Technological University Dublin

Final Year Project

Computing with Data Analytics

Classification of Brain Cancer from MRI images using Convolutional Neural Networks

---

Author:
Jason Walsh

Supervisor:
Sean Mc'Hugh

Contents

1 Project Plan & Research

1.1 Project Summary

1.1.1 Background

Image processing is a very important subject in the field of medicine. X-ray images, CT scans and MRI images are all methods medical physicians use to scan for abnormalities in the human body, these techniques can discover the slightest form of cancer in the human body. The basic goal of the project is to use the MRI image scans and perform a type of image processing to detect the presence of some form of cancer that is located somewhere on the brain. The subject of the images will be tumour classification and the model will predict whether the image contains a tumour or not.

Brain tumours are an abnormal growth of cells. The images that will be dealt with will involve two types of tumour. Malignant and Benign. Malignant tumours are considered a form of cancer as the tumour will invade surrounding cells and gradually spread to other areas of the host. Benign tumours do not spread like malignant tumours, so they are not considered cancerous also meaning they do not invade their surrounding cells. The outlook for benign tumours is considered very good but can be serious if the tumour is applying pressure to nerves or vital blood vessels.

The importance of medical image analysis has never been greater. Using image segmentation and other techniques to detect boundaries of images and perform analysis on digital images is pushing the boundaries for new medical breakthroughs. Cancer detection could potentially save many lives by identifying the presence of either a malignant or benign tumour.

1.1.2 Objectives

To achieve this goal of fully creating this model that can predict brain cancer the following objectives will need to complete several objectives:

- Pre-process the images acquired.
- Preform Image segmentation and feature extraction.
- Build the model that will predict the presence of a tumour with minimal error.

To successfully pre-process the images, OpenCV will be used to perform tasks such as grey scaling, normalization and re-scaling. Once the images have been fully pre-processed image segmentation and feature extraction can begin. To build the model the python library Keras will be used, which runs directly on top of Tensor Flow. This is where the majority of the work on this project will commence.

These objectives will later be split into different approaches and then into separate phases that will help with further development of the project. These phases will break down the main three objectives listed into smaller more achievable goals that can be achieved in perfect order and with a feasible amount of time.

1.1.3 Prototyping and Testing

Testing the model and the data is one of the most important stages of this project. To successfully training and testing the model will be a guide to choosing the best optimal parameters for the goal. Model building is a difficult task and it can depend entirely on the type of input data used. The data collected varies in three different perspectives of the human brain, this will cause training issues with the models as the data is at times extremely random. Although the majority of the data is in one perspective this will still have a negative effect on the models. This is why prototyping and testing are important as more time can be allocated to collecting higher quality data or slowly mining away the best performing model that best fits the data originally collected.

Prototyping and testing will allow for the addition or extraction of existing stages that contribute to the outcome of this project. The Image Segmentation and Feature Extraction goal is not important to the

outcome of this project. If time is an issue this stage will be made redundant as its outcome does not affect the final goal of this project.

1.1.4 Data Requirements

The data collected is comprised of both healthy where the patient's brain is free of any such disease and un-healthy where the image contains visible signs of brain cancer. The images are 2 specified formats JPG and PNG Each brain scan is different from another, some vary in size and shape and others in colour. When medical physicians undertake an MRI test on their patient, they will perform multiple different scans to get different viewpoints of the suspected case of cancer.
Such scans would include:

- Fluid-Attenuated Inversion Recovery Scan
- T1 Weighted Scan
- T2 Weighted Scan

The data obtained contains a mixture of each of these scans. The data is also varying in perspective. Sticking to only one perspective would limit the complexity of the model this is why all perspectives of the brain were chosen to increase the overall complexity of the model. All images collected were collected through a search engine with a single query. To enforce reproduce-ability a small note section was used to record all queries that were searched to find these images. All images collected are easily identifiable from one another and the cancerous images all have visible signs of a brain tumour.

Collecting more data is an important step but time must be allocated accordingly as too much time spent on one single stage could be detrimental to another more important phase. There are automatic ways of increasing the number of images in the data-set and these steps will outline much later in the Generalization sub-section. While collecting the data file formats such as MHA and DCM where heavily avoided as these are 3d representations of the brain rather than a normal 3-dimensional image. The most common file formats such as JPG and PNG were chosen as these where the easiest file formats to work with. Although MHA and DCM are medical imaging formats where countless amounts of data are stored, these formats represent 4-dimensional data which is out of the scope of this project.

There will most likely be only one file format used as there have been previous studies on the effects of varying file formats on models. As the amount of JPG images outweighs the PNG and the converted GIF images, these images will be converted to JPG so there can be no possible risk of file formats affecting the pre-processing stage or any later stages of this project.

1.1.5 Risk Assessment

Several obvious risks could alter both the progression of this project and the predictive power of the model. There is a lot of risks associated with the data-set. Once Model Selection & Model Investigation begins there will be an indication of how the model performs using the small amount of data that was collected. This is why using a small data-set is a risk on the outcome of the project. No amount of data is perfect, and no individual image is perfect.

Meaning when the pre-processing stage begins there will be images that will have to be cut from the group either because they are too low resolution, or they are simply unusable. Another risk will come from what software is used to build my model. The software chosen is Python with Keras, however as the project advances, this may be a regrettable decision if obstacles occur that cause my work to suffer from severe setbacks. These risks are not for certain and are only precautions to future mishaps that may occur as work on the project progresses.

When certain stages in this project are reached, different techniques can be used to achieve different results. For example, once pre-processing the images begins different pre-processing steps will need to decide upon to achieve the best result. These steps all depend on the data that was originally collected. So, each

step must be carefully considered and researched before any steps are taken. This is considered a risk because either of these phases (pre-processing, image segmentation and feature extraction) could go wrong and cause misleading results in the Model Selection & Model Analysis section.

1.2 Project Methodology

1.2.1 Introduction

To begin creating this project a plan of splitting the overall project into several stages which are accompanied by several sub-phases. This decreases the workload required on each stage and can give a better understanding of what goals must be achieved to acquire project outcome. One of the crucial steps is the pre-processing stage which will utilize the software OpenCV. This software will allow for the process of pre-processing which will crucial to cleaning the images for the later stages. Some steps could include de-noising, downsampling, feature scaling etc. These steps taken to pre-process the image all depends on the images collected. What size, shape and resolution they are etc. These properties all affect how they will perform in the neural network.

To effectively plan this process out the project will be split into two objectives. A Project Approach which will outline each step that will be needed to successfully predict the presence of a tumour and A Project Phase's which will group the steps outlined into 3 stages/phases where deliverable dates can be set for each step that needs to be completed.

1.2.2 Project Approach

To create this project, a set of goals must be considered to help decide on how each project stage will progress and the main overall objective that must be achieved from each goal. A simple set of goals have been compiled, each containing a short description of the goals overall objective and how it will affect the progress of the project. The goals listed in order are as follows:

1. Collect the data
    - Collecting and creating data-set is a crucial step. The main images that will be sought for will be images which are completely healthy and contain no signs of cancer. Images that contain an obvious presence of brain cancer will also be one of the types of the main images collected during this process. These images are different shapes, sizes, colour and completely different perspectives. No two images are the same so the model that will be built will always be trained and tested on new data. This step may seem simple and ineffective but the more images that are collected would provide the model with better predictive power.
2. Pre-process the data
    - There are many stages that must be undertaken to effectively pre-process the data that obtained. The steps to pre-process the images are as follows:
        - Re-scale the images, it is important to have a uniform aspect ratio across all images that were collected.
        - The images must be scaled to the same pixel size ($height \times width$).
        - Decisions must be drawn on whether the data will be normalized or standardized, this procedure will allow for faster convergence.
        - The images will need to be converted to greyscale and also removed of any corrupted noise present in the images as noise reduces the overall image accuracy. To remove noise, various filters will be introduced such as the Median and Gaussian filters which are known to help remove the two most popular types of noise.

3. Preform image segmentation

    - The images can now be inputted to perform image segmentation. Image segmentation allows for the further gain of insight into what information the digital images contains. Segmentation breaks the image into multiple sets of pixels, each pixel is tagged, and each tagged pixel will tell its own story of what features are shared with the neighbouring tagged features. Several segmentation techniques will be used to experiment with to find the best solution for the objectives. The main 3 which will be researched are as follows:
        - Threshold Segmentation
        - Edge Detection
        - Skull Stripping

4. Feature Extraction

    - Due to the importance of minimal error because of the type of classification being preformed (Brain Tumour Classification) an accurate prediction of the tumour's location inside the complex structure that is the human brain. The purpose of feature selection is to identify the difference from image to another. It will look for unique properties to precisely identify the image. These features that will be extracted will be used to help classify the image on whether it contains a tumour or not.

