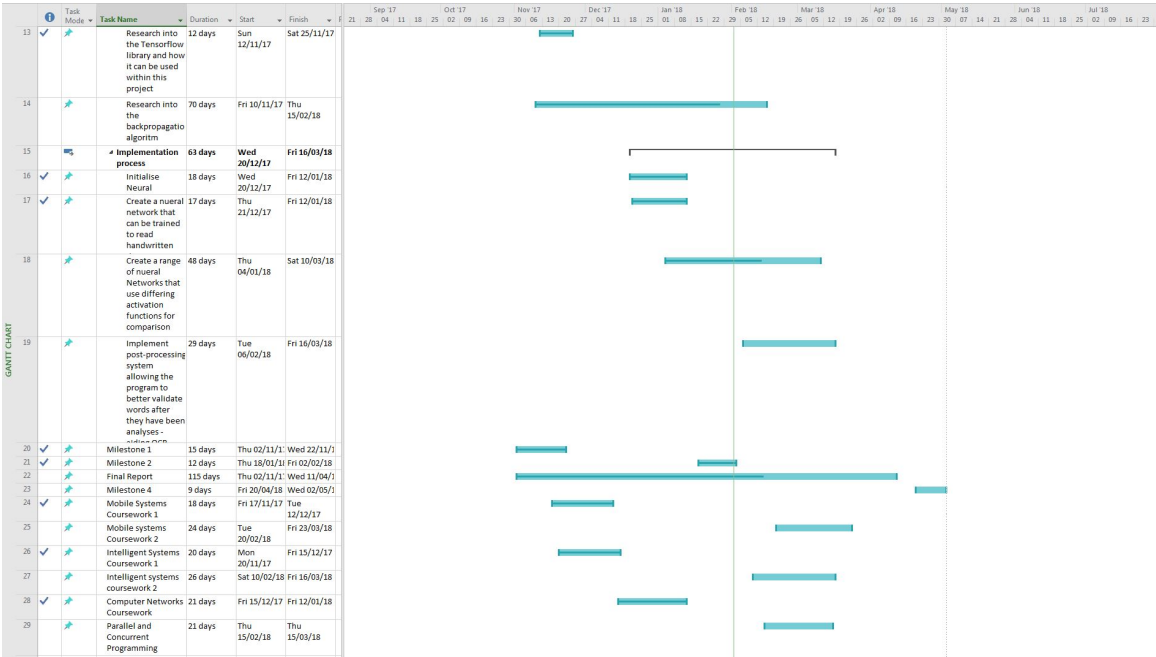
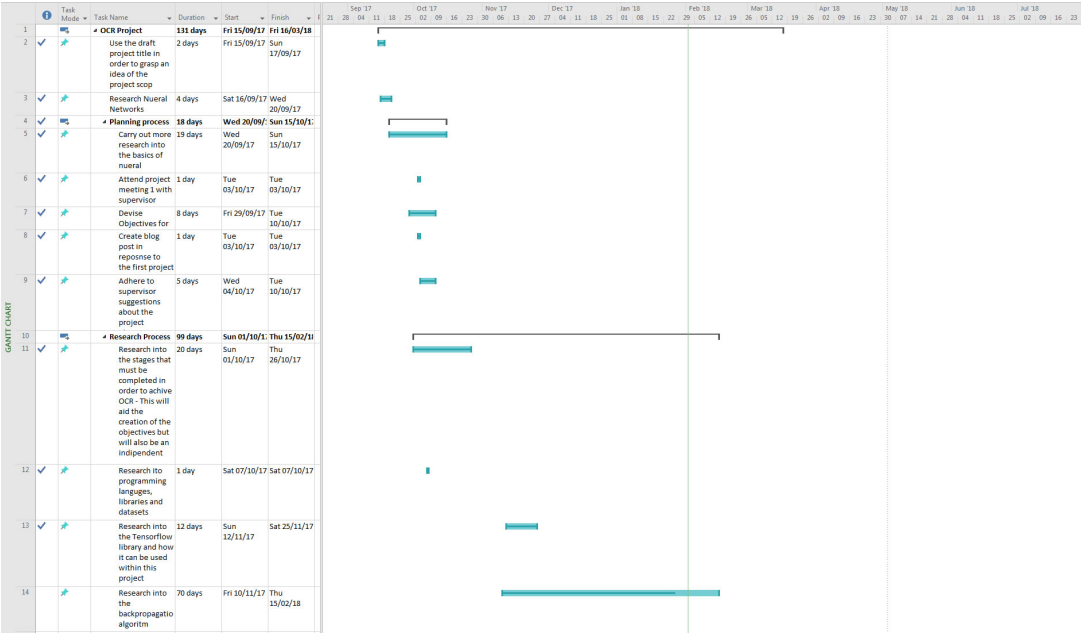
- Research into character pre-processing has been carried out
- fundamentals such as noise reduction and segmentation.
- It has been determined how an instance of a neural network can be saved for use and then implemented in order to achieve OCR.

