**Classification of Brain Haemorrhages in Head CT Scans**

**Kirsty Sant**

**Supervisor:** Prof. Ing. Carl James Debono

**Co-Supervisors:** Dr. Paul Bezzina, Dr. Francis Zarb



Faculty of Information and Communication Technology

Department of Communications and Computer Engineering

University of Malta

May 2018

*Submitted in partial fulfilment of the requirements for the degree of B.Sc. (Hons.) Computer Engineering*

# Declaration

# Abstract

# Acknowledgements

Table of Contents

[Declaration 2](#_Toc513120648)

[Abstract 3](#_Toc513120649)

[Acknowledgements 4](#_Toc513120650)

[List of Abbreviations 6](#_Toc513120651)

[List of Figures 7](#_Toc513120652)

[1 Introduction 8](#_Toc513120653)

[2 Background and Literature Review 9](#_Toc513120654)

[2.1 Medical Aspect 9](#_Toc513120655)

[2.1.1 Clinical definition and types of brain haemorrhage 9](#_Toc513120656)

[2.1.2 Computed tomography 10](#_Toc513120657)

[2.2 Computer aided diagnosis – brief overview and developments 13](#_Toc513120658)

[2.3 Pre-Processing, haemorrhage detection and segmentation 14](#_Toc513120659)

[1.1. Pre-Processing 15](#_Toc513120660)

[2.3.1 Segmentation 15](#_Toc513120661)

[2.3.2 Haemorrhage Detection 16](#_Toc513120662)

[2.4 Technological Aspect 17](#_Toc513120663)

[2.4.1 Machine Learning 17](#_Toc513120664)

[2.4.2 Classification 18](#_Toc513120665)

[2.4.3 Classifier Types 19](#_Toc513120666)

[2.4.4 Previously developed systems 23](#_Toc513120667)

[2.5 Conclusion 25](#_Toc513120668)

[3 Design and Implementation 26](#_Toc513120669)

[3.1 Overview of the Proposed System 26](#_Toc513120670)

[3.2 Haemorrhage Detection System 27](#_Toc513120671)

[3.3 Feature Extraction 29](#_Toc513120672)

[3.3.1 Grayscale Image Features 30](#_Toc513120673)

[3.3.2 Shape Image Features 31](#_Toc513120674)

[3.4 Neural Network Design 32](#_Toc513120675)

[3.5 Neural Network Training Phase 34](#_Toc513120676)

[3.5.1 Feedforward step 35](#_Toc513120677)

[3.5.2 Difference calculation 35](#_Toc513120678)

[3.5.3 Backpropagation step 36](#_Toc513120679)

[3.6 Neural Network Testing Phase 37](#_Toc513120680)

[4 Testing and Evaluation 38](#_Toc513120681)

[5 Conclusion and Further Work 39](#_Toc513120682)

[Bibliography 40](#_Toc513120683)

# List of Abbreviations

# List of Figures

# Introduction

# Background and Literature Review

## Medical Aspect

### Clinical definition and types of brain haemorrhage

Brain haemorrhage is defined as a bleed in or around the brain tissue. It may be spontaneous, precipitated by an underlying vascular malformation, induced by trauma or related to therapeutic anticoagulation [1]. This can be caused by a blood vessel rupturing, typically due to high blood pressure, stroke or trauma to the head. The resulting bleeding, especially within the brain tissue, results in an increase in pressure on the part of the brain near the bleed, which pressure can potentially damage the effected brain tissue. The place of the ruptured blood vessel, the speed at which blood flows into the brain and the volume of the bleed can all be factors of the severity of the case, potentially leading to death. The region of the brain in which the bleed occurs tends to indicate the functions and bodily abilities the patient might lose, such as movement of one side of the body or speech. The increase in pressure in the region where the bleed occurs is due to blood irritating brain tissue thus making it swell. Although high blood pressure is often one of the main causes of brain haemorrhage, haemorrhagic stroke only amounts to roughly 20% of all stroke cases.

In the cases of head trauma, traumatic brain injury (TBI) is a possible occurrence. In the case of traumatic brain injury, the bleed in the brain is a result of an external force, such as a blow to the head in instances like motor vehicle crashes. In such cases, the brain is highly likely to move within the skull, potentially shearing the dura matter, of the innermost layer of the skull, and causing contusions, or bleeding and bruising in the brain. This thus implies that in the cases of trauma, a haemorrhage may be present even if there is no visible skull fracture. Given the typical causes of such pathology, it is most common in young adults. TBI is considered to be the highest cause of death in the people aged 15-24, and the third highest, after heart disease and cancer, in other ages [2]. For patients diagnosed with brain haemorrhage, a late or wrong diagnosis can potentially lead to disabilities or death, making the quick and correct diagnosis imperative for such cases.

There are five different types of haemorrhage. These are intracerebral haemorrhage (ICH), intraventricular haemorrhage (IVH), epidural haemorrhage (EDH), subdural haemorrhage (SDH) and subarachnoid haemorrhage (SAH) [3]. The location of the bleed determines what the original source was, for instance subdural and epidural haemorrhages are usually due to experienced trauma, sometimes having a scar related to them and are located towards the side of the brain, not inside [1]. Each of these types of haemorrhage has different qualities that are used to distinguish one from the other. At a basic level, a bleed can be classified as intra-axial or extra-axial. These two classifications are the broader categories, with each having more sub-categories. Intra-axial bleeds are ones which occur within the brain itself and include ICH and IVH whereas extra-axial bleeds are those occurring within the skull but external to the brain tissue and tend to be easier to treat. Extra-axial bleeds can be further classified into 3 sub-categories, namely EDH, SDH and SAH.

<Possibly enter image of haemorrhages?>

### Computed tomography

Computed tomography, or CT, relies on the principle of using x-rays transmitted through the body to analyse interior parts of the body in a non-invasive way, meaning without the need to operate. The underlying principle behind CT and how it works is that x-rays going through the body are absorbed to some extent, with the rate of absorption varying according to the medium they pass through. The rate of absorption is calculated by measuring the attenuation coefficient of the detected x-ray wave in comparison to the fixed emitted x-ray waves, which properties are known. The attenuation coefficient quantifies how much the detected x-ray signal is weakened after passing through a given material, and is used to calculate the density of the material [4]. This is the same principle behind conventional radiography, which is traditionally used to examine the skeletal structure for fractures and assess lung pathologies, amongst other applications [5].

The main difference between these two types of scans is that in CT a more sensitive detection system, typically making use of glass or crystal. This increased sensitivity gives the ability of detecting smaller changes in the absorption values, which in turn implies that finer detail, such as the distinction of density gradation within soft tissues can be observed. The detection system is connected to a computer, which is used to visualise these results [6]. Another important distinction between conventional radiography and CT is that while in the former a fixed x-ray tube is used, CT uses a motorised x-ray source that rotates around the circular opening through which the patient is passed, referred to as the gantry [7]. The advantage of this method over conventional radiography is that the images produced are cross-sectional, which eliminate the issue of the superimposition of structures obtained in an image produced by conventional radiography.

#### How CT works in general

When a CT scan is to be performed, the patient is placed such that the part to be examined is within the gantry, within which the x-ray emitter tube and detector are enclosed, opposite one another [8]. As the patient is passed through the gantry, the emitter and detector are quickly rotated, with the emitter producing a narrow x-ray beam. For every rotation, the data collected forms a cross-sectional image, known as a section or ‘slice’. The typical thickness of each section is between 1 and 10mm. Given that multiple rotations are performed, a number of adjacent sections are obtained, which can then be digitally stacked to form a 3D image of the patient. The 3D image can in turn be used to identify the location of basic structures and any abnormalities in an easier way [7]. The radiographer who plans and performs the CT examination decides the thickness of each section, considering the trade-off between accuracy and number of sections, according to the case at hand.

There are two main ways of scanning, which are the slice-by-slice, conventional CT scanning and the volume acquisition scanning, which is also referred to as ‘spiral’ or ‘helical’ [6]. In conventional slice-by-slice scanning, the patient is in a fixed position while a section is being scanned. In this way, the thickness of each section can be defined by how much the patient is moved between one complete scan and the next, and the position is defined by the position from which scanning starts [9]. The main drawbacks of this form of scanning are that it takes a longer time to scan the body section of interest, which can prove to be challenging with children and people who cannot hold still for a long time, and that the sections’ position and thickness are fixed while scanning and cannot be changed afterwards. In spiral scanning, the patient is constantly moved through the scanner, while the x-ray emitter and detector move continuously in one direction such that it traces a spiral path, collecting data continuously. In this way, a shorter time is taken to scan the section in question, thus eliminating inconsistencies due to breathing or slight movements. The data collected is stored as a volume, thus any required position within the body section can be obtained from the data set, either because it was scanned at that position or via reconstruction. Furthermore, spiral scanning facilitates the reconstruction in 3D form and the possibility of reconstructing the image in a different plane [6] [9].