5. Classification

    - The final step in the project approach involves the completion of our goal. To predict whether the image contains a tumour or not. This step will also include various Generalization and Regularization techniques which will be used to help the model achieve its final goal.

1.3 Project Plan

1.3.1 Project Phases

For the project phases, there will be one major phase for each approach listed. The major phase of this project will mainly be the pre-processing phase. This stage is critical to the development of the project as it is where the data will be cleaned which is crucial for later phases if the data is not cleaned properly this can have negative effects on the outcome. There will be a significant amount of time dedicated to this stage to assure the data is properly pre-processed. Once this phase is complete, I will enter the second phase which is Model Section & Model Analysis.

Phase 1 will mainly include the collection of more data and the pre-processing of that said data using OpenCV. Phase 2 will include the model building phase where the model will be continuously tested and trained to get an understanding of how the model functions. The final phase the shortest of the 3 will include classification and modifications, where the models will be Generalized and predictions will be made.

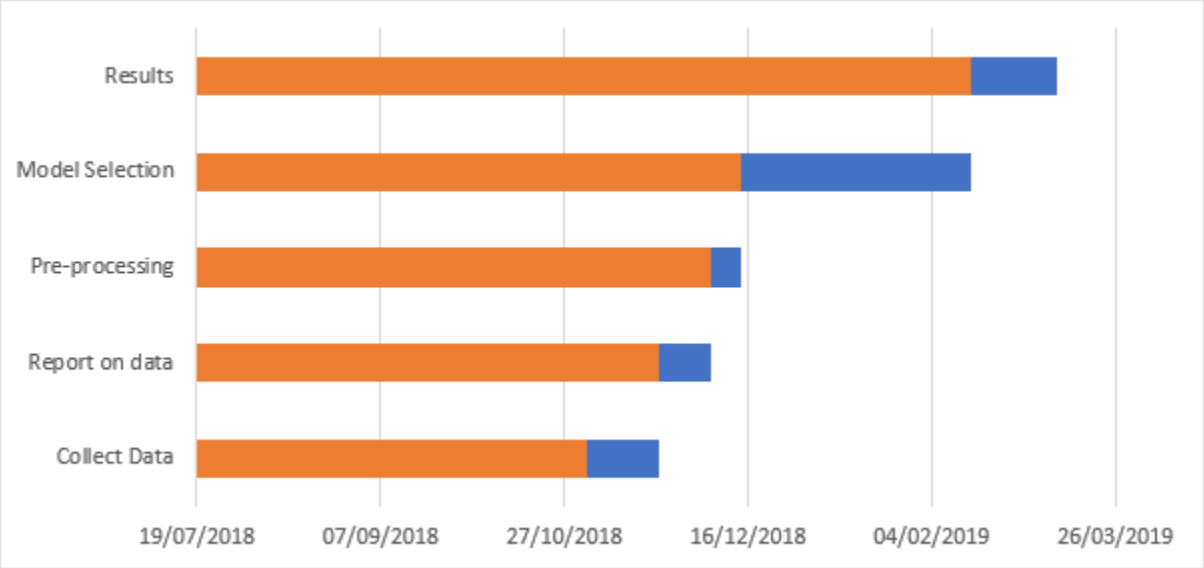
1.3.2 Deliverables and Milestones

There are several deliverables expected for this project. Along with these deliverables, several milestones have been included to enforce a time limit on each stage which will need to be completed by each due date. The deliverables expected for this project are as follows:

- Research Document
- Data preparation
- Data pre-processing
- Model selection and performance

After a draft of each document is complete work on the next phase will begin. For the Data preparation and pre-processing stage, work will begin on phase 1 of the project which includes research into the data and the process of altering the data for the next phase. The model selection and performance deliverable will include the results from the pre-processing stage, these findings will include what methods will be used for each outcome and how each outcome was obtained. The final deliverable is the overall main document which includes all methods, observations and experiments which were performed to achieve the outcome. This will make up the bulk of the overall project.

1.3.3 Gantt Chart



1.4 Conclusion

Each project phase presents a challenge that requires both extensive research and implementation. Using the Gantt Chart and Deliverables has set an achievable deadline so the progress on the project will continuously push onwards. Understanding the risks associated with this project is important and should not be overlooked. As the project progress and stages are completed the project's documentation should also be completed in line with the completion of each stage. This will allow for more achievable goals which will eventually give the outcome needed.

2 Pre-processing

2.1 Images

2.1.1 Background

Images have existed for decades and have been the centre of attention for Artificial Intelligence researchers who strive to emulate the human mind in a robotic or machine form. Optical images are a 2d representation of a physical object in space, which is obtained by the propagation of light through some optical system which depicts object contours and image features. Digital cameras in the modern age capture their picture through a model known as the pinhole method. This essentially allows an individual to capture the shape of an object from projected light. The method captures a pencil of rays, all rays must go through a single

point known as the centre of projection and the image formed from the procedure is referenced as the image plane. The image captured is not always of use, which is why images need to be pre-processed for further analysis.

2.1.2 Pre-Processing

Images captured are not perfect. Images are prone to having defects which will affect the overall quality of the image. Images taken can have low contrast or wrongly shown colours which can confuse their viewers. Images can also be noisy or blurry and can also have non-uniform lighting when the image is captured in a dimly lit location. These defects can be improved upon with the idea of image pre-processing, a way of preserving the original image while enhancing its overall visual quality.

So why should an individual pre-process their images?

- Image improvement for human perception.
  - By improving the overall subjective image quality, the image is now better view able for the viewers.
- Image improvement for machine perception.
  - By improving the quality of the image you can simplify image both image analysis and recognition procedures.
- Image transformation for technical purposes.
  - Change the images resolution or aspect ratio. Can also include steps taken to preform image segmentation where a region of interest in the image is segmented and analyzed for future research.
- Entertainment purposes.
  - Many mobile and computer applications allow their users to alter their images to retrieve artistic impressions, the applications provide the user with tools that can automatically change the visual aspect of the image.

2.1.3 Images as Functions

Machines cannot directly interpret images like us humans do. For machines to read images, images must be represented as functions. Such as f or I from R^2 to R. The function $f(x, y)$ gives the intensity or the pixel value at position (x, y). Furthermore, images can be practically defined over a finite range.

$$f : [a, b] \times [c, d] \rightarrow [min, max] \tag{1}$$

Digital images are mostly represented in Red, Green and Blue (R,G,B) format. These are 3-channel coloured images which are essentially mapped from a 2-dimensional space of locations to a 3-dimensional space of colour intensity values. This is represented in the formula below.

$$R \times R = R^3 \tag{2}$$

Because images can be represented as functions basic arithmetic operations can be performed on the image as the image is essentially a table of numbers ranging in pixel intensity. This allows such operations of the addition and subtraction of one or more image or the addition of a constant value n to produce a new output of z. These operations will help define how the images will be pre-processed, for example finding the overall mean image will give a direct insight into the overall structure of the images collected and the various outliers or noise that may be present in the images.

2.2 Pre-Processing Pipeline

2.2.1 Introduction

As previously discussed images must be pre-processed before any meaningful analysis can be performed on the images. The images collected of both healthy and unhealthy human MRI images have several issues which need to be corrected before the images can be trained and tested using a Convolutional Neural Network. Some issues recorded:

- The images vary in size. Some having the resolution of 500×500 and some ranging much further from this value.
- Some images vary in colour. Meaning the data-set consists of both 3-channel and 2-channel coloured images.
- A lot of the images vary in format. The data-set consists of JPG, PNG and JPEG images.
- The images contain a substantial amount of noise which is effecting their overall quality. This noise varies from Salt & Pepper to Gaussian Noise.

These issues are affecting the overall quality of the images and can negatively impact the performance of our neural network once we begin training and testing the data. These are the general steps of the pre-processing pipeline but more steps which will help for analysis will be implemented depending on their overall effectiveness on the data-set. To perform each procedure outlined above the Python library OpenCV was used for the majority of the pre-processing tasks. To note, however, although OpenCV has a method of displaying an image the said method, unfortunately, does not work well with Jupyter Notebooks causing the notebooks kernel to crash frequently. To counter this, the Python library Matplotlib was used instead for displaying any image results to the screen.

Not all pre-processing tasks will help for each image collected as the images are quite random from one another, varying in size, shape and image projection (Coronal, Sagittal, Axial). The majority of these images share one common trait and this is they are high in image quality. The blurrier the image the harder it would be for a Neural Network to identify the images most important edges. This is why the first procedure carried out before any pre-processing could begin was the removal of 'bad images' which are essentially images which have been deemed unusable based on their bad image quality. Each image removed was replaced with a newly collected MRI image which was of better overall image quality. This process was undertaken to ensure that each image kept a respectful amount of their original image quality after they were fed through the Pre-processing Pipeline.