Plans for the near future:

- To implement my findings concerning pre-processing and achieve the segmentation of words and noise reduction of single characters.

Conclusion of Project Meeting Records

# Gantt Chart



## Ethics Checklist


This form is only applicable for assessed exercises that use other people ('participants') for the collection of information, typically in getting comments about a system or a system design, or getting information about how a system could be used, or evaluating a working system.

If your proposed activity does not comply with any one or more of the points below then please contact your project supervisor and/or project coordinator for advice. If your evaluation does comply with all the points below, please sign this form and submit it with your assessed work.

1. Participants were not exposed to any risks greater than those encountered in their normal working life. *Investigators have a responsibility to protect participants from physical and mental harm during the investigation. The risk of harm must be no greater than in ordinary life. Areas of potential risk that require ethical approval include, but are not limited to, investigations that occur outside usual laboratory areas, or that require participant mobility (e.g. walking, running, use of public transport), unusual or repetitive activity or movement, that use sensory deprivation (e.g. ear plugs or blindfolds), bright or flashing lights, loud or disorienting noises, smell, taste, vibration, or force feedback.*
2. The experimental materials were paper-based, or comprised software running on standard hardware. *Participants should not be exposed to any risks associated with the use of non-standard equipment: anything other than pen-and-paper, standard PCs, mobile phones and PDAs.*
3. All participants explicitly stated that they agreed to take part, and that their data could be used in the project. If the results of the evaluation are likely to be used beyond the term of the project (for example, the software is to be deployed, or the data is to be published), then signed consent is necessary. A separate consent form should be signed by each participant. Otherwise, verbal consent is sufficient, and should be explicitly requested in the introductory script.
4. No incentives were offered to the participants. The payment of participants must not be used to induce them to risk harm beyond that which they risk without payment in their normal lifestyle.
5. No information about the evaluation or materials was intentionally withheld from the participants. Withholding information or misleading participants is unacceptable if participants are likely to object or show unease when debriefed.
6. No participant was under the age of 16. Parental consent is required for participants under the age of 16.
7. No participant has an impairment that may limit their understanding or communication. Additional consent is required for participants with impairments.
8. Neither I nor my supervisor is in a position of authority or influence over any of the participants. A position of authority or influence over any participant must not be allowed to pressurise participants to take part in, or remain in, any experiment.
9. All participants were informed that they could withdraw at any time. All participants have the right to withdraw at any time during the investigation. They should be told this in the introductory script.
10. All participants have been informed of my contact details. All participants must be able to contact the investigator after the investigation. They should be given the details of both student and module co-ordinator or supervisor as part of the debriefing.
11. The evaluation was discussed with all the participants at the end of the session, and all participants had the opportunity to ask questions. The student must provide the participants with sufficient information in the debriefing to enable them to understand the nature of the investigation.
12. All the data collected from the participants is stored in an anonymous form. All participant data (hard-copy and soft-copy) should be stored securely, and in anonymous form.

Student Name: Daniel Luges

Student ID: 15006972

Student's Signature: 

Date: 24/11/17