#### Brain CT Scans

For the vast majorities of neurological disorders, CT or Magnetic Resonance Imaging (MRI) are used, since both give more information when compared to conventional radiography [6]. In CT scanning, scans are performed along the axial plane. Should other planes be required, in some cases they can be reconstructed from the axial plane result set.

When analysing a CT scan for abnormalities, there are three main radiological signs that one should look for. These are abnormal tissue density, mass effect and enlargement of ventricles. Abnormal tissue density refers to areas in the brain tissue that have higher or lower densities, thus are seen as lighter (hyperintense) or darker (hypointense), when compared to the rest of the brain. The mass effect is the displacement of the brain’s soft tissues due to an intracranial lesion [10] that was not always present and is taking up space, such as a bleed or a tumour. This can be seen in the scan as a compression or displacement in the lateral ventricles and shift in midline structures of the brain [6]. Finally, an enlargement of ventricles can be due to an increase in the volume of cerebrospinal fluid (CSF) in the brain. This can be easily noted in the scan when comparing the size of regular and abnormal lateral ventricles.

In CT scans, a fresh bleed is characterised by an area of hyper-intensity, and possibly surrounded by a hypo-dense area, known as oedema, which is caused by swelling [6]. The importance of correct diagnosis of the case based on the scan is vital in determining whether the patient requires immediate surgery or not, since in some cases operating the patient could potentially be harmful. CT scans, albeit being the best initial way of detecting these bleeds, can only visualise the largest aneurysms, which are usually the source of such bleeds.

ICH in CT scans can be easily seen as a hyperdense area, or a white patch, within brain tissue, and thus gives little difficulty in diagnosis provided the bleed is large enough to detect. IVH in CT has the similar hyperdense property but the location of the bleed is within one or both brain ventricles. EDH is usually lens-shaped, distinct, hyperdense area and usually associated with a skull fracture. Its shape is heterogeneous, meaning it is not uniform. SDH classically appears as hyperdense crescent-shaped area and is situated over the surface of a cerebral hemisphere, with a skull fracture potentially present but not necessarily so. SAH is the most difficult to detect, since generally the bleeds are small and dispersed [11].

## Computer aided diagnosis – brief overview and developments

Computer aided diagnosis (CAD) is a research area which bridges the gap between technology, more specifically Artificial Intelligence (AI) and the medical world. CAD can be currently considered as a major research area in medical imaging, technology and radiology. The background idea behind CAD is for the computer to analyse a medical case in a particular field and offer a “secondary opinion” to the radiologist, who is the person taking the final decision on the case. This thus implies that the computer, using a suite of tools in multiple areas, such as image processing and classification, can reach a conclusion on whether or not there is the pathology being tested for. With the use of such software, there can be cases where the radiologist misses to spot a problem, yet the computer detected a region of concern, thus triggering the attention of the radiologist and possibly improving the diagnosis. Considering the way CAD works, one can note that the concept of the system is to put equal weight on the computer result and the role of physicians and radiologists, without having one undermining the other, to reach a conclusion on whether the pathology being tested for is present or not. This alleviates some pressure from the performance of the system itself, in the sense that it is not necessary for the system to have a performance which is superior to that of the radiologist, but rather it being comparable and complimentary [12].

Although CAD has been popularised as a research area only recently, it has been around, in different forms, for quite some time. The history of computers as tools for diagnosis dates to the 1960s, where the idea of automated computer diagnosis started to emerge. The notion behind this movement was the general assumptions that computers can completely replace radiologists and physicians in identifying and diagnosing certain pathologies, given that the computer was proven to be superior to humans in other areas. This did not work out due to several factors, including the lack of sufficient computing power at the time, lack of advanced image-processing techniques and the lack of access to digital medical images. Overall, at the time, too much was being expected from computers, which led to the notion losing popularity and being deemed as unfeasible and impossible.

In the 1980s however, a different approach to automated computer diagnosis was introduced, where the system output can be used by the radiologist to help in the decision-making process, but not replace them. This system, dubbed as Computer-Aided Diagnosis was widely accepted as a concept and research interest in the area grew widely and rapidly. This was due to multiple factors, including the reduced performance expectations since the computer program’s output is combined with, rather than replacing, the radiologist’s expertise. This is not the case for automated computer diagnosis, where computer performance is of utmost importance given that the result is being issued solely by the computer. Performance in such cases is measured by specificity and sensitivity. For automated computer diagnosis, both of these factors are required to be very high, that is comparable or higher than that of radiologists, but given that the performance level of radiologists and physicians is much higher than that of a computer, lower specificity and sensitivity values are allowed in a CAD system. [12]

CAD is currently being researched extensively in the medical imaging realm. It has proven to be a very useful tool in the industry, particularly when it comes to the detection of breast [13] [14] and lung [15] [16] cancers. These areas were of great interest since lots of screening tests are performed to check for these pathologies. Given that most screened cases are normal, it may be tedious and time-consuming for the radiologist to go through each result, thus with CAD, these results can be fed through the system, and the computer flags which results have areas of concern. Such systems are nowadays available for clinical use.

## Pre-Processing, haemorrhage detection and segmentation

Once the CT scan images are obtained, the following step would be processing these images such that they can be checked for bleeds. With such techniques, one can determine if there is a region within the section which is suspected to contain haemorrhage, in this case, or any other pathology being tested for. The three main processes to detect whether there is a haemorrhage present in a CT image are pre-processing, segmentation and detection. For the purpose of this project, the procedures mentioned below have already been covered by the system that was developed by a graduate student, Mr. Napier, in his final year project [17].

### Pre-Processing

The pre-processing part involves mainly noise removal techniques. Medical imaging is known to produce images susceptible to visual noise, which can potentially make it harder to identify the bleed from other parts of the brain, specifically the brain’s white matter (WM) and gray matter (GM). Following the analysis carried out by Mr. Napier, the best noise removal technique was found to be a bilateral filter with a 5-pixel neighbourhood, colour standard deviation value of 10 and a space standard deviation of 2.5.

### Segmentation

Segmentation is the process of extracting information from a region within an image while removing other parts of the same image that are not required. This step was carried out twice in the course of determining whether a bleed is present or not. The first segmentation step was carried out to extract the brain tissue information and discarding the part of the image representing the skull, and the second segmentation was carried out to extract the bleed from the rest of the brain tissue.

The segmentation process is initiated with the removal of the head tissue from the skull. This was accomplished by making use of thresholding and contour methods. Once this step is complete, morphological operations, namely dilation and erosion were applied on the segmented image. Incorporating these two techniques has proven to help the system achieve better results. Dilation involves convoluting an image with a kernel containing a defined anchor point, which is usually situated at the centre of the kernel. This results in regions within the image containing a particular colour value expanding, based on the description of the kernel. This process was required so as to join disconnected bone structures within the brain. Conversely, erosion is a similar process with the opposite effect obtained. With erosion, the anchor point in the kernel is replaced such that bright areas in the image are shrunk, once again based on how the kernel is implemented.

Following the segmentation process, the result was presented as an image of the brain tissue on a black background. This was achieved by detecting the largest contour in the original image and using it as a mask. In this way, any pixels within the mask are retained as in the original image, whereas all other pixels outside the mask were set to black.

### Haemorrhage Detection

The main techniques used for the detection of the haemorrhage are clustering, thresholding and contour detection. In this step, the segmented brain image is considered, and the pixel intensities present in the image were grouped into four clusters, representing CSF, WM and GM, brain parenchyma and haemorrhage pixels respectively. Once this clustering is performed, the lower and upper pixel intensity thresholds were determined. The lower intensity threshold was set to be a value found half way through the third cluster and the upper intensity threshold was determined, through testing, to be 40 intensity levels above the lower threshold. By making use of a contour-finding algorithm, all the contours of joined pixel masses were obtained, and should the pathology be present, it is identifiable as the largest detected pixel mass.

In order to minimise the number of false positive detections, a number of features were implemented, which can be categorised in three different tests. The first test states that for a pixel mass to be considered a haemorrhage, the area of the mass must be greater than 3788 pixels, and at least 10,000 pixels smaller than the area of the original brain. The latter check was performed to eliminate the possibility of GM and WM being mistaken for haemorrhage should the skull be incorrectly segmented. Furthermore, the perimeter of the closed contour assumed to be the bleed itself must be 1.5 times smaller than the contour of the segmented brain image.

For the cases when a contour perimeter is less than 2000 pixels, the area enclosed by this contour is required to be between 2800 and 15,000 pixels and the contour perimeter of the segmented brain image is at least 4000 pixels longer than that of the haemorrhage contour.

The final test carried out takes into consideration the volume of the bleed. Should a bleed be present in the brain, it will appear in multiple consecutive slices. When the case is being processed, if a pixel mass obeys one of the two described tests, a counter is incremented. Should there be 4 or more consecutive sections containing a pixel mass corresponding to a bleed, the pathology is confirmed whereas if the consecutive slice count is less than 4, the case is considered free of haemorrhage.