2.2.2 Image Formatting

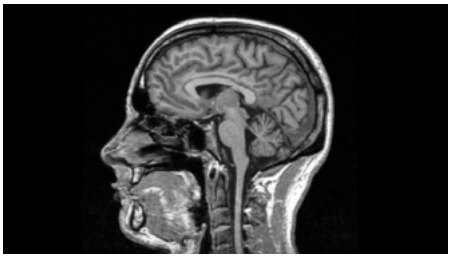
Images varying in formats can have dire consequences on the outcome your neural network produces. To illustrate the effect of this outcome the experiment was undertaken by ..... who visually describes the effect of using different image formats. The result was odd spikes of performance in image segmentation and training which affected the overall performance of the network. To convert the images a small python script using glob to read in and save the images in their new formats was used. To note, no particular research was found on whether or not different image formats affect a Convolutional Neural Networks performance. The research outlined above was regarding a semantic image segmentation project which is quite different then this projects overall main goal. However, the images were still converted as segmentation of the brain tumours from the MRI scans are still a side goal for this project.

2.2.3 Image Scaling

As the images collected vary in size, this can negatively affect the training of the Convolutional Neural Network. Re-scaling the images to a more uniform size can improve the neural network's stability and can increase the overall convergence speed. Furthermore, re-scaling is a necessary and important step before

any experiments can be conducted. Most neural networks will assume the images being used for training & testing are square images, meaning we cannot feed in an image of a different shape. To establish an even image scale across the data-set a size of 64×64 has been chosen. The size chosen is not too small nor too big and should help the neural network converge faster upon its local minimum. The size was chosen based on research conducted on the image beforehand. Any image size could have been chosen however it is recommended that the image size be recursively divisible by 2 this is why the size of 64 was chosen.

Once the image is scaled it is not too blurry were the images features are no longer readable. However, this assumption can not be made as machines see images differently than humans. The overall goal when re-sizing the images was to re-size the images to a respectful $height \times width$ so the network would not need to be substantially large to deal with large images. Smaller image size also allows the images to train quicker. However, the main reason for this size is because the images collected vary in size. As previously stated most of the images do not have the same dimensions. Some image dimensions are below 100 making rescaling these images to a size of 128×128 impossible without adding border regions to the images.

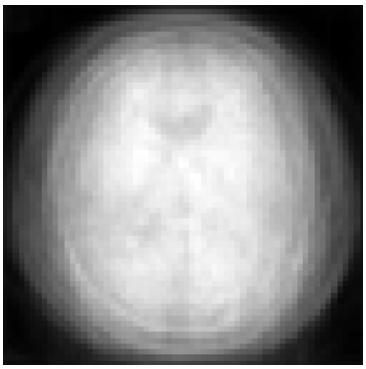


2.2.4 Converting to Grayscale

As previously discussed images can be represented in 2 distinct categories an image with 3 colour channels is considered to be RGB representing the colours Red, Green and Blue. An image with 2 colour channels is considered to be a grey-scaled image where each pixel contained within the image represents a different intensity for the light present in the image. Gray-scaling the images allows for faster training which allows for more time to be allocated to investigating network architectures etc. The process of Gray-scaling was achieved primarily using OpenCV to convert its RGB channel (which is BGR in OpenCV) to GRAY. The figure below shows a highly coloured MRI scan of the brain which is then converted to grey-scale. The features and edges are now more pronounced and will work better with edge detection algorithms and various steps undertaken by Convolutional Neural Networks.

2.2.5 Mean & Standard Deviation

A useful but not entirely necessary step in pre-processing is the use of mean and the standard deviation. Plotting the mean and SD (standard deviation) of an image will show some originally un-noticeable structure or underlying features in the image. Finding the mean and SD will also show potential outliers in the image. Outliers which will be discussed in further steps can be presumed to be un-wanted noise and can be removed using Denoising techniques such as image smoothing. Visualizing the Mean & Standard Deviation of the total images can also give an overall insight into the most prominent structure evident throughout the data-set. The figure below is off the Mean & Standard Deviation off the data-set collected. The figure indicates that a small portion of noise which looks to be a form of Gaussian Noise is shrouding the images and reducing their overall quality. Finally, the overall prominent structure evident within the data-set is unsurprisingly the shape of a brain.



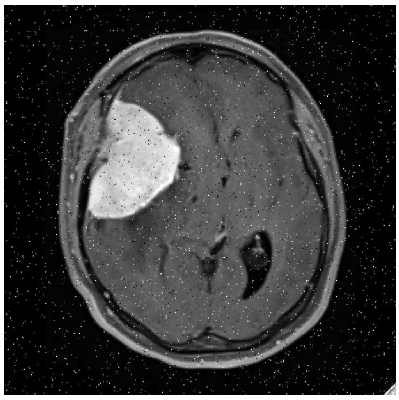
2.2.6 Image Smoothing

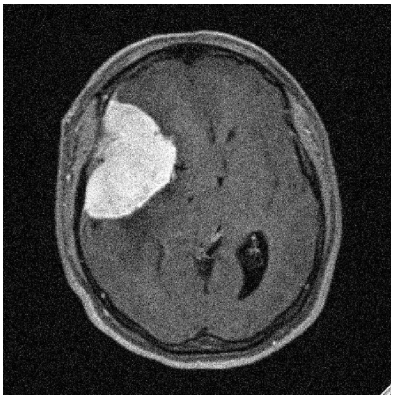
Image de-noising is a form of image smoothing where a particular algorithm is used to smooth the image free of noise. This is an important step as the goal of this project is to identify the brain scans which contain the cancerous tumours. The most common types of noise are Gaussian, Salt & Pepper and Impulse Noise.

- Salt & Pepper Noise: Represents random occurrence's of black and white pixels that occurs in images.
- Impulse Noise: Represents random occurrence's of white pixels.
- Gaussian Noise: Represents variations of intensity drawn from a Gaussian distribution.

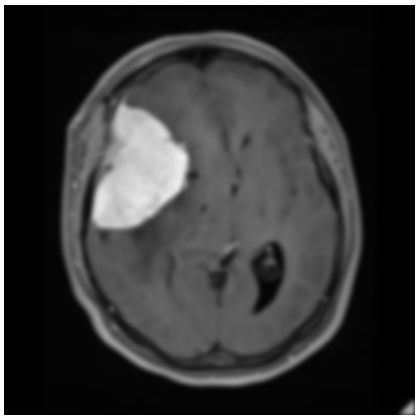
The figures below represent both Gaussian Noise and Salt & Pepper Noise. Although Impulse Noise is quite common there is no evidence of the presence of this type of noise in the data-set, the data-set primarily contains Gaussian Noise and there are several ways to remove noise from an image. Without going into too much detail on the mathematics behind each type of convolution process one must understand that both Correlation and Convolution are built on addition and multiplication, these are linear operators making the whole notion of filtering or smoothing an image a linear operation.

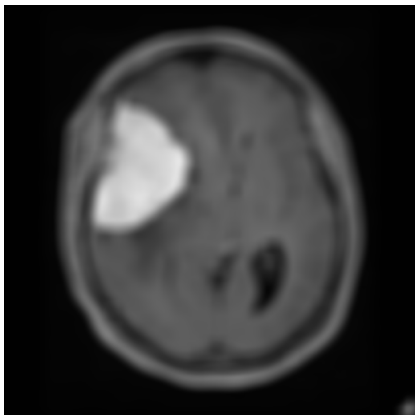
In Gaussian Filters the nearest neighbouring pixels have the most influence, the kernel that is used is an approximation of the Gaussian Filter. The kernel K will be placed over all the middle values in the image and the result will be calculated based on the kernel values. The middle value essentially represents all of the outer pixels or pixels present in the kernels box filter averaged to obtain one single value. The operation is known as convolution. This process essentially averages all pixel values contained within the image. Once the value is averaged using a chosen algorithm the noise contained in the image is overwritten using these new values. Noise in an image is controlled by the value of σ. σ both controls the amount of noise added to an image and the amount of noise removed from an image. The value of sigma when removing noise works over space. Essentially the higher the overall value for sigma the higher the noise that will be added and or removed from the images contained in the data-set.





