DEEP LEARNING BASED APPROACHES APPLIED TO PATIENT'S ULTRASOUNDS WITH THE AIM TO DIAGNOSE THE DISEASE MYOSITIS

A DISSERTATION SUBMITTED TO MANCHESTER METROPOLITAN UNIVERSITY
FOR THE DEGREE OF MASTER OF SCIENCE
IN THE FACULTY OF SCIENCE AND ENGINEERING

2019

By

[redacted]

Department of Computing and Mathematics

Contents

List of Tables

List of Figures

Abstract

Medical images is a key tool used in the diagnosis of many diseases and illnesses. Myositis is usually diagnosed using MRI scans, however, this option is not available to some patients. Ultrasound imaging offers an alternative to MRI when this is the case. However, potentially problems can arise when using this method. Convolutional neural networks can be used to help mitigate these problems and help save time and ultimately money for health services such as the NHS.

Convolutional neural networks have been dominating the field in the classification of images. However, to achieve state of the art results there needs to be a large dataset. To help relieve this problem, transfer learning can be employed. This study will focus on three different pretrained models.

The three different models that were tested, along with a variety of augmentation techniques, were VGG16, MobileNet and Inception-V3. MobileNet and Inception-V3 both performed similarly using different augmentations. MobileNet achieved an accuracy of 81.5% using random zooming between 0.7 and 1.3. Inception achieved an accuracy of 81.3% using random rotations up to 30°. VGG16 was the least successful with an accuracy of 72.9% by rescaling the pixels between 0 and 1.

Declaration

No part of this project has been submitted in support of an application for any other degree or qualification at this or any other institute of learning. Apart from those parts of the project containing citations to the work of others, this project is my own unaided work.

Signed: [redacted]

Date: 27/09/2019

Copyright

Acknowledgements

I would like to thank Dr. Ryan Cunningham and Luciano Gerber for their support and guidance in this thesis.

I would like to thank my partner, [redacted] and my family for their continued support.

I would also like to dedicate this to my grandmother, [redacted], who passed away.

Abbreviations

CNN	Convolutional Neural Network
IBM	Inclusion Body Myositis
ReLU	Rectified Linear Unit
DR	Digital Radiography
CAD	Computer-Aided Diagnosis
FASP	Fetal Abdominal Standard Planes
CAKUT	Congenital Abnormalities of the Kidney Urinary-Track
RAFDB	Real-World Affective Faces Database
SFEW	Statistic Facial Expression in the Wild
GAN	Generative Adversarial Network
IVUS	Intravascular Ultrasound

Chapter 1

Introduction

Image classification is a key technique that can be identified throughout modern technology fields, such as the medical and security industry. Today, it is becoming widely accepted that image classification can be especially useful in the medical field as it has the potential to provide fast, high accuracy results which are essential to help reduce cost and relieve health service waiting times, such as in the National Health Service in the United Kingdom.

Imaging is currently used in the diagnosis of many diseases that affect humans today. Images provide important information to clinicians, such as the severity and condition of the disease, however, the assessment of the images is time consuming, particularly in rare and uncommon diseases. This study will focus on the disease myositis, a rare condition that causes inflammation of the muscles which can lead to pain and loss of muscle strength for the affected person. Myositis is the name given to a group of inflammatory myopathies and there are four inflammatory muscle diseases that fall under this broad category. They are; dermatomyositis, juvenile dermatomyositis, polymyositis and inclusion body myositis (IBM).

In general, an MRI scan would be used to scan the muscles, however, this method is both expensive and time consuming. Furthermore, this method excludes patients who are fitted with implants and pacemakers as this method of scanning can interfere with the aforementioned devices (Burlina et al. (2017)), therefore, another method must be employed. When this is the case ultrasound imaging offers an alternative.

Ultrasound imaging has its own set of challenges that make it unsuitable for widespread

use. It is subject to operator and interpreter bias, and often there are echo-intensity changes due to the lack of standardisation in equipment across clinics and hospitals. Therefore, it is difficult and ambitious to compare the results from different systems (Burlina et al. (2017)).

The data used was recorded from the Johns Hopkins Myositis Clinic in Baltimore, Maryland. The images were acquired using a GE Logiq E System with a 12 MHz linear array transducer. The imaging parameters were kept constant. These were as follows: frequency at 10MHz, gain at 40, dynamic range at 87 with cross beam and other enhancers turned off. The depth was 4cm for each muscle except for the rectus femoris which was set to 6cm. Seven muscles groups were imaged bilaterally per subject (Burlina et al. (2017)). The dataset has been made publicly available at `https://github.com/jalbayd1/myopathy_US`.