## Technological Aspect

### Machine Learning

Machine learning is defined as an “application of Artificial Intelligence (AI) that provides systems with the ability to automatically learn and improve from experience without being explicitly programmed” [18]. In this area of computing, the main aim is to design software that uses a given dataset to learn and adapt itself without being hardcoded to do so. A machine learning algorithm is thus designed with the aim of finding patterns in the data given and alter the internal workings to make better decisions based on the training set provided. Machine learning algorithms can be divided into three main categories, which are supervised learning algorithms, unsupervised learning algorithms and reinforcement algorithms.

In supervised machine learning algorithms, the training dataset is clearly labelled, in the sense that for a given set of input parameters, the expected output is known. When the training data is fed to the algorithm, the learning algorithm develops a function that produces predictions of the output values. These predictions are then compared to the expected outputs to find errors and correct the developed model accordingly. This model is more widely used, and the majority of practical machine learning techniques use supervised learning.

In unsupervised machine learning algorithms, on the other hand, the training dataset is neither classified nor labelled. When this training dataset is fed to the algorithm, there is no right or wrong answer that is produced. This means that for the input training set, there is no expected output defined. The algorithm is designed to find similarities in the provided dataset and derive a function that defines the hidden structure within the unlabelled data, thus this technique is mainly used to learn more about the data being fed to the system.

In real applications however, the vast majority of machine learning systems incorporate a combination of these two techniques. In these applications, which are referred to as semi-supervised learning, there is a large volume of input data, with only a small subset being labelled. These applications were built given that the process of labelling all the input data is highly time-consuming and can potentially be expensive if the data labelling process requires the help of experts in the domain, while on the other hand, unlabelled data is cheap and easy both to collect and store. In some applications, and for the problem being tackled, a small dataset of labelled data can be used to train the application such that when feeding unlabelled data, a better prediction can be made based on the modifications to the model made via the labelled training data. The newly labelled data can then be fed back to the supervised learning system to further adjust the model for new unlabelled data [19].

The third type of machine learning algorithms is reinforcement learning. In this approach, machines and software agents determine the ideal behaviour based on the context that the application is being used for by interacting with the surrounding environment, producing an action and discovering if the said action causes a reward or error via the reinforcement signal. In these cases, machines and software agents learn, through trial and error, which is the optimal decision based on the context to provide maximum performance [18].

### Classification

Classification is the task of labelling data from a set into one of many subsets or classes. This technique can be applied to data in different formats, such as images, audio and text. While classification is considered to be a relatively easy task for the human being, it has proved to be quite a challenging task for machines, given the complexity of the problem in itself. However, with the increase of computational power, classifiers have gained more popularity and performance power, which is leading to them being used in an ever-growing number of applications.

Image classification, in particular, refers to the task of extracting information from a raster image, and categorising all pixels in an image into one of multiple classes. The whole concept of image classification is mapping regions of an image to particular predefined classes, and with sufficient training this can be achieved quite accurately. Given that image classification is a subset of machine learning, all forms of learning algorithms can be applied to the problem. Some of the most common image classification techniques include support vector machines (SVMs) and Neural Networks (NNs).

In most applications, supervised classification algorithms are applied, either for the entire span of the network operation or as the training potion of the classification algorithm. These algorithms include Nearest Neighbour Algorithms, SVMs and NNs. In supervised classification, each case within a class tends to have common characteristics, such as the mean, covariance matrix and the minimum and maximum grey levels in the applicable class, which values are obtained by calculations during the image processing stage.

There are various classifier types that can be used for this application. The main types of classifiers can be divided into different categories such as linear classifiers, such as Naive Bayes and statistical such as logistic regression, decision trees, SVMs and NNs. The choice of which classifier to use depends on multiple factors, such as the number of cases available to train the classifier, previous knowledge of class probabilities and exactness of such knowledge, and computation complexity requirements.

Apart from choosing the classification algorithm, another decision needs to be made with regards to the evaluation methods. These methods give an estimate to the performance of the classifier by measuring the error rate. These methods describe how to use the currently available dataset to train and test the classifier. Methods such as redistribution uses full dataset to first train the classifier and then reuse the same dataset to test it, which is easy to implement yet gives a highly optimistic error rate. Other methods, which are more realistic, include partitioning the dataset into two groups for training and testing, which can be done in various ways, such as splitting the dataset once, with the training set being significantly larger than the testing set as done in data partitioning [20].

### Classifier Types

As already mentioned in the previous section, there are various types of classification algorithms that can be employed based on a number of factors. The three most commonly used classification algorithms are KNNs, SVMs and NNs, as indicated above.

#### K-Nearest-Neighbours (KNN) Classifiers

The KNN algorithm is an easy, robust and versatile way of implementing classification, and is typically used as a form of comparison metric when considering other more complicated classification algorithms. Despite the algorithm being relatively simple and straight-forward to understand, it has proven to beat more complex classifiers in a number of tasks. This algorithm is considered as a supervised learning algorithm, as well as a non-parametric and instance based learning algorithm. This means that no explicit assumptions are made by the algorithm about the form of the function that eventually maps the inputs to the outputs, and that the algorithm remembers the training instances which are used as the knowledge when predicting a test case outcome, rather than an entire model. While KNNs are great as an initial form of testing on data that is to be fed into more complex systems, it is a very computationally intensive algorithm, which leads to a very long testing phase.

The algorithm makes a prediction on a test case by seeing under which class the nearest *k* neighbours fall, and the test case is classified in one of the classes based on the classification in which the majority of the nearest neighbours are classified in. Programmatically, this is done by computing the difference between the test case and each of the training cases, selecting the *k* training instances that give the minimum difference. Considering the subset of *k* training instances, the test case is assigned to the class with the highest conditional probability, i.e. to the class in which the majority of these *k* neighbours is found.

As it can be seen, the value of *k* is one of the most important parameters in this classifier setting. In this setup, *k* can be any positive integer. A small value of k provides a fit with low bias but high variance, whereas a large value of *k* help in making the algorithm resilient to outlying points, which translates to a lower variance but increased bias. The choice of *k* depends on application, and a parametric analysis can easily determine the best value for *k* [21].

#### Support Vector Machine (SVM) Classifiers

The SVM algorithm is a supervised learning algorithm which can be applied to both classification and regression. Each data point in the training set is plotted in an n-dimensional space, with the number of dimensions corresponding to the number of features being taken into consideration. Following some processing, each data point is translated into a coordinate in this space [22]. The algorithm tries to find an optimal decision plane that defines the boundary between any two classes, referred to as a hyperplane. Hyperplane selection is based on two main principles: best possible class segregation and maximising the margin, which is how far the nearest data points are from the hyperplane. In order to find the ideal hyperplane for a dataset, one might require the use of a kernel, which is a transformation function that converts a low-dimension space into a more complex, higher dimension space in which the dataset can be linearly separable.

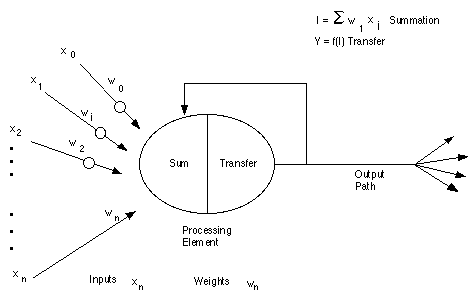
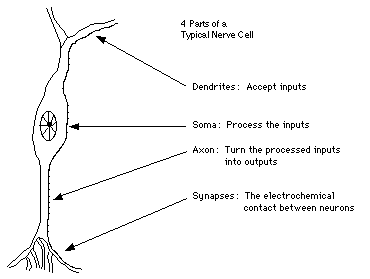
This algorithm has proven to be useful in multiple cases, particularly those where a clear distinction between the classification classes exist, in highly dimensional spaces and in classification cases where there exist more dimensions than samples. However, the algorithm performs poorly when the dataset is large due to the increased training time and in cases with overlapping classification classes. Furthermore, the algorithm does not inherently provide probabilistic measures of correctness, which would have to be computed using other, computationally expensive, functions [23].

#### Artificial Neural Networks (ANNs)

An ANN is a “parallel computational network made up of interconnected neurons” [24]. An ANN is formed of multiple nodes, referred to as neurons, which are connected to other neurons in previous and following layers, and were designed as a computational version of how the human brain works. The structure of a neural network is analogous to the structure of the brain, in the sense that each neuron is connected to several other neurons [25].

In a biological neuron, the dendrites are hair-like extensions of the soma which collect inputs from other neurons connected to it, the soma processes these inputs. Following the input processing by the soma, the axon converts these processed inputs into an output, which is then sent out of the neuron via the synapses, which are the structures connecting one neuron to another, effectively passing the output of the neuron in question as an input to a set of other neurons connected to it. In a similar way, an artificial neuron is designed to incorporate these four different functions performed by the biological neurons. The inputs to the neuron are multiplied to some weight value *w*, which vary for each input. There weighted inputs are then summed together, and a transfer or activation function is applied to them to produce an output. This output is then fed to neurons in the successive layer or as an output to the network should the neuron be in the output layer [26]. This similarity between a biological and an artificial neuron can be seen in Figure 2.1.