Gaussian Averaging the method described above is a reasonable solution to the noise that is present in the data-set. However, different types of noise will require different types of filtering. One could not expect good results when using Gaussian Averaging on a data-set that primarily contains Salt & Pepper Noise. Gaussian Averaging operated buy replacing the pixel values by the images local average. When random values are spread out through the image indicating the presence of Salt & Pepper Noise this is where a Median Filter must be used. Median Filters replace the random numbers present in the image (outliers) with the images median filtering value. Median Filters remove spikes in the image which is good for the images impulse and is perfect for removing Salt & Pepper Noise. This filter is also known to be edge-preserving. However, the data-set collected primarily contains Gaussian Noise, only a small portion to be exact which will require the use of a Gaussian Filter with a small σ value to remove any presence of the noise. The figures below represent the effects of σ on the images. The figure on the left has a very small σ value which removes the noise and causes no blur. While the image on the right removes the noise but damages the overall quality of the image causing a heavy blur to be applied to the image. This small experiment indicates that a small σ value will be used along with a Gaussian filter to smooth the images free of noise.





2.2.7 Feature Scaling

Feature Scaling indicates setting all of the numerical data onto the same scale. This can be done either by using Normalization or Standardization. Normalization is arguably the most common of the two feature scaling methods. When using Normalization one would expect their numerical values to be scaled on a range of 0.0 to 1.0. Feature Scaling helps a neural network converge faster which makes this an important step for the next stage of Model Selection & Model Performance. Normalization is a regular choice when data being dealt with does not have an evident bell curve. This distribution would indicate that standardization should be used instead of the latter. This 'rule of thumb' also works for image data as images as functions are essentially a matrix of pixel values. These values are numerical and range from 0 to 255 (RGB). To understand the distribution of the data multiple histograms were plotted to visualize how the data is plotted. The data was sparsely distributed and varied in skewness between the pixel values which represented the light intensities evident in the images. Due to this discovery Normalization was chosen as the choice of feature scaling.

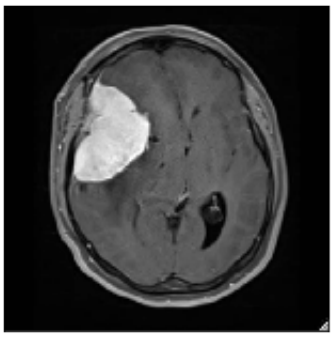
2.3 Image Segmentation

Image Segmentation is the process in which a digital image is separated into multiple segments of pixels. Image Segmentation is used to remove sections of an image so that the features of the image can be analyzed further. The most common type of image segmentation is Thresholding. Using Thresholding a greyscale image can be converted to Binary methods such as K-Means and Otsu's Method are arguably the most commonly used. Thresholding is a form of Region-based Segmentation which could allow for the use of the

Bounding Box method. This section of the report will discuss possible uses for segmentation of the brain tumour. It will cover topics and research that has already been tried and tested on the data-set collected and how the results could be improved.

2.3.1 Edge Detection

Convolutional Neural Networks are used to analyze the features in images. The features refer to the edges of the images. The pre-processing stage is an important stage to any image classification problem where an end prediction is an outcome desired. The problem, however, is images are dirty, they contain noise and unnecessary information. To get an idea of what edges are represented in the images one can use the canny operator. The canny operator in simple terms is an algorithm which will indicate the edges which are present in the images. This can give a helpful understanding of what a Convolutional Neural Network needs to interpret in regards to image classification.



2.3.2 Binary Thresholding

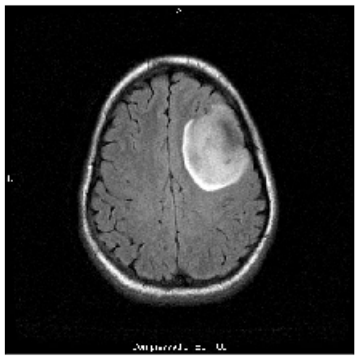
Binary Thresholding and Threshold Segmentation is both the most popular type of segmentation and the easiest. Threshold Segmentation uses its algorithm to divide the collected grey-scale image/s based on different target values in the image. Binary Thresholding will convert the image to a scale of 1 and 0. Segmenting the entire image into different regions. On the data-set collected this was a successful method on 'some' of the un-healthy brain scans. The results returned a segmented tumour which could easily be identified. However, it was quickly discovered that this method would only work if the intensity of the brain tumour was higher than the actual brain. The brain contains white matter which can show in the MRI scan and will not be segmented, as well as white matter the skull of the brain will also appear as a white outer rim. If kept, this could cause later problems for brain tumour classification using Convolutional Neural Networks where the network may mistake the skull as a tumour. This leads on to the process of 'skull stripping'. A process which will use an algorithm or separate program to physically remove the skull from the brain scan.

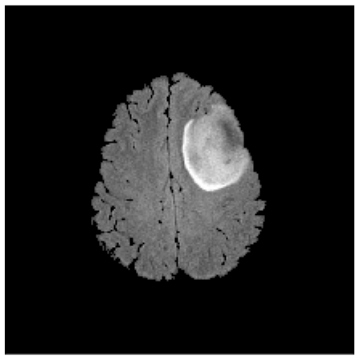
2.3.3 Skull Stripping

As previously discussed, the skull when segmented will appear as an outer rim. Through extensive research, a method has been discovered using the segmentation method 'The Watershed Algorithm'. This algorithm performs a marker-based type of segmentation when the grey-scale image is viewed from a top-down perspective. The areas of high intensity are marked as 'peaks and hills' while the low-intensity areas denote the presence of valleys. These isolated valleys are denoted with different labels. As the water rises, water from different valleys will begin to mix as they are labelled differently. This is prevented by creating barriers, so the water will not merge. When the peaks have fully submerged the boundaries created are the segmentation result.

The results of skull stripping have had greatly improved the chances of segmenting the tumour. As you can see in the above figures the segmentation operation has improved greatly as the tumour has now been isolated from both the brain and the skull. However, this does not work for all images. If the tumour/skull has a very low intensity, then we cannot perform this method. The process of skull stripping is based on the application of Otsu's method. Once applied the image is covered with a mask of the original image but grey-scale. The image intensity is then observed using a histogram if the plot returns a binomial curve then the operation of stripping the skull using the watershed can begin. This method has seen good results but can be improved. One such improvement which will be researched further will be the use of inverted binary thresholding.

Both of these topics can be researched further which could either update or improve on the goal of this project. Topics that could be of assistance and could potentially improve upon existing segmentation techniques are the 'Bounding Box' method. This methods goal is to split the image of the brain into 4 quadrants. Upper left, Lower Left etc. It will do this based on the centre point axis in the image and will use readings from the intensity of the image to locate the tumour. After the tumour has been found models such as Support Vector Machines or K-Means are used to further analyze the regions. (Ségonne et al. 2004)





2.4 Conclusion

The pre-processing stage is one of the most important stages in Image Classification. Properly preparing the images for the Neural Network is a crucial stage as images collected with a different colour pattern, size and shape will cause your network to struggle with the training and testing phases of the network. As the images are now fully pre-processed the Model Selection & Evaluation stages can now begin. Although not a necessary step in the development of this project the Image Segmentation stage did provide useful insight into what further developments could be made to the project if a longer time span was provided.

This will be discussed further in the final chapter but to briefly clarify on the topic, the Image Segmentation stage was undertaken as an extra step coherent to the overall goal of the project. The research discovered in this stage would help for further investigations into a type of 'automatic segmentation' which would automatically segment the tumour in the identified cancerous image. Due to time limitations, this

side objective was ultimately cut from the final iteration of the project but the research conducted on this goal has been left in this document as an indication of what was originally planned.

3 Model Selection & Model Performance

3.1 Introduction

Now that the images collected have been pre-processed, the next stage is the Model Selection & Model Performance. To predict the pre-processed images this report will focus primarily on the concept of Convolutional Neural Networks. Convolutional Neural Networks have revolutionized the objective of classifying images based on the similar features they possess. The objective of this section of the report is to demonstrate the various procedures/research which was carried out which led to the choice of the model's architecture, optimizer and value for various hyperparameters. (Liao et al. 2017)

3.2 Convolutional Neural Network

3.2.1 Background

Convolutional Neural Networks (CNN) are deep Artificial Neural Networks (ANN) who's primary goal is to classify images. In this case, MRI brain scans. To classify these images the model will process the images through several layers, an architecture which will be discussed in later sections. These various layers each have their purpose but share a similar goal of identifying features in the image. The image will be flattened and will gradually get smaller and smaller as each layer tries to find the images most prominent features.

CNN's have brought a lot of attention to the world of deep learning, attention which can be considered both good and bad. CNN's have driven deep learning into a new era allowing for major advances in the area of Computer Vision which has obvious applications such as self-driving cars, robotics, drones etc. Just like a normal CNN, these advanced operations will use real-time images with semantic image segmentation to accurately make out features which will indicate the presence of objects.

To understand CNNs they can be thought of as an advanced application of a 'normal' ANN. The idea is still based around a system of interconnected artificial neurons that exchange important information between one another, but this is only a subsection to what a CNN consists off.