Burlina et al. (2017) explored the use of deep learning methods with the aforementioned myositis dataset. Three different datasets were created from the original dataset, thus, each dataset was given its own classification problem to address. These were as follows: normal vs myositis, normal vs IBM patients and IBM vs other types of myositis. The AlexNet model was used for classification and was pretrained on the imagenet dataset.

The objectives for this study were as follows:

- Create models that perform binary classification of healthy patients vs patients with myositis.
- Train several networks from scratch and analyse the results.
- Train three networks using transfer learning from three different models. (VGG16, Inception-V3 and MobileNet).
- Use data augmentation to enhance the dataset and vary parameters to see if there is a change in accuracy/F1 measure.
- Explore the use of generative adversarial networks (GANs).

Chapter 2

Background

2.1 Convolutional Neural Networks (CNNs)

In general, CNNs are substantially more successful than the traditional classic neural network when using images. One reason for this is because CNNs take advantage of local spatial coherence. This means that convolutions can be used on groups of pixels as a result of adjacent pixels normally being meaningful. In contrast, when using a classic neural network it does not draw on the meaningful information between the vectors. Therefore, CNNs are naturally faster and more powerful for the purpose of image classification.

A typical simple CNN is composed of three different layers; a convolutional layer, a pooling layer and a fully connected layer. When using these layers, the CNN learns representative and discrete hierarchical features from the data (Duan et al. (2018)). Other layers can also be employed to improve the performance of the CNN, for example, a dropout layer. A dropout layer removes a given proportion of the neurons from the network, thus, slowing down the rate of learning to allow the network to be more effective when generalising as well as reducing overfitting.

2.1.1 Convolutional Layer

The convolutional layer is the most important layer in a CNN. A kernel, a matrix which contains weights, passes along the image and each pixel is multiplied by its corresponding weight within the kernel. These are then summed up to provide a numeral. Figure 2.1 shows a kernel performing the first operation on the image. When the image

is an RGB image this process would be repeated three times as a result of of the the image being three dimensional. Every layer creates several feature maps for the image. Each layer in the network also consists of a number of set filters and a set kernel size. The number of filters set establishes how many times the kernel will pass along the image. This results in more feature maps being created. The size that is set for the kernel establishes how many pixels will be evaluated, this has a direct impact on the feature map. Common kernel sizes are 1x1, 3x3 and 5x5.

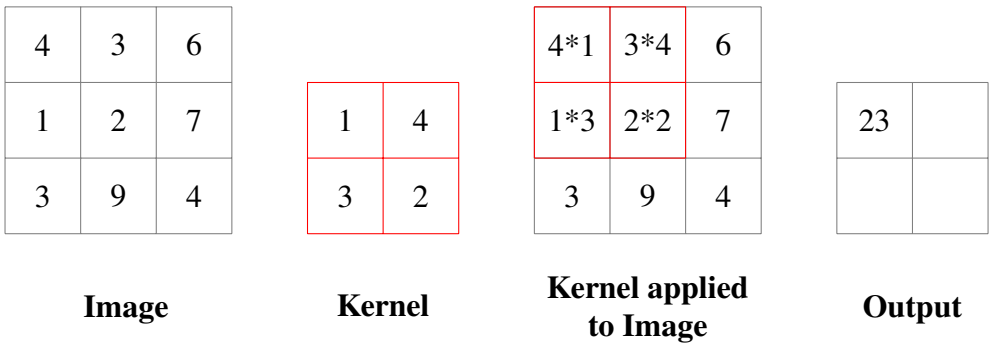


Figure 2.1: Kernel passing over an image producing an output

In some cases, the feature maps that are created by the convolutional layer are then used to create more complex features maps if that is the prerogative of the developer. An example of this would be when a convolutional layer is applied to the image of a cat. Different feature maps would be created, for example, the head, body and legs. At this stage we could then apply another convolutional layer. The feature map of the head may then be broken down again into more detail resulting an a feature map of the ears, whiskers and eyes.

Finding the best values for the number of filters and the kernel size can be challenging. It requires the testing and monitoring of both accuracy and loss within the models produced.