Figure 2.1: Analogy between a biological neuron and an artificial neuron [26]



In an ANN, neurons are divided in a number of layers, namely the input layer, output layer and one or more hidden layers. In any one of these layers, each neuron is connected to every other neuron in the successive layer, if any, and receives inputs from every neuron in the previous layer, making the structure fully-connected. The input layer for the neural network represents the number of dimensions in the data, whereas the output layer consists of the number of classes in which an input can be assigned to. The number of dimensions in the input are fed into the network one by one as a normalised weighted value to each of the input layer neurons. Each of the input neurons performs the weighted sum which passes through the activation function and produces an output, which is propagated into the next layer. The output propagates forward in the network in the same way until reaching the output layer, where each neuron outputs a normalised value corresponding to the probability of the given input being classified into that class.

ANNs can be designed to follow supervised learning and unsupervised learning, based solely on the data being fed and how the weights are adjusted to reach a local minimum, however the most common structure for an ANN used as a classifier is using supervised learning to train the network and backpropagation to adjust the weights after every training iteration.

One of the major advantages of neural networks over other systems is their ability to infer results based on input-output relationships and other underlying rules in the data. Even though the network requires a significant amount of training thus requiring large processing times especially for larger structures, the structure allows for information to be processed in a highly parallel way, thus speeding this process. Given that the network is designed to learn patterns in the data, the same structure can be reused for other purposes without having to alter the structure of the network within itself.

### Previously developed systems

The task of developing an automated classification system for brain haemorrhage in CT and MR images is no new topic in CAD systems. There have been many attempts in the development of a reliable, robust system that can cater for such a need. While for MR images this was quite successful, the research in CT images still has room for improvements. Considering what was done previously in the area, one can see that there have been several attempts to address the research area, using different algorithms. The two most common classifiers used in Brain CT scan classification are the K-Nearest Neighbour Classifier and the Artificial Neural Network. Shahangian *et.al.* [27] compared the accuracy of KNN and ANN algorithms for classification and found that using a multilayer perceptron (MLP) neural network with 14 input nodes representing the 14 features extracted from the CT image, 12 hidden-layer neurons and 3 output neurons gave much better results when compared to using the KNN algorithm. An improvement on the obtained result was observed by applying a genetic algorithm (GA) to the input features to select the best features to be used for classification. This yielded an improvement for both the KNN and ANN classifiers. When testing the classifiers, the image set consisted of EDH, SDH and ICH bleeds mixed with normal images. 50% if the image set was used as the training set for the KNN classifier and the other 50% was used for testing, whereas the dataset distribution for the ANN structure was 70% for training and 30% for testing the system.

In another study, Sharma *et.al.* [3] created an ANN structure with 16 features fed as inputs, 30 neurons in the hidden layer and 3 outputs, classifying EDH, SDH and ICH. Their study used 100 brain CT images, with haemorrhage present in all images. 70% of the dataset was used to train the network and the remaining 30% were used to test the system.

In a third study carried out by Balasooriya *et.al* [28], an ANN structure was designed and tested. The number of input features were 3 – the number of closed-contour objects detected; 1 if SDH since the skull and brain are ‘attached’ or 2 if ICH since it is separate from the skull; the area of the first closed contour typically the brain itself, and the area of the second closed contour if applicable. For this study, the number of hidden-layer neurons was varied and it was determined by the author that 15 neurons was the best option. The number of samples available for this implementation were 50, where 80% were used for training and 20% for testing. In the table below, one can see the observed accuracy levels for each of the cases mentioned above.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Author | Classifier type | Hidden layer Neurons | Total cases | Training set | Testing set | Accuracy |
| Shahangian [27] | KNN without GA | N/A | No information | 50% | 50% | 53.33% |
| Shahangian [27] | KNN with GA | N/A | No information | 50% | 50% | 60% |
| Shahangian [27] | ANN without GA | 12 | No information | 70% | 30% | 86.7% |
| Shahangian [27] | ANN with GA | 12 | No information | 70% | 30% | 93.3% |
| Sharma [3] | ANN | 30 | 100 | 70% | 30% | 97% |
| Balasooriya [28] | ANN | 15 | 50 | 80% | 20% | 97.8% |

Table 2.1: Findings from previous work

When considering all the above studies, which were the ones most relative and comparable to the study being carried out, it was noted that there seemed to be a similarity in the haemorrhage cases being analysed. As noted above, the previous research papers limited their systems to cover only ICH, SDH and in some cases EDH. This is mainly because SAH is very difficult to detect due to it being highly dispersed throughout the brain and is spotted in CT scans as numerous small patches of blood in the subarachnoid space.

## Conclusion

From the analysis of previous work carried out in relation to this problem, one can see that while there has been some research carried out, there is still room for improvement and further research. It can be clearly seen from the comparison performed in the previous section that the best results thus far have been achieved using a single hidden layer ANN using several features from the CT image as inputs to the system. Considering this, one can argue that there is still room for experimentation with the type of classifier being used, with the intention of seeing if the accuracy can be increased further.

The first conclusion that can be reached following the analysis of previous work is that the types of haemorrhage being classified is not the full range of possible diagnoses. Out of the five different haemorrhage types, only two were considered for classification in these studies. This can be due to different reasons, one of which is the potential lack of cases of the types of haemorrhage not classified. Another reason behind the lack of classification of different types could be that the developed system was not intelligent enough to recognise the haemorrhage type since not enough information was being extracted about the bleed. As Balasooriya *et.al* [28] mentioned in their section on further improvements, the three parameters being fed into their network were not enough to help distinguish between EDH and SDH since they both give the same number of objects and roughly the same area. For the detection of SAH, their suggestion is to take into consideration the histogram value of all objects, since such a pathology can be seen spread throughout the image.

Another conclusion that can be reached is that the ANN structure is the most successful in classifying correctly these haemorrhage types. Considering the study that targeted the widest variety of haemorrhage detection, Sharma *et.al.* [3] classified between EDH, SDH and ICH with 97% accuracy, using 16 features as inputs to the designed ANN classifier. This shows that the current research is highly accurate, but there is still room for improvement.

# Design and Implementation

This chapter contains an in-depth explanation of each of the components and modules forming the proposed CAD classification system. The haemorrhage detection system designed by Mr. Napier [17] will be treated as a standalone, autonomous system, given that how that system was designed is not within the scope of this project, however, this chapter will contain some explanation of the slight modifications made to this algorithm such that the required images were extracted, and more bleeds were detected. The components relating to feature extraction from images, the neural network structures and the training and testing phases of the neural network classifiers built will be explained in more detail. The system will then be presented as a final program, with an overview of the required inputs, how the proposed system communicates with the detection system and produced outputs.

## Overview of the Proposed System

Figure .: Flow Diagram of the implemented CAD Classificaiton System

Read Input Images of Haemorrhage Case

Haemorrhage Detection System

Feature Extraction

Training of Neural Network

Testing of Neural Network

Output Classification of Input Haemorrhage Case

As it can be noted from Figure 3.1, the proposed system will be formed of four major steps. These are haemorrhage detection, feature extraction, neural network training and neural network testing. The implemented system, excluding the detection system, was built using Python in conjunction with OpenCV and SciKit libraries in Ubuntu. It is important to note that since the implemented system was designed to build upon the output of the detection system, all the restrictions of the detection system were inherited to this system. The functionality of each of the steps depicted in the flow chart will be further explained in detail in the coming sections within this chapter.

## Haemorrhage Detection System

As indicated earlier on, the study being carried out is based on the work carried out by Mr. Napier in his final year project. This work involved performing image processing techniques on individual brain CT slices and determine if a bleed is present or not. This system was built using C++ and OpenCV libraries, and as an output, each CT slice was displayed on the screen, with a red drawing around the area considered to contain a bleed, if present. The detection system determines if a CT slice contains the pathology based on the area and perimeter of the largest closed contour noted, other than the brain itself, and the number of consecutive slices in which the pathology was deemed present by these same checks was used to determine if these marked closed contour form a volume which translates to an actual bleed in the brain. The latter check was added, along with other checks, to ensure that the number of false positive cases detected is kept at a minimum.

The first step in the development of this full program was to understand what Mr. Napier’s program does, how the images are being processed and how haemorrhage was being considered present or not. Firstly, it was noted that for Mr. Napier’s program to work, the directories from which the input images are read and images from the different stages of the algorithm are stored need to be changed since absolute paths were used to access such directories. This was outlined in the program execution instructions text file included with Mr. Napier’s work, and thus before being able to run the program these paths were changed accordingly, and the folder hierarchy was set up as suggested.