3.3 Layers

CNN's consist of a multitude of different layers and parameters that can be changed to produce an entirely different accuracy or prediction. This section of the report will outline each layer a CNN consists off and the various parameters it contains.

3.3.1 Input Layer

Though not an actual layer in a CNN it is important to understand how images are read into a CNN and the best practices based on image size. As previously discussed in past sections images are represented in three colour channels, these being Red, Green and Blue or more commonly known as RGB. Computers read images in as arrays of numbers represented by the images $H \times W \times D$. An RGB image will have a *Depth* of 3 while a greyscale image will have a *Depth* of 1. Since it does not contain three colour channels but the only one which contains varying values of the colour grey.

(O'Shea and Nash 2015) The size of the input image is import when creating a CNN. Large image sizes will require larger networks and more sophisticated models which will break the images down and extract the important information they possess. Pre-existing networks such as VGG-16 are examples of large networks which can break down large images, however, the larger the image the more computation power needed to fully identify and extract the features present in the image. Smaller networks like LeNet-5 will have a much smaller input image which will require less computation power and will be substantially faster than a larger

network. This report will discuss pre-existing networks in a later section. The image being passed into the input layer should technically be recursively divisible by two. Such numbers could include 32×32 or 64×64.

3.3.2 Convolutional Layer

As the name suggests this layer of the network is off extreme importance and plays a vital role in how the network will classify the images presented. Each layer of parameters focuses primarily around the concept of learn-able kernels. These kernels are usually small in dimensionality and act the same as the filter chosen for noise reduction did in the pre-processing section. The layer will convolve each filter across the entirety of the image which will produce a 2D activation map. (Hijazi, Kumar, and Rowen 2015)

As the kernel moves across the image a new value will be produced for each product of that kernel. The value is known as the scalar product. After the kernel has glided over each pixel the output image will have shrunk. For example, consider an image size N of 32×32 and a kernel F of size 5×5 the output image size O can be calculated as follows. (Nedjah, Santos, and Macedo Mourelle 2019)

$$\begin{aligned} O &= (N - K + 1) \times (N - K + 1) \\ O &= (32 - 5 + 1) \times (32 - 5 + 1) \\ O &= 28 \times 28 \end{aligned} \tag{3}$$

Thus, the output of the image using a kernel of size 5×5 is 28. However, the Convolutional Layer contains several hyper-parameters which can drastically change the complexity of the network these hyper-parameters are Depth or also known as Feature Maps, Stride and Zero Padding. Feature maps represent the number of filters which will be produced after the kernel has created the output image. In the example above an image of 32×32 was used with a kernel of size 5×5. Now imagine a filter value F of 10 was set. The equation would not change but the output images dimensions would be altered to represent the value which was set for the filter parameter. Thus, the output now represents the height, width and depth of the image.

$$H \times W \times D = 28 \times 28 \times 10 \tag{4}$$

The filter size simply represents the number of images of size $N \times N$ will be produced after the process of convolutional. Alternatively, one could say the output of a convolutional layer with an input kernel will produce 1 output while a convolutional layer with K kernels will produce F output features. Stride is another important hyper-parameter to consider when building the model. Stride represents the distance between spatial locations where the kernel convolutional was applied. Essentially Stride is the number of pixel shifts over our input matrix. When Stride is set to a low value such as 1 the kernel will shift through image 1 pixel at a time while if Stride is set to a higher value of 3 or 4 then the kernel will now glide over the pixels in steps of 3. Consider an image N of size 7×7 and a kernel size K of 3×3 the following formula can help calculate whether or not a Stride S could fit an image N.

$$(N - K)/S + 1 \tag{5}$$

The table above indicates that as the value of Stride increases the image size will decrease. After a Stride of 3, the image is now in a decimal format indicating that the kernel will not fit the dimensions of the images hence the return of a decimal value. Naturally, a Stride of 1 is considered to be the default value for the parameter, Stride can help with large images but as demonstrated above small images will cause the kernel to convolve dimensions that do not exist within the image space. The final parameter that requires attention is Zero-padding. Zero-padding is the process of padding the border of the image that was inputted, it essentially gives the user further control over the spatial dimensionality of the image that is outputted after the process of convolutional. Consider an image N of size 7×7, a kernel size K of 3×3, a Stride S of 1 and a Zero-padding P of 1 we can use the following formula to calculate the output image after Zero-padding was applied.

$$W_2 = (W_1 - F + 2P)/S + 1$$
$$H_2 = (H_1 - F + 2P)/S + 1 \tag{6}$$

Where W indicates the width of the image and H indicates the Height of the image. Note, the addition of 1 after the variable Stride is for reasons of bias.

3.3.3 Pooling Layer

Very similar to the convolutional Layer, the Pooling Layer is responsible for reducing the spatial size of the outputted image after convolution which is also more commonly known as the Convolved Feature. The image is once again decreased in size so that less computational power will be needed to further process the data. It is also very useful for extracting dominant features which are rotational and positional invariant. When creating the model one must choose to either use Max Pooling or Average Pooling.

Max Pooling simply returns the maximum value from the portion of the image which is covered by the kernel. While Average Pooling will return the average of all the values from the portion of the image which is covered by the kernel. Max Pooling can also act as a Noise Suppressant, the process can perform de-noising along with dimensionality reduction. The above processes of convolution and pooling have enabled the created model to understand the features that exist within the image. The final output will be flattened and passed into a regular Neural Network for classification.

3.4 Existing Architecture

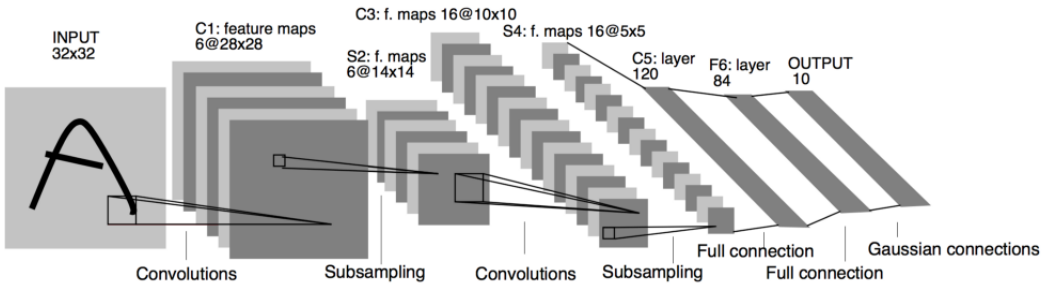
This sub-section of the report will focus on the major network architectures that exist in the field of deep learning. These networks will act as the foundation of the model which will be created to predict the presence of brain cancer from the images collected. Three important networks signify important aspects that will be used later in the creation of the model. The networks that will be discussed are:

- LeNet-5
- VGG-16
- AlexNet

3.4.1 LeNet-5

(El-Sawy, Hazem, and Loey 2016) LeNet-5 is a classical CNN which was used to classify hand-written digits on bank cheques in the United States, the process was automatic and was widely used. LeNet-5 is considered a very simple network. The network itself only contains 7 layers, 3 of which are the convolutional layers, 2 pooling layers and finally 1 fully connected layer that has an accompanying output layer. The network uses a kernel of 5×5 with a stride of 1. The sub-sampling layers or pooling layers are 2×2 average pooling layers. As for the activation functions used throughout the network, LeNet-5 uses tanh for the majority of the architecture and sigmoid for the output layer. LeNet-5 was revolutionary for its time but made interesting architectural design choices as most of these do not meet today's standards or best practices.

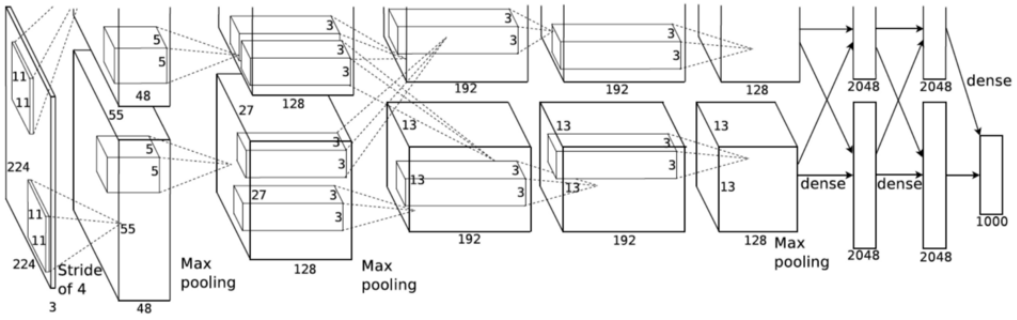
For example, in the LeNet-5 architecture, one of the networks convolutional layers does not use all of the features produced in the previous sub-sampling layer. This is highly uncommon and is rarely seen in networks produced to today's standard. In total the network LeNet-5 has 60,000 parameters.