Downsampling is another technique that, on occasion, is applied to the convolutional layer. The result of applying this technique to the convolutional layer is that the images are smaller and therefore use less of the precious memory resources available. This is

usually applied to images that are very large or when the network itself is very large. The technique creates a lower resolution image that focuses on the important areas within an image, such as the shape, but it does not include any fine detail. Downsampling is achieved by adjusting a parameter to add strides to the layer. A stride is the number of pixels the kernel will move, therefore, by assigning a higher number to the stride, the outputting image will be smaller.

2.1.2 Pooling Layer

The pooling layer is a layer which aims to reduce the spatial dimensions of the image. It does this by using an operation and a stride. Operations that are commonly employed are maximum pooling and average pooling. Similarly to the convolutional layer, the pooling layer makes use of a kernel, however, in this layer the kernel selects the pixels to be used and performs the chosen operation. For example, if the maximum pooling method is used, the largest pixel of the kernel is chosen. The result of this would therefore be a newly generated image in which the rows and columns contain the values obtained from the chosen operation. This would result in a summarised version of the features detected by the convolutional layer. This is another form of downsampling, which reduces the spacial dimensions of the image, but provides a summarised version of the image.

2.1.3 Fully Connected Layer

A fully connected layer is a layer of neurons where each neuron is connected to every output from the previous layer. The fully connected layer, also know as a dense layer, implements the operation shown in equation 2.1. It uses an element-wise activation function. An example of this would be the activation function, rectified linear unit, this will be discussed later. The kernel is a weights matrix and the bias is a vector which are both generated by the layer. Unlike the convolutional layer, spatial information is not retained (Guo et al. (2017)). If the purpose of the CNN is classification, then the last layer in a network would be a fully connected layer where the number of neurons matches the number of classes to be classified. Alternatively, if there are only two classes to be classified only one neuron may be required as the binary crossentropy loss function could be employed.

Operation used in a fully connected layer is as follows;

$$output = activation(dot(input, kernel) + bias). \tag{2.1}$$

2.2 Activation functions

An activation function takes the results from one layer and creates an output for the next layer within the network. For example, an activation function would create an output from the the results from the convolutional layer and the pooling layer would then use this as an input. Therefore, the activation function determines which neurons are to be fired in the next layer. It is a non-linear transformation that takes place on the input and sends the transformed output to the next layer as an input.

In CNNs different activation functions are used throughout the network. These are some common activation functions used:

2.2.1 Rectified Linear Unit (ReLU)

The activation function, ReLU, converts all negative values to zero and all positive values become linear. If the input of the ReLu contains any negative values then it will not be activated. Equation 2.2 shows the formula for the ReLU activation function and figure 2.2 is the graph of the function.

ReLU activation function is defined as

$$y = max(0, x). \tag{2.2}$$

2.2.2 Sigmoid

When using the Sigmoid activation function a number between the values of zero and one is created. Therefore, Sigmoid is especially useful when predicting the probability of a value between zero and one. Therefore, because the result is differentiable, the slope of the Sigmoid curve can be found at any two points. The Sigmoid activation function is generally used when performing binary classification. Equation 2.3 shows the formula for the Sigmoid activation function and figure 2.3 shows the graph for the function.