For the program being developed in the study, it was determined that two images were required to extract the features of the bleed, if present in the first place. These were the grayscale image following the segmentation process and the binary image obtained following the binary thresholding step. For this purpose, some additional lines of code were written to save these images in a different directory, which will then be referenced by the newly developed system to perform feature extraction. The location in which these two images were stored varied on what the purpose of the input CT case was. This was done so that images with the intention of being used for training the neural networks were stored in a different folder from those that were used to train the system.

Given that the system built by Mr. Napier was written in C++ and it was chosen that the built system for this study was to be written in Python, the main issue to sort out was how to make the developed system communicate with the system being built. To mitigate this difference, there were two options to choose from, which were developing a wrapper class for the C++ code of the detection system to be then used from the classification program directly or else treating Mr. Napier’s program as a standalone application that can be run directly without the application having any direct communication with the new system. Given that the detection algorithm was already designed to be completely standalone, the only information deemed required from this system are the two images outlined above, and there is no need for direct communication between one system and the other during runtime, it was decided that the latter option is a more suitable solution. Thus, any changes made to the detection program were done just once, the code was compiled and only the executable file was accessed by the new system. Furthermore, it was deemed appropriate to have the final proposed project run as an entire system by using just one command. For this purpose, the *system* function within Python’s OS module [29] was used. This function takes as parameter a string which is then executed directly within a subshell. In this way, the haemorrhage detection program was executed within a subshell by passing the command to execute a C++ executable file as the parameter to this function. It is important to mention at this stage that should this program be executed in an operating system other than Ubuntu or another Debian Linux distribution, this command parameter might need to change since it is operating system dependent.

For the haemorrhage detection program, the original system outputs were retained and are still presented to the user. These outputs include the image of the segmented brain, which would have a dark pink outline around the detected haemorrhagic region and the preliminary classification output displayed through command line output.

## Feature Extraction

The first step performed by the classification system was to extract features from the images obtained from the detection system. Feature extraction is the process of analysing input data that is too large and containing redundancies such that it is transformed into a reduced set of more valuable features. The main goal behind this is obtaining more relevant information from the data using a smaller number of dimensions [30]. The dimensions of the input data will eventually become the number of inputs used by the neural network architecture. Given that the study concerns images, the full input set for one image would be all the pixels within the image being analysed. Apart from the fact that these images can be of different sizes, there are lots of regions within the image that have identical or near-identical pixel values, which can be grouped together to reduce the number of input dimensions of the data, hence reducing the number of input layer neurons needed for the neural network.

Following the analysis of previous work carried out in the area, as seen in Table 2.1 above, all work featuring the use of neural networks makes use of feature extraction to reduce the number of inputs. The two most relevant studies show that the average number of neurons was 15 - Shahangian *et.al.* [27] used 14 whereas Sharma *et. al.* [3] used 16 different features. Most of these features are common to both studies, and it can be seen that they are divided into three main categories: shape features, intensity features and texture features. The combination of all the extracted features would result in a unique input pattern, which can then be used to train the neural network or test the trained neural network for accuracy. Based on what was already built, it was noted that there were some features that were presented in one study and not the other, and thus it was decided to implement all the features seen in these studies. For this reason, the built system extracts a total of 18 features.

The 18 features extracted for every image with a suspected haemorrhagic region in it were divided into two main groups in the program, which were the grayscale features and the shape features. Correlating these feature classes to the ones mentioned above, the grayscale features class incorporates the intensity and texture feature classes. This was done since overall, the intensity and texture features were related to the grayscale image, whereas the shape features were more easily obtainable from the binary threshold image, which in turn explains why two different images were extracted from the detection system with the intention to be used as inputs for the classification system.

As mentioned above, the feature extraction process was divided into two main categories, with each category having its own function of computing all features that can be extracted from the image assigned to it. From the grayscale image a total of ten features were extracted and from the contour image the remaining 8 features were extracted. The following subsections will contain more in-depth information on which features were extracted for each of the two images.

### Grayscale Image Features

The grayscale image feature extraction mainly covers the collection of information that relates to the intensity and texture of the segmented brain image. The intensity features mainly give first order statistical information on the image and depend on all the individual pixel intensity values within the image. The extracted intensity features include:

1. Mean: the average pixel intensity value within the image. The mean was obtained by using the inbuilt *mean* Python function [31].
2. Variance: how far the pixel intensity values are from the mean within the image. The variance was obtained by using the function within the NumPy library [32].
3. Skewness: the measure of asymmetry of the data around the mean. The skewness was obtained by using the *skew* function within the SciPy library [33].
4. Kurtosis: the amount of probability in the tails of the distribution compared to the normal distribution. The kurtosis was calculated using the function within the SciPy library [34].

For the skewness and kurtosis, the SciPy functions require that the input array on which these statistics are to be performed should be 1-dimensional. For that reason, a copy of the image matrix was made, and the Python default *flatten* function was used to convert the 2D image array into a 1D array as required.

The second set of features that were extracted were texture features. In order to obtain these, the Gray Level Co-occurrence Matrix (GLCM) was required. These GLCM matrices were generated using the *graycomatrix* function within the SciKit image toolbox for SciPy [35]. The GLCM describes second order texture analysis, meaning the relationship between groups of two pixels, referred to as the reference and neighbour pixel within the image [36], where the typical neighbouring pixel considered is the one to the right of the reference pixel. This creates GLCM matrices that vary according to the orientation of the image. Considering this, in order to obtain a rotationally invariant GLCM matrix, four individual matrices were generated, each one with a different rotation angle and the mean of all four matrices was obtained. The angles of rotation chosen are 0°, 45°, 90° and 135°, which altogether cover all neighbouring pixels of any reference pixel. Once the rotation-invariant GLCM matrix was obtained, the following properties were obtained:

1. Contrast: a measure of the difference in intensity between the reference and neighbouring pixel.
2. Homogeneity: a measure of how close the distribution of elements within the GLCM matrix are to the diagonal of the same matrix. The lower the image contrast, the higher this value is.
3. Angular Second Moment: quantifies how smooth the image is.
4. Correlation: a measure of how correlated the reference pixel is to its neighbouring pixel.
5. Energy: the sum of squared elements of the GLCM.
6. Entropy: how random an image is, generally taking small values for smooth images.

Programmatically, features 5 through 9 were obtained using the *gretcoprops* [37] function, which takes as parameters the GLCM matrix generated before and the feature in question. For the entropy, the *shannon\_entropy* function within the SciKit toolbox was used on each of the four rotated GLCM matrices, and then divided overall by four to get the rotation-invariant entropy measure.

### Shape Image Features

For the shape feature extraction, the contour image is being used. The extracted features from this image are:

1. Area of bleed: number of pixels within the largest closed contour
2. Perimeter of bleed: number of pixels forming the closed contour
3. Major axis length: the length in pixels of the longer edge of the bounding rectangle
4. Minor axis length: the length in pixels of the shorter edge of the bounding rectangle
5. Angle of rotation: the angle between the x-axis and the major axis of the bounding rectangle
6. Extent: the ratio of haemorrhage pixels to total number of pixels in the bounding rectangle
7. Solidity: the ratio of haemorrhage pixels to the convex hull pixels
8. Convex hull: number of pixels in the convex hull region

Firstly, the largest closed contour was found and stored. It was shown from Mr. Napier’s work that once the haemorrhage image is thresholded, the largest closed contour would be the bleed. By making use of OpenCV’s *findContours* function [38], the list of all closed contours was obtained and traversed to determine which had the largest enclosed area. Once this contour was found, the perimeter of this contour was found using the *arclength* function [38]. The smallest bounding rectangle and its properties were found by making use of OpenCV’s *minAreaRect* function [38], which was then used to find the major and minor axis length and rotation. Finally, the *convexHull* function [38] was used to determine the convex hull of the haemorrhage area, with the area of the convex hull calculated in the same way as the area of the haemorrhage itself.

Combined altogether, these features would then form a row in the training or testing input matrix which would eventually be passed through the neural networks.

## Neural Network Design

For the purpose of this study, it was decided that two different neural network architectures will be tested to compare between them. It was chosen to implement a single hidden layer (1HL) neural network and a deeper, two hidden layer (2HL) neural network, which structures can be seen in Figure 3.2. This was done so as to analyse the effect of how the distribution of the hidden layer neurons would affect the confidence of the final classification. The chosen activation function for these neural network structures is the Sigmoid activation function.