3.4.2 AlexNet

(Canziani, Paszke, and Culurciello 2016) AlexNet can be referred to as the direct successor to the network LeNet-5. Just like the previous network both AlexNets and LeNets architecture are very similar however AlexNet is a considerably larger model. AlexNet contains 5 convolutional layers with 3 fully connected layers. AlexNet improved upon LeNets network by changing the activation functions used. Previously shown LeNet uses tanh for its activation function while AlexNet decided to use RELU the most widely used activation function to date. (Sarwar et al. 2018)

To handle the problem of over-fitting AlexNet introduced the concept of Dropout. The term Dropout can be simply understood as the dropping of units from various layers in a neural network. The role of Dropout is to force the CNN or ANN to learn more robust features from the data between fed through the network. This report will abbreviate further on the use of Dropout in later sections. In total the network AlexNet has 60 million parameters.



3.4.3 VGG-16

(Yu et al. 2016) VGG or VGGNet is a deep CNN designed for object recognition which was developed and trained by Oxfords renowned Visual Geometry Group hence the name of the network. The network achieved very good performance on the Image-Net data set. The network is very large with 13 convolutional layers which are deep stacked into groups containing 2-3 layers. After these groups, there is a pooling operation meaning the network has a total of 5 sub-sampling layers with 3 fully connected layers at the end containing over 4000 neurons each.

This network is considerably large but for good reason. The input layer will take in an image of size 224×224. The network was created for considerably large images which require more layers to break the image down so the model can learn the images features. Unlike, LeNet-5 this network uses RELU as the activation function for each convolution and fully connected layer. Finally, instead of sigmoid VGG makes use of Softmax because the network will have more than 1 output. In total the network VGG-16 has 138 million parameters.

3.5 The Model

3.5.1 Introduction

This section of the report will focus on the model that was created to classify the medical images that were collected and pre-processed. The section will focus on the architecture that was created and the design choices that were chosen in the creation of the network. The model was created based on numerous tests carried out, each time the architecture was tuned based on previous test results.

3.6 Architecture

3.6.1 Input Layer

The input layer of the network will take an image of a size of 64×64. Multiple tests were done previous to this selection of image size, the previous tests where very similar to the tests undertaken after however, the image size was much larger which proved difficult to manage. Smaller image size was then chosen, as already explained in the pre-processing section of the report the image size was perfectly chosen based on the visibility of the features shown in the image. The smaller the images got it was hard to pin-point the features in each image i.e the tumour or cancerous region. The image size 64×64 allows for the creation of larger networks without using too much computational power.

3.6.2 Convolutional Layer

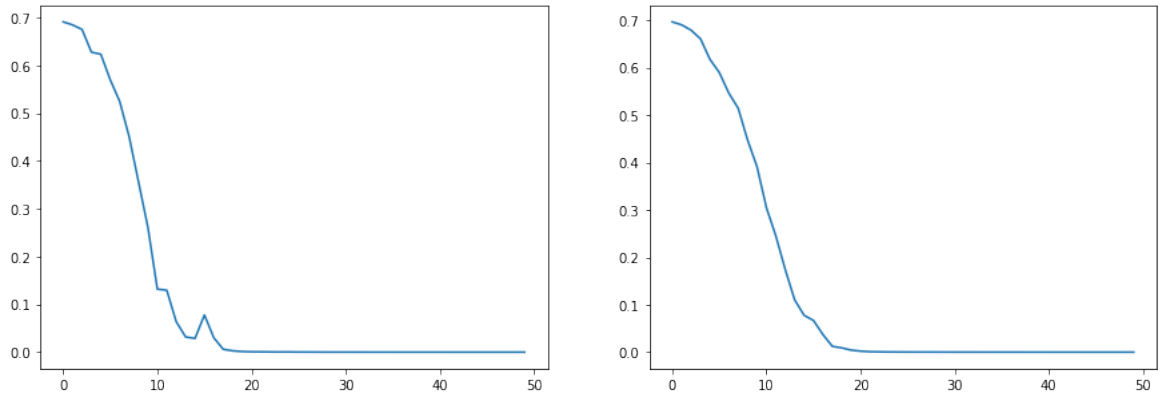
Choosing the correct amount of convolutional layers was the most difficult choice when creating this model. Numerous tests were performed before the image size of 64×64 was confirmed to be the final input image size. Once the image size was finally decided, architectures could then be decided based on the final output size achieved by using different sized kernels. The two main kernel sizes focused heavily on where sizes of 3×3 and 5×5.

These two kernel sizes produce two architectures that fit the model perfectly. The first architecture using a kernel size of 3×3 will produce an output image of 1×1 after being convolved through 4 convolutional layers. The model uses 4 sub-sampling layers and does not use deep-stacking to further decrease the size of the image. To come to this conclusion the process of testing started layer by layer. For example, 1 Convolutional Layer was tested on using the filter range 16-256 and the results were recorded, the process was repeated until the max number of convolutional layers was reached which in this was four layers. A small number of tests where preformed using deep-stacking where convolutional layers are stacked onto one another without a sub-sampling layer. The results were inconclusive and there was no increase in accuracy but a decrease in how well the model's loss converges.

The tests mainly stuck to a pattern of the filter size being recursively divisible by 2. Below is a table showing the model architecture using a kernel size of 3×3 with accompanied mathematical reasoning for the design choices. To note this network with a value of 128 neurons for its one dense layer has a total of over 120,000 parameters.

Kernel Size	Convolutional Layers	Sub-Sampling Layers	Dense Layers
3×3	4	4	3

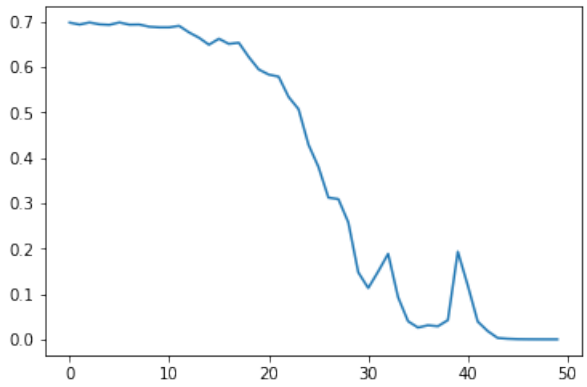
Below is the loss recorded from using four convolutional layers. The model's loss indicates that the model is converging after a certain range of epochs. As the epoch range increases, however, the model will fail to learn any more important information meaning a small epoch range of 50-100 epochs is perfect for small batch sizes such as 32 & 64 which were the primary values used to test the model at this time.



This was not the only reasoning for choosing the structure of this model. As previously discussed there was a significant amount of time spent on researching pre-existing network architectures that were used for a similar purpose to the objective of this report. The majority of these networks used either the LeNet, AlexNet or VGG-16 as their baseline model and altered the network to fit the data that they had collected. This was a technique that was also used for this project, segments of all three of the discussed classical networks where chosen. Thus, this leads to the creation of the two proposed models using two different sized kernels. To note the network shown below, with a value of 128 neurons for its one dense layer has a total of over 500,000 parameters that is 300,000 parameters less than the previous model.

Kernel Size	Convolutional Layers	Sub-Sampling Layers	Dense Layers
5×5	5	3	1

The second model designed is a significantly larger model which contains five convolutional layers, 3 sub-sampling layers and 1 output layer. Using a 5×5 kernel size results in a good overall validation accuracy, the training accuracy for both model overfits easily and should not be taken into account for the procedure for identifying the best possible model. The main difference between these models, however, is the lack of convergence in the model's loss. Using the 5×5 kernel makes the model's loss 'spike' in performance. This is evidently why a kernel size of 3×3 was used. A similar approach to the investigation of the kernel size 3×3 was undertaken to investigate the kernel size of 5×5 to ensure that a better performing model was not being missed. Once the architecture was chosen later experiments were performed on this model to increase both the accuracy and convergence time. One such experiment was the use of Dilation. Unfortunately, as the model cannot use a stride value greater than one as the images are physically too small a parameter known as Dilation was used.



The parameter Dilation allows for the use of a 5×5 kernel when in fact a kernel size of 3×3 was being used. This parameter allows one to have the benefits of using a larger kernel size when they are using a smaller kernel size. Dilating the filter means expanding its size filling the empty positions with zeros. The filter elements (the weights) are matched to distant (not adjacent) elements in the input matrix. The distance is determined by the dilation coefficient D. The images collected of brain scans have a lot of unnecessary information contained within each. As the value for stride cannot be increased Dilation is a perfect parameter to use to artificially increase the size of the kernel. Other experiments were performed using a padding on the convolutional layer. However, this was removed later on during testing as it is more common to use Zero paddings or padding in general on the accompanying sub-sampling layers.