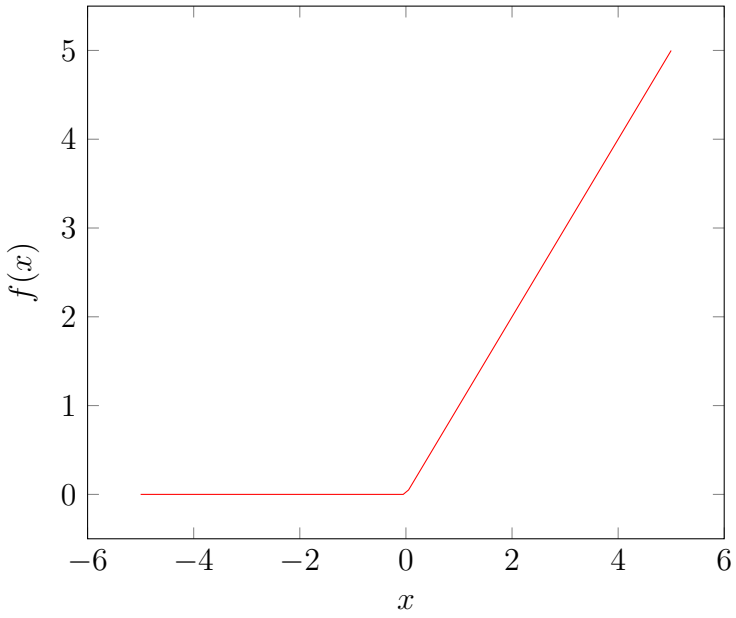


Figure 2.2: ReLU Activation Function

Sigmoid activation function is defined as

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.3}$$

2.2.3 Softmax

The Softmax activation function is a generalised logistic activation function which is used for multiclass classification. This is achieved by turning arbitrary values into probabilities meaning that the higher the value, the higher the probability. Equation 2.4 shows the formula for the Softmax activation function. X is a vector of inputs. Therefore, if there are five neurons in the output layer then there will be five elements in the vector. i is the index of the vector of inputs.

Softmax activation function is defined as

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}} \tag{2.4}$$

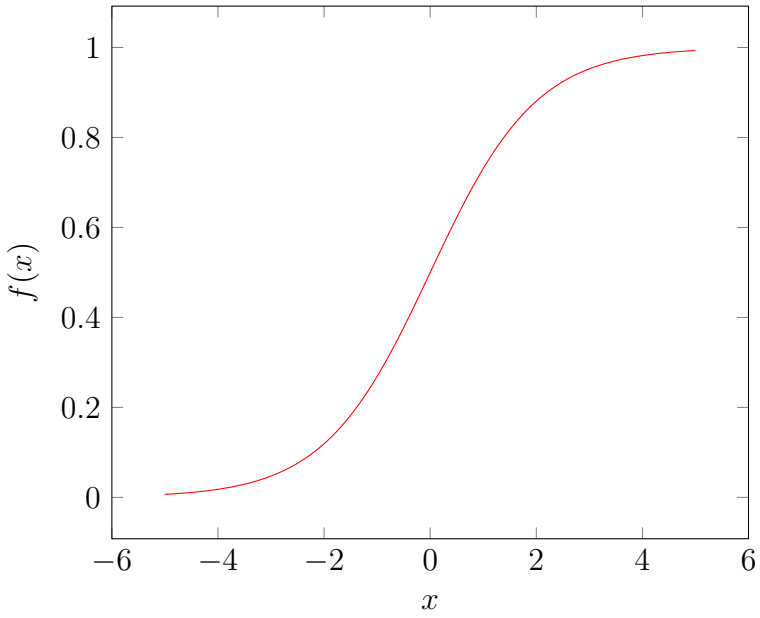


Figure 2.3: Sigmoid Activation Function

2.3 Loss Functions

During training of the network, if the aim of the network is classification, then the network would strive to attain the best weights possible, and this is evaluated by the lowest loss. The loss is calculated using a loss function by matching the real value with the predicted value.

There are a variety of loss functions that can be used. Some common ones are binary crossentropy and categorical crossentropy.

Binary crossentropy would be used if the data trying to be classified contains two classes. The output value created would be passed through a Sigmoid activation function. Only one output node is needed in the final layer of the network. If the value created is less than 0.5, then it would classify the input image as class 0. If it is greater than 0.5 then it would be classified as class 1.

Categorical crossentropy would be used if the data trying to be classified contains multiple classes. The output value created would be passed through a Softmax activation function. The last layer in the network must have the same amount of nodes as classes. These would then create a probability for each class. For example, if there are

four classes to be predicted then the output may be: class 0 = 0.7, class 1 = 0.15, class 2 = 0.1 and class 3 = 0.05. It is important to note that these probabilities will always add up to one.

2.4 Optimizers

As has previously been discussed, it is vital that the best weights are found for the network. These are decided by the network as it analyses the lowest loss value to give an accurate prediction.

A common optimizer, which is the foundation of a variety of optimizers, is the gradient descent optimizer. It works by calculating the affect a change in weight would have on the loss function. This then adjusts the weights based on the gradient. The main aim of the optimizer is to find the absolute minimum point. This is achieved by using a learning rate. The product of the learning rate and gradient ensures that the change is very small.

If the change is too big, the network will not converge at the absolute minima. A problem with using too small of a learning rate is that it may find a local minima but never find the absolute minima. Figure 2.4 shows the point where it has found the local minima but as a result of the learning rate being too small, the network will never find the absolute minima.