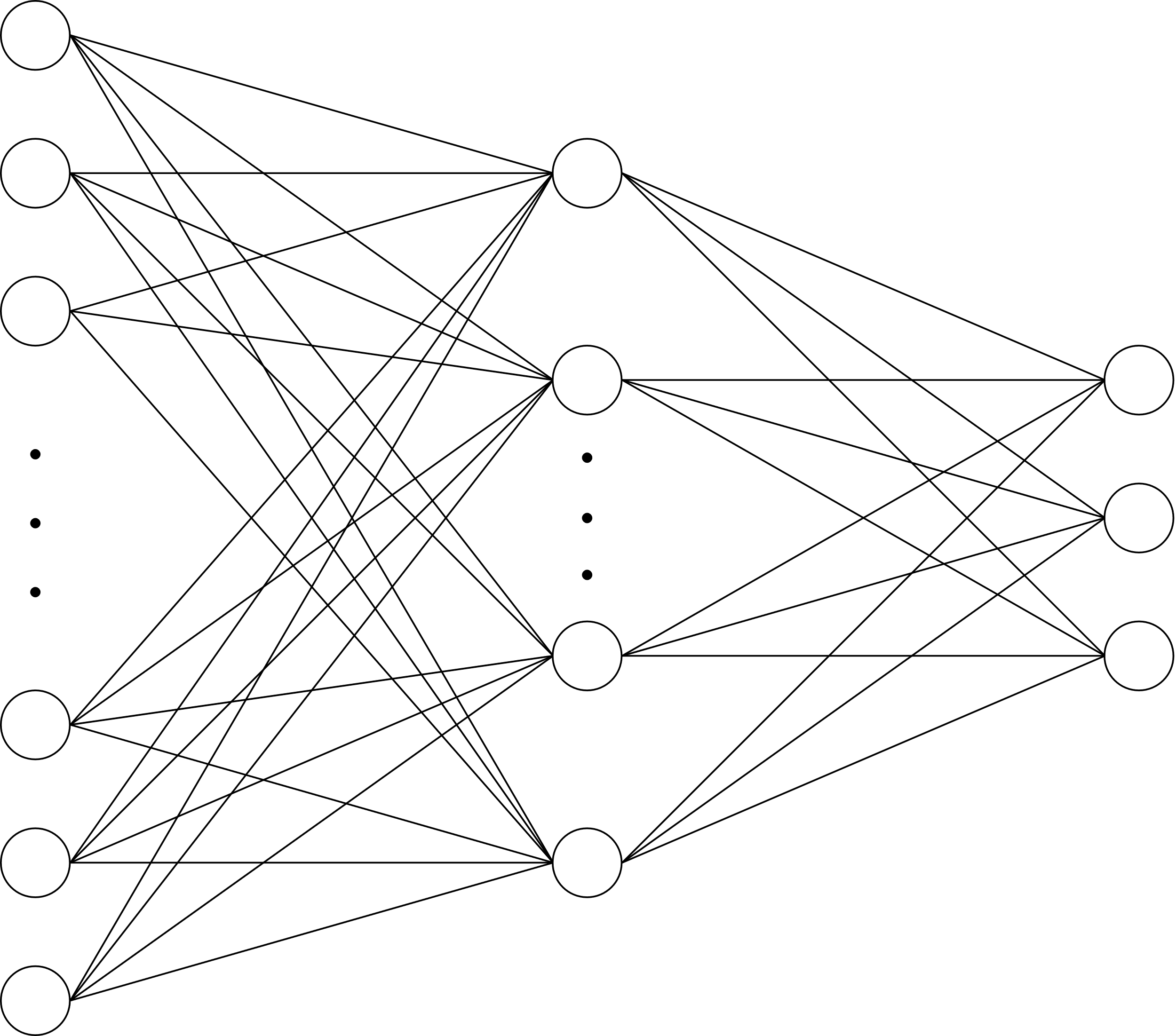
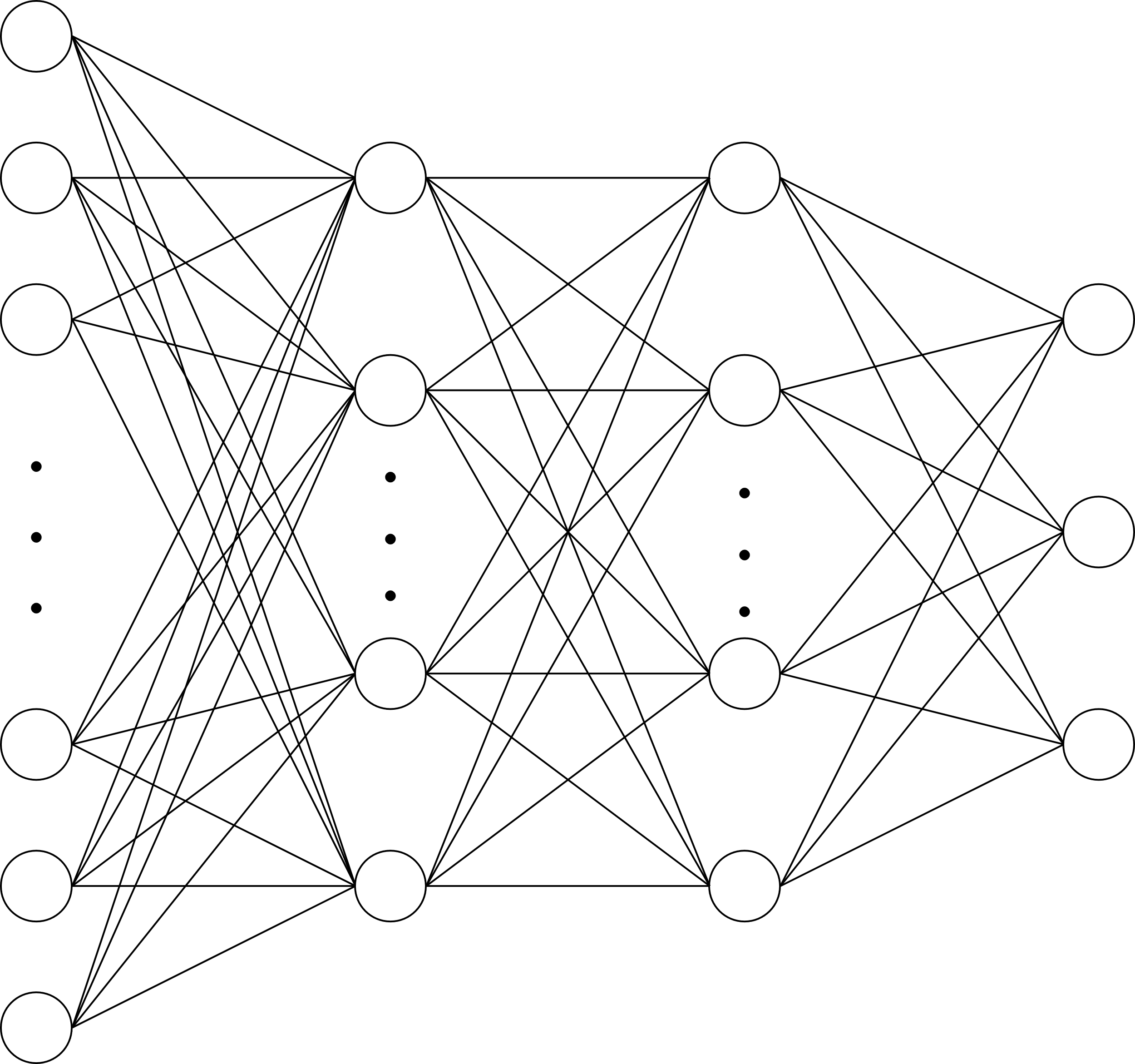


Figure .: Structures of the Neural Networks designed in this system - 1HL (left) and 2HL (right)

Considering how a neural network works, it was determined that the key factors that effect the performance of the design are the number of neurons in the hidden layer and the learning rate of the network. The learning rate is defined as a hyperparameter that controls how much the weights are being adjusted with respect to the gradient descended [39]. Essentially, the learning rate represents the size of the step taken towards the network reaching a local minimum and converging. These two variables were used to perform a parametric analysis on the networks, so as to understand what occurs when these two values vary. It was decided that 10 different learning rates would be tested, between 0.005 and 0.05, whereas the number of neurons in the hidden layer will vary from 4 to 25. This resulted in a total of 220 different network settings, all of which were trained and tested.

Programmatically, a neural network is built as a set of matrices, each representing a different element within the network structure. This means that there is a total of five matrices for the 1HL structure and seven matrices for the 2HL structures. These matrices represent the input layer, which contains 18 neurons, the output layer which contains 3 neurons, the hidden layers, which contains 4 to 25 neurons, and the different sets of weights connecting each layer mentioned above to the next. The input layer matrix is a single row matrix, containing the 18 feature values extracted from one pair of images, as described above. The output layer matrix is once again a single row matrix, containing a number between 0 and 1 which represents what the neural network ‘thinks’ is the category under which the input image belongs to, based solely on the inputted 18 features. The hidden layer matrix is once again a row matrix, where the number of columns represents the number of hidden layer neurons. The weight matrices are 2D matrices, where the number of rows is equivalent to the number of neurons in the left-hand-side neuron layer and the number of columns is equal to the number of neurons in the right-hand-side neuron layer.