3.6.3 Pooling Layer

The pooling layer as previously discussed acts very similarly to the convolutional layer as it will essentially shrink the image further to reveal further features that the image may possess. As discussed in the previous section two types of pooling can be performed, Max-pooling or Average-pooling. Research already carried out on the pooling layers suggests that Max-pooling is the most common and best solution to use. However, to be sure of this 2 tests were performed on model 1 (model containing a kernel size of 3×3) which were undertaken to evaluate if the type of pooling used changed would affect the overall accuracy or loss the model produces. The outcome of these tests where surprisingly inconclusive, there was no apparent change in either the loss or the accuracy when testing with both Max and Average pooling operations. These tests have to lead to the conclusion that 'model 1' will now use Max-pooling for each pooling layer as best practices and prior tests on irrelevant projects have proven that Max-pooling is the better of the two.

As previously discussed in the previous section pooling was applied to the Convolutional Layer but studies consistently show that it is a better choice to use paddings such as Zero or Valid on the sub-sampling layer. Although it didn't make a major difference in performance this choice was substantially chosen based on research in similar experiments.

3.6.4 Fully Connected Layer

Not much attention was paid to the Dense layers of either model. Previous to finalizing the model architecture several tests were performed using an increased number of dense layers with a higher number of neurons in each layer after each test was performed. These tests where again inclusive but still contained little information that gave a general overview of the of how increasing or decreasing the number of layers & neurons affected the final output of the model. Generally from the tests performed there was a significant pattern recorded in tests that contained 1 dense layer will a small number of neurons, this pattern indicated an increase in accuracy and decrease loss while the vice versa was recorded with a greater number of dense layers containing a higher amount of neurons in each layer.

The fully connected layer or Dense layer was previously considered a hyper-parameter however, this was later removed as adding more Dense Layers and more Neurons did not benefit the overall accuracy and loss.

The decision to use three dense layers came from the experiment of increasing the number of dense layers and neurons manually. Although there was not a major difference in terms of results there was a small significant increase in training time and convergence time which did benefit the final model.

3.7 Grid Search

3.7.1 Introduction

Grid Search is a form of hyper-parameter optimization where a list of hyper-parameters are defined and tested on the image data. Sklearns Grid Search uses cross-validation to test the hyper-parameters on a validation set a certain number of times based on the value set for the number of k-folds (cv). Grid Search is used to exhaust all possible parameter choices based on previous assessments in the Model Selection & Model Performance phase. There are several different choices for hyper-parameter optimization. The most common being Grid Search.

However, Random Search and Bayesian Optimization are also extremely common and are rising in popularity with other Deep Learning Artificial Intelligence projects. Random Search randomly search's for the best parameters based on a pre-set list of hyper-parameters. While Bayesian Optimization is similar to Grid Search as it will use a list of hyper-parameters to exhaust all possibilities to achieve the best type of model. Bayesian Optimization allows for stopping and essentially re-running the search in different stages in ranges for each hyper-parameter.

3.7.2 Hyper-parameters

There are several Hyper-parameters to consider when optimizing the model's performance. Throughout the creation of this model, default parameters were used to gain an insight into what results can be expected when optimizers, batch size, dense layers or the learning rate changes. Hyper-parameter tuning will automate the process of having to manually tweak each one of these variables. The process will be carried out using Grid Search. Several hyper-parameters will require this process, these parameters are:

- Optimizer
  - At the moment the optimizer being used is ADAM, ADAM is arguably the most popular type of optimizer. All tests referenced in this report were done so using the ADAM optimizer. The optimizer is a hyper-parameter but it is still important to gather a general idea of how the accuracy and loss would change with our selected model. Multiple tests were conducted with each type of optimizer available. Each optimizer showed very good performance in both accuracy and loss however the two best optimizer's recorded where Adam and Adamax because of these test these two optimizers were used for the process of grid search.
- Learning Rate
  - The leaning rate of an optimizer can control how large or small of a step it will take. This parameter essentially controls the ability of our model to converge. This parameter will be tested against a range of values to see which value most suits the model. The learning rate is an important hyper-parameter in any deep-learning project. The manual testing performed using the learning rate however did not have any beneficial effects on the model. The best-recorded learning rates were values of 0.01 and 0.001, these are the values that will be used for a grid search.
- Neurons
  - Although manual tests proved that the number of Neurons does not have a major effect on the overall outcome of the model's loss or accuracy the number of neurons was still considered a hyper-parameter because the batch size and epoch range was also being altered.
- Weights Initializer

  - The weights initializer represents the random value that will be set for the weights in the Convolutional Neural Network. Weights initializers also known as a kernel initializer can be used on either the Convolutional Layers or the Dense Layers. As previously stated they control how the weights are set in the said layer.

- Batch Size
  - Although multiple tests have already been performed on different batch sizes there are still questions to be asked on whether these changes affect the model when it has a different optimizer, learning rate, loss function, activation function or a varied amount dense layers.

- Epoch Range
  - The epoch range will control how long the model will train for. This parameter along with the batch size can control how the model will converge and what number of neurons will be needed to achieve this.

3.7.3 Adam

The first run of Grid Search was done so using the Optimizer Adam. Each test was split up based on the optimizer so both the learning rate and the decay rate could be set as a hyper-parameter. The test took a substantial amount of time to complete. The best-recorded model using ADAM was a model using 64 Neurons per layer with a learning rate of 0.001. The test suggested that a batch size of 64 and 100 epochs was the range that would be needed to achieve the best-recorded results. Also, the bets recorded initializer was suggested to 'Normal' which is no surprise since this was one of the best performing initializers during manual testing. The final recorded accuracy was 76%.

3.7.4 Adamax

The second run of Grid Search was done so using the Optimizer Adamax which is a common counterpart to Adam. The test took a substantial amount of time to complete. The best-recorded model using ADAMAX was a model using 32 Neurons per layer with a learning rate of 0.01. The test suggested that a batch size of 64 and 50 epochs was the range that would be needed to achieve the best-recorded results. Also, the bets recorded initializer was suggested to 'Normal' which is no surprise since this was one of the best performing initializers during manual testing. The final recorded accuracy was 78%.

3.7.5 Conclusion

Grid Search presented itself to be the toughest task to complete as the process of grid search is long and contains many risks. Choosing the hyper-parameters for Grid Search was a quick and painless task but executing Grid Search itself seemed to be the downfall of this project. Multiple tests were ran using a variety of values for each hyper-parameter and as each test would take days to complete the list began to get smaller and smaller as the time to work on this project is limited and there where many steps left to be completed.

To finish Grid Search Amazon Web Services were used to run the tests over several days using allocated credits. The tests were run on an EC2 instance which used a Ubuntu Amazon Machine Image (AMI) which had all the pre-required libraries installed on it. To run the code while the session was closed the python library 'tmux' was used which allowed the code to execute if the session was terminated on the local machine. This issue with Grid Search is the ultimate reason why only 2 k-folds were chosen over 3 (default) or a higher value of 5 and why the hyper-parameter list was so short.

Grid Search was an inefficient way of preforming hyper-parameter optimization which unfortunately wasted a substantial amount of time which could have been spent on more important areas of this project. Future projects should research further into both Random Search and Bayesian Optimization as these optimization techniques would be a better choice for finding the best performing model.

3.8 Generalization

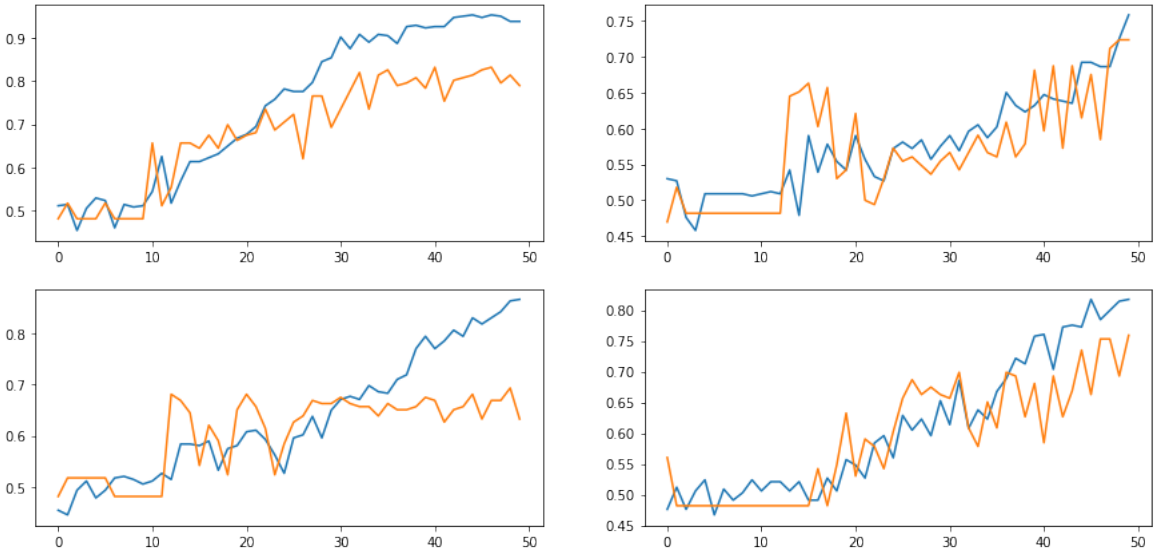
3.8.1 Introduction

It is no surprise that this model is overfitted. Since the first test run the training accuracy has had a tendency to shoot up to 100% and have its loss drop below 0.0. The data-set collected is quite small being only 500 images in size and was always likely to overfit. The ensembles of Convolutional Neural Networks and Artificial Neural Networks with different model configurations are known to reduce overfitting. However, they require the additional computational expense of training and testing various models.