Upon initialisation of the program, the weight matrices of the neural network were initialised to a random value between -1 and 1. Following this, the matrix containing all the extracted features of all the images is normalised, such that all values passing through the neural network are normalised. As indicated in Section 2.4.3.3, a neuron within an ANN also contains a transfer or activation function. The main purpose of this function is to map the value of the weighted sum to a number which determines if the output of this neuron is to be ‘on’ or activated or not, in its most simplistic form. For this program, it was chosen to implement the sigmoid activation function to all layers within the network, excluding the input layer. The sigmoid function is described as

The sigmoid function maps any input value into a probabilistic value between 0 and 1, with the midpoint of the curve falling at input 0 which results in an output of 0.5. In this way, the neural network is always working with normalised data, hence making it easier to obtain probabilistic values at the output.

## Neural Network Training Phase

Any neural network system can be considered as divided in two main phases of operation, which are the training and testing phases. In the training phase, each row of the generated feature matrix, meaning all the features for each image in the training set, were passed through the network to generate the output, known the feedforward step, the difference between the expected output and the actual output was computed and the weights within the network were adjusted, going from the output to the input, which step is called backpropagation. This procedure was repeated for all the rows in the feature matrix, which marks the end of one epoch.

The number of epochs executed to train the network depended on the recorded difference between the weights at the end of the last executed epoch and the one before that. It was decided that the networks will continue to train until the average difference between the newly computed weight matrices and the existing ones is less than 0.000001. Once the difference reaches that number, the network would be assumed converged, since any changes smaller than that are considered negligible. The procedure of training the networks was applied to both the 1HL and 2HL structures simultaneously and was repeated for every network parameter variation, which is 220 times.

### Feedforward

In the feedforward step of training a neural network, the input is fed to the network, and after a series of matrix multiplications, a result is obtained. For the case of the IHL structure, the input matrix, which is a row of the feature matrix extracted before, was matrix multiplied with the first weight matrix for the structure. The resulting matrix was then passed through the sigmoid activation function to obtain values for the hidden layer structure that were between 0 and 1. This result was then used as the ‘input’ matrix for the next step of the feedforward algorithm. The sigmoid activation function output was once again matrix multiplied with the second weight matrix, which result was passed through the sigmoid activation function once more to obtain the neural network computed output, which signifies the probabilistic value of what the network ‘thinks’ the image contains. For the 2HL structure, the same procedure applies, with the difference that there is an additional weight matrix, which connects the ‘extra’ layer within the network.

### Difference Calculation

The sigmoid activation function output at the output layer displays the computed result of classification by the network. In the training phase of the network, this produced result was compared to the actual results, which are obtained directly from the report corresponding to the CT patient case for this application, to obtain a measure of the difference between the obtained result and the actual result. The measure chosen is the Mean Square Error (MSE).

In order to get the actual results, the actual classification of each bleed image was written down manually in a text file, which is stored in a specific directory, as referenced in the code. Each image’s diagnosis is written in the abbreviation of the type of bleed, namely EDH, SDH or ICH. Each line in the text file refers to a different image, and thus the order in which the images are read in the feature extraction step must match the order in which the bleed’s classification is written in the file. Given that all the images are referenced with a case number in front, this issue was mitigated by reading the files in sorted order which was done by wrapping the *glob* function used to get the filenames if the images in a sorting function. Once the text file was read, each line was parsed, checked which of the three different classifications is written and add a row in the expected output matrix being compiled. Since the network has three different classes, and three different output neurons which get results of their own, the expected output for any of the three cases would contain 3 elements, where one of which would be ‘1’ and the rest would be ‘0’, and the element in which the ‘1’ is placed is changed for every class.

As mentioned above, the MSE is being used to compute the difference between the difference between the computed result and the actual result. The MSE is calculated as follows:

Where *N* is the number of elements in the matrices, *Y* is the actual result and is the computed result from the neural network. This error was computed for every row in the feature matrix for the training set and the average of this value was computed and stored in a text file, to be used in checking if the neural network converged or not at a later stage.

### Backpropagation

Once the MSE for the error was computed, the backpropagation process is initiated. The backpropagation algorithm is a way of altering weights within a supervised learning neural network architecture. The difference between the expected output and the actual output is computed, and this error is propagated backwards through the neural network and altering the weights slightly every time, with the intention of minimising the observed error. The weight adjustment process is done layer by layer, moving from the output layer to the input layer.

Starting from the output layer, the error is calculated by subtracting the value of the expected output from the actual output, and the resulting error matrix was transposed. Following this, the weight matrix connecting the output layer to the previous layer was scalar multiplied by . This result was then multiplied element-wise by the transposed derivative of the activation function matrix, computed in the feedforward pass of the neural network. Mathematically, the error for the output later can be expressed as

Whereas for any hidden layer going form index *i* to *j*, the error for hidden layer *i* can be described as

Where , *W* is the weight matrix going from layer *i* to *j*, is the sigmoid derivative of layer *i* and is the sigmoid activation matrix for layer *i.*

Once the errors were computed, the following step involved computing the weight updates which will be then added to the weights themselves. In general, the weight updates for the weight set going from layer *i* to *j, ,* is obtained by multiplying the learning rate element wise to the matrix result of the scalar multiplication of the error for layer *j* and the sigmoid activation matrix of layer *i*.

For the case of the weights leaving the input layer, is replaced with the input matrix, which is the current row in the feature matrix. Once these updates are computed, they are added to the corresponding weight matrices [24] [40].

This process was repeated for each of the 220 network variants, until convergence was reached. Once the network converges, the training routine stops, and the program stores all weight matrices along with some other data such that it can be accessed in the testing phase and the recorded values can be analysed.

## Neural Network Testing Phase

The testing phase of the neural network is much less cumbersome and time consuming when compared to the training phase. This is mainly due to the fact that testing the network only consists of one feedforward step, as opposed to the numerous feedforward and backpropagation iterations in the training phase. Another reason is that the number of images, or CT cases, being used for testing is much smaller.

The process of testing the neural network is thus identical to the feedforward stage in the training part of the neural network. The difference in the testing phase is how the input images are treated. Whereas in the training phase, multiple bleed images from the same case were treated as individual cases to increase the total number of cases, for the testing phase one CT case is considered at a time. All images marked with haemorrhage are stored in a directory which is different from that allocated for training images. These images are fed into the classification program where the same feature extraction process is used to obtain the feature matrix, which is used as input to the testing phase. The weight matrices are no longer initialised using random weights but loaded from the directory in which they were stored in the training phase. For each network variant, a total of five weight matrices were stored after the training was completed, two belonging to the 1HL structure and three belonging to the 2HL structure. These matrices are loaded from file using the *loadtxt* function from NumPy. Once these matrices are loaded, each row in the test set feature matrix is passed through the neural network, and the result at the output layer is recorded, since it provides the output prediction of the neural network.

Once the full image set has passed through the network the result matrix for both the 1HL and 2HL structures were passed through a procedure to print the images with the final result of the network’s classification. The result matrix is first multiplied elementwise by 100 to get the percentage value of the accuracy. Following this, the index at which the largest probabilistic value is located is found and based on this index the bleed classification is determined. This was done using the opposite procedure to that used when reading the actual results for the training phase of the network. Following the identification of which element has the highest percentage, a check is being made to find which image gave the highest percentage confidence over all, which will then be used to display the result on screen. For each image, a result string was put together containing a reference to the image, the type of bleed detected and the percentage confidence in the result. These result strings were written to a text file, which was stored locally so as to have a record of the results for each individual image within a set.

It was decided that the final result presented to the person using the system should be the percentage confidence of the entire case. For this purpose, the average percentage value for each of the three classes was calculated and the result was formatted in a string containing the type of bleed and the percentage confidence for the whole case overall. This final string was also appended to the same text file containing all the individual results.

As a final step of the system, the image pair that refer to the case with the highest percentage probability was retrieved from the original image filenames list. The largest closed contour in the binary threshold image was once again found. At this stage, the grayscale segmented brain image is converted to 3-channel RGB by using OpenCV’s *cvtColor* function [41]. This was done so that the coloured bleed perimeter can be seen on the image. Furthermore, by making use of the *drawContours* function [42] within OpenCV, the perimeter around the closed contour marking the bleed was superimposed on the grayscale image. In order for this perimeter to be distinguishable from the rest of the image, a vibrant green colour (RGB 0, 255, 0) was chosen. To make the results more readable to the person using the system, it was decided to print the percentage confidence of the classification on the grayscale image with the drawn bleed perimeter. To do this, the *putText* function [42] was used to place the final result string onto the image. The text was set to be printed at the top of the image in white text.

# Testing and Evaluation

# Conclusion and Further Work

# Bibliography

|  |  |
| --- | --- |
| [1] | A. M. Naidech, “Intracranial Hemorrhage,” *American Journal of Respiratory and Critical Care Medicine,* vol. 184, no. 9, pp. 998-1006, 2011. |
| [2] | B. Pourghassem and H. Shahangian, “Automatic brain hemorrhage segmentation and classification in CT scan images,” in *proceedings of the 8th Iranian Conference on Machine Vision and Image Processing (MVIP)*, Zanjan, 2013. |
| [3] | B. Sharma and K. Venugopalan, “Classification of hematomas in brain CT images using neural network,” in *proceedings of the International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*, 2014. |
| [4] | M. Nadrljanski, “Attenuation coefficient | Radiology Reference Article | Radiopaedia.org,” Radiopaedia.org, [Online]. Available: https://radiopaedia.org/articles/attenuation-coefficient. [Accessed 28 02 2018]. |
| [5] | W. |. W. H. Organization, “Who.int,” WHO | World Health Organization, [Online]. Available: http://www.who.int/diagnostic\_imaging/imaging\_modalities/dim\_plain-radiography/en/. [Accessed 28 02 2018]. |
| [6] | P. Armstrong, Diagnostic imaging, 6th edition ed., Blackwell, 2009. |
| [7] | “Computed Tomography (CT),” National Institute of Biomedical Imaging and Bioengineering, undated. [Online]. Available: https://www.nibib.nih.gov/science-education/science-topics/computed-tomography-ct. [Accessed 28 02 2018]. |
| [8] | M. Nadrljanski, “Computed tomography | Radiology Reference Article | Radiopaedia.org,” Radiopaedia.org, undated. [Online]. Available: https://radiopaedia.org/articles/computed-tomography. [Accessed 28 02 2018]. |
| [9] | P. Sprawls, “CT Image Quality and Dose Management,” [Online]. Available: http://www.sprawls.org/resources/CTIQDM/. [Accessed 28 02 2018]. |
| [10] | “Acute CT Brain - Mass effect,” Radiologymasterclass.co.uk, [Online]. Available: https://www.radiologymasterclass.co.uk/tutorials/ct/ct\_acute\_brain/ct\_brain\_mass\_effect. [Accessed 28 02 2018]. |
| [11] | F. Gaillard, “Intracranial haemorrhage | Radiology Reference Article | Radiopaedia.org,” Radiopaedia.org, [Online]. Available: https://radiopaedia.org/articles/intracranial-haemorrhage. [Accessed 28 02 2018]. |
| [12] | K. Doi, “Computer-aided diagnosis in medical imaging: Historical review, current status and future potential,” *Computerized Medical Imaging and Graphics,* vol. 31, no. 4-5, pp. 198-211, 2007. |
| [13] | R. V. M. a. I. C. P. Raha, “Fully automated computer aided diagnosis system for classification of breast mass from ultrasound images,” in *proceedings of the 2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, Chennai, India, 2017. |
| [14] | V. Kumar, F. Mohanty, B. Dash and s. Rup, “A hybrid computer-aided diagnosis system for abnormality detection in mammograms,” in *proceedings of the 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, Bangalore, 2017. |
| [15] | F. B. S. A. F. a. M. B. A. K. Al Zubaidi, “Computer aided diagnosis in digital pathology application: Review and perspective approach in lung cancer classification,” in *proceedings of the 2017 Annual Conference on New Trends in Information & Communications Technology Applications (NTICT)*, Baghdad, 2017. |
| [16] | M. P. P. a. S. Choomchuay, “A computer aided diagnosis system for detection of lung nodules from series of CT slices,” in *proceedings of the 14th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, Phuket, 2017. |
| [17] | J. Napier, *Image processing techniques for brain haemorrhage detection in head CT scans,* University of Malta, Faculty of Information and Communication Technology, Department of Communications and Computer Engineering, 2017. |
| [18] | “What is Machine Learning? A definition - Expert System,” Expertsystem.com, [Online]. Available: http://www.expertsystem.com/machine-learning-definition/. [Accessed 28 02 2018]. |
| [19] | J. Brownlee, “Supervised and Unsupervised Machine Learning Algorithms - Machine Learning Mastery,” Machine Learning Mastery, [Online]. Available: https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/. [Accessed 28 02 2018]. |
| [20] | N. T. N. U. Department of Computer Science and Information Engineering, “Classifier Training and Evaluation,” Department of Computer Science and Information Engineering, National Taiwan Normal University, Taiwan. |
| [21] | K. Zakka, “A Complete Guide to K-Nearest-Neighbors with Applications in Python and R,” Kevinzakka.github.io, 2018. |
| [22] | S. Patel, “Chapter 2 : SVM (Support Vector Machine) — Theory,” 03 05 2017. [Online]. Available: https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72. [Accessed 21 04 2018]. |
| [23] | S. Ray, “Understanding Support Vector Machine algorithm from examples (along with code),” 13 09 2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/. [Accessed 21 04 2018]. |
| [24] | A. Cortis, “Artificial Neural Networks,” *The Collection,* vol. 7, pp. 28-33, 2003. |
| [25] | M. A. Nielsen, “Neural Networks and Deep Learning,” 2018. [Online]. Available: http://neuralnetworksanddeeplearning.com. [Accessed 21 04 2018]. |
| [26] | U. o. Toronto, “Artificial Neural Networks Technology,” [Online]. Available: http://www.psych.utoronto.ca/users/reingold/courses/ai/cache/neural\_ToC.html. [Accessed 21 04 2018]. |
| [27] | H. P. Bahare Shahangian, “Automatic Brain Hemorrhage Segmentation and classification in CT scan images,” in *proceedings of the 8th Iranian Conference on Machine Vision and Image Processing (MVIP)*, Zanjan, 2013. |
| [28] | U. B. a. M. U. S. Perera, “Intelligent brain hemorrhage diagnosis using artificial neural networks,” in *IEEE Business, Engineering & Industrial Applications Colloquium (BEIAC)*, Kuala Lumpur, 2012. |
| [29] | P. S. Foundation, “16.1. os — Miscellaneous operating system interfaces — Python 3.6.5 documentation,” Python, [Online]. Available: https://docs.python.org/3/library/os.html. |
| [30] | G. Kumar and P. K. Bhatia, “A Detailed Review of Feature Extraction in Image Processing Systems,” in *proceedings of the 2014 Fourth International Conference on Advanced Computing & Communication Technologies*, Rohtak, 2014. |
| [31] | “numpy.mean — NumPy v1.14 Manual,” Docs.scipy.org, [Online]. Available: https://docs.scipy.org/doc/numpy-1.14.0/reference/generated/numpy.mean.html. |
| [32] | “numpy.var — NumPy v1.14 Manual,” Docs.scipy.org, [Online]. Available: https://docs.scipy.org/doc/numpy-1.14.0/reference/generated/numpy.var.html. |
| [33] | “scipy.stats.skew — SciPy v1.0.0 Reference Guide,” Docs.scipy.org, [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.skew.html. |
| [34] | “scipy.stats.kurtosis — SciPy v1.0.0 Reference Guide,” Docs.scipy.org, [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.kurtosis.html. |
| [35] | “GLCM Texture Features — skimage v0.14dev docs,” Scikit-image.org, [Online]. Available: http://scikit-image.org/docs/dev/auto\_examples/features\_detection/plot\_glcm.html. |
| [36] | “The Grey Level Co-occurrence Matrix, GLCM (also called the Grey Tone Spatial Dependency Matrix) | Personal and research,” Ucalgary.ca, [Online]. Available: http://www.ucalgary.ca/mhallbey/glcm1. |
| [37] | “Module: feature — skimage v0.14dev docs,” Scikit-image.org, [Online]. Available: http://scikit-image.org/docs/dev/api/skimage.feature.html#skimage.feature.greycoprops. |
| [38] | “Structural Analysis and Shape Descriptors — OpenCV 2.4.13.6 documentation,” Docs.opencv.org, [Online]. Available: https://docs.opencv.org/2.4/modules/imgproc/doc/structural\_analysis\_and\_shape\_descriptors.html. |
| [39] | F. Gaillard, “Subdural haemorrhage | Radiology Reference Article | Radiopaedia.org,” radiopedia.org, 2018. [Online]. Available: https://radiopaedia.org/articles/subdural-haemorrhage. |
| [40] | F. Galliard, “Extradural haemorrhage | Radiology Reference Article | Radiopaedia.org,” Radiopaedia.org, 2018. [Online]. Available: https://radiopaedia.org/articles/extradural-haemorrhage. [Accessed 24 02 2018]. |
| [41] | “Education. Whats an MRI,” Multiple-sclerosis-research.blogspot.com, 10 01 2015. [Online]. Available: http://multiple-sclerosis-research.blogspot.com/2015/01/education-whats-mri.html. [Accessed 28 02 2018]. |
| [42] | “Spontaneous Intracerebral Hemorrhage,” Clinical Gate, 03 12 2015. [Online]. Available: https://clinicalgate.com/spontaneous-intracerebral-hemorrhage/. [Accessed 28 02 2018]. |
| [43] | “Computed tomography,” TheFreeDictionary.com, [Online]. Available: https://medical-dictionary.thefreedictionary.com/Computed+tomography. [Accessed 28 02 2018]. |