There are various ways to reduce overfitting in a model or to apply generalization. The goal of this section is to take the best-recorded model that was recorded from Grid Search and apply generalization techniques to reduce the effects of overfitting. The problem of overfitting will cause the network to produce poor performance results when the model is evaluated on new unseen data. The generalization error increases due to the model overfitting.

3.8.2 Dropout

One single model can simulate having a large number of different network architectures by dropping out certain nodes during training. This process is known as Dropout and offers a very effective strategy for beating the issue of overfitting and improve the generalization error of the Convolutional Neural Network. Dropout can be used in several layers in the Convolutional Neural Networks. Most notably the Convolutional Layers & the Dense Layers. It is also very common to use Dropout as a hyper-parameter were just like in grid search values were applied to each hyper-parameter, an array of values would be assigned to a dropout parameter which would then loop through these values and find the best performing model based on the values for Dropout provided. This project, however, did not use Dropout as a hyper-parameter as a manual procedure is just as effective at discovering what values of Dropout work best which each layer in the model. A very common value for Dropout is the probability of 0.5 for retaining the output of each node in a Hidden Layer while a small value of 0.2 or 0.3 is a common occurrence in the Convolutional Layers.



3.8.3 Bias

Bias is an important generalization technique which helps tackle the issue of overfitting and help the model converge faster. Multiple tests were performed to determine the effects of using bias on each layer (Convolutional & Dense). Using Bias on each type of layer returned inconclusive results when matched to have a bias

on only the Dense or Convolutional layer. Using Bias on each layer allows for a small increase in convergence although this is not entirely noticeable as the model now considerably longer to converge since grid search was performed.

3.8.4 Image Augmentation

Image Augmentation is the process of extending the capacity of a data-set without collecting more data. Image Augmentation entails the use of sheers, flips, zooms and various other modifications that can be applied to images to make them look visually different from their original form. These operations are performed to increase the size of the data-set to decrease the issue of overfitting on the original data-set.

Image Augmentation does not require the process of collecting more image data. The operations work on the original data collected and will perform these modifications to create a new data-set with the original images with various modifications applied. This can substantially increase a data-set from hundreds of images to thousands. This is a very effective technique to increase the data-set and decrease the problem of over-fitting for the data-set used in this project. As the data-set being used is substantially small in size with 250 Healthy images and 250 Un-Healthy this can help increase each category by a pre-determined set number of modifications. Two data-sets were created using the python library 'Augmentor' which makes augmenting the images a quick and easy task.

There were several modifications used to augment the data. These modifications allowed for a great variety of images to be produced without having to worry about duplicates within the data-set. The modifications used are as follows:

- Sheer: Sheers allow for the image to rotate inwards at a specified rotation rate.
- Rotation: Rotation rotates the overall image on a specified rotation axis. Meaning once the Augmentor begins it will rotate the image by the value specified by the user. In this case, the value specified was 90-180 degrees.
- Zoom: Zoom will zoom the image in at various ranges which are specified as a parameter.

3.9 Prediction

3.9.1 Making the Prediction

The final step of this project was to perform a prediction on unseen data. The un-seen data was collected from a medical journal website by saving the interactive scan as a JPG image. This website has a collection of solved and unsolved cases regarding medical operations and patient studies. Each case has a collection of images that are displayed in an interactive environment where the user can use a slider to get a different perspective of the image. Each perspective of the image can be considered a new individual image and works perfectly as the un-seen data that the model will attempt to predict.

To predict an image, the images must once again be pre-processed. A similar approach that was undertaken during the Pre-processing stage will also be used here. However, the majority of the steps undertaken in the pre-processing are unnecessary and can be disregarded. There were several issues related to the prediction of unseen data. For example, loading PNG images and using the model to predict the category unfortunately only predicts one class of images. To understand this issue existing data was used to monitor if the model can predict data that the model has already learned. Using PNG images the model still did not identify the second category of data, the main category being predicted for every image was Un-Healthy.

To ensure this issue was not based on the file extension these images were converted to the originally used JPG format. Once the files were converted the issue with predicting images was no longer evident as the model was successfully predicting both categories of image. However, the model is not predicting every type of image this may be down to the issue that the model is still over-fitting or that the data is just too random for this type of image classification. Overall the model done very well and predicted 8 out of the 10 images shown to the model. However, if more time had been allocated to this section more research could have been conducted to ensure that the model was predicting correctly.

3.10 Final Recommendations & Conclusions

3.10.1 Recommendations

This type of image classification experiment was undertaken to successfully predict the presence of brain cancer. The project itself was undertaken in 6 months. The scope of this project was not fully completed in this set time span and the model predicted with average performance on unseen data. Too much time was wasted on less important stages such as the pre-processing stage which included the side goal of image segmentation. If the side goal of image segmentation was never suggested and sought after, more time could have then been allocated to the stages which needed the most time E.G Grid Search.

A deadline increase would have allowed for more research to be conducted on certain stages in this project and have allowed for more attention to be paid to the more important stages in this project. If further research was to be conducted in this area one would plan and prioritize the process of Model Selection & Model Performance as this stage is crucial, Grid Search should be performed as early as possible and more attention should be made to what environment the test is performed in. As it is very computational heavy it can be very easy to break while running. This was an unfortunately common issue for this project and caused the decrement of hyper-parameter range.

3.10.2 Conclusions

This project was very hard to conduct as there was a lack of data from the beginning. The pre-processing stage was interesting and allowed for important research to be conducted on the image segmentation techniques. However, the Model Selection & Model Performance stage was the most crucial to this project. The project struggled through this stage as the model was overfitting very early on from the beginning. Grid Search was ultimately the this projects downfall as the project struggled through this stage.

As each run of grid search took longer than anticipated the hyper-parameter list shrunk substantially to acquire results for the next stage. Generalization allowed for tackling the issue of overfitting which was a major issue throughout this project. However, using image augmentation allowed the model to converge as well as fix this over the fitting issue to receive the final result of the prediction of brain cancer.

References

Canziani, Alfredo, Adam Paszke, and Eugenio Culurciello (2016). "An analysis of deep neural network models for practical applications". In: *arXiv preprint arXiv:1605.07678*.

Hijazi, Samer, Rishi Kumar, and Chris Rowen (2015). "Using convolutional neural networks for image recognition". In: *Cadence Design Systems Inc.: San Jose, CA, USA*.

Liao, Lixin et al. (2017). "Finding the Secret of CNN Parameter Layout under Strict Size Constraint". In: *Proceedings of the 25th ACM international conference on Multimedia*. ACM, pp. 997–1005.

Nedjah, Nadia, Igor Santos, and Luiza de Macedo Mourelle (2019). "Sentiment analysis using convolutional neural network via word embeddings". In: *Evolutionary Intelligence*, pp. 1–25.

O'Shea, Keiron and Ryan Nash (2015). "An introduction to convolutional neural networks". In: *arXiv preprint arXiv:1511.08458*.

Sarwar, Syed Shakib et al. (2018). "Energy efficient neural computing: A study of cross-layer approximations". In: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 8.4, pp. 796–809.

El-Sawy, Ahmed, EL-Bakry Hazem, and Mohamed Loey (2016). "CNN for handwritten arabic digits recognition based on LeNet-5". In: *International Conference on Advanced Intelligent Systems and Informatics*. Springer, pp. 566–575.

Ségonne, Florent et al. (2004). "A hybrid approach to the skull stripping problem in MRI". In: *Neuroimage* 22.3, pp. 1060–1075.

Yu, Wei et al. (2016). "Visualizing and comparing AlexNet and VGG using deconvolutional layers". In: *Proceedings of the 33 rd International Conference on Machine Learning*.