2.5 Improvement of Image classification over Traditional Machine Learning Techniques

CNNs outperform traditional machine learning techniques such as random forests (Burlina et al. (2017)). Usually, traditional machine learning techniques use feature engineering to find figures such as averages and standard deviations of the data. However, the advantage of using CNNs is that feature engineering does not need to take place. Another advantage is that the the weights are found by the network itself rather

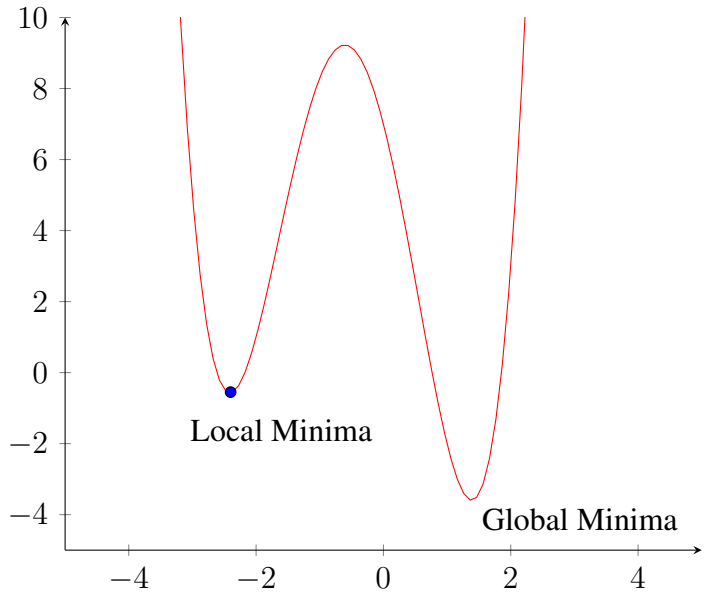


Figure 2.4: Minima points

than the developer changing the weights. This allows for potentially better results because computers are generally better at choosing the the best weights for a model.

Burlina et al. (2017) compared the use of CNNs with random forests when classifying ultrasound images. They concluded that CNNs performed better than random forests in every problem that they tested (Burlina et al. (2017)). Furthermore, before the random forest approach can be used, a clinician must perform a manual muscle delineation to create usable features such as muscle strength. Implementing this approach in a real scenario would therefore only create a semi-automated method whereas, a CNN can use an entire image, meaning that it would be fully automated. Furthermore, as the CNN only uses a single image, it allows for a much simpler approach when collecting data as the clinician only needs to perform the scan unlike in the collection of data for random forests.

2.6 Transfer Learning

A problem with CNNs is that to achieve a high accuracy, the dataset has to be large and the network needs to be trained for a significant amount of time. Transfer learning can help mitigate this problem by using information that has been learnt from a previous model and then apply it to its current problem. This enables the use of a smaller

dataset to be trained and tested and achieve a higher accuracy than a standalone model (Temdee & Uttama (2017)).

The goal of transfer learning is to use the knowledge learned from one environment to help the learning tasks in another environment. The transfer learning based on CNNs means that the trained CNN model is retrained on the data set of a new target task. This step is often referred to as network fine-tuning (Wang et al. (2018)).

2.6.1 VGG16

VGG16 was created to help further increase the success that CNNs have been having. The focus was to improve the depth of the network by both, adding more convolutional layers and using smaller kernels. The model was trained using the ImageNet dataset and the images were pre-processed by subtracting the mean RGB value from each pixel. The input to the network was a 224x224 RGB image. The structure of the VGG16 model is showing in table 2.1. The structure is very simple using only convolutional, pooling and fully connected layers.

Simonyan & Zisserman (2015) found that other networks were also using a kernel size of 7x7, however, it was found that by using three convolutional layers of a kernel size 3x3, it achieved the same effective receptive field. By using this method, they found that the number of parameters used is significantly lower. If the the input and output to the three convolutional layers had 3 channels, then it would be parameterised by *C* channels. It would the have *3(*3^2C^2*)* = 27^2 and a convolutional layer of size 7, then it would be parameterised by $7^2C^2 = 49C^2$. So by having three convolutional layers using a kernel size 3 would not only have the same effective receptive field but also uses significantly less parameters. In total the VGG16 network uses *138,357,544* parameters.

The images used in the training was randomly cropped from rescaled images and underwent some augmentation such as random horizontal flipping and random RGB colour shifting.

Simonyan & Zisserman (2015) also developed an even deeper model, VGG19 which utilises an extra three convolutional layers to create an even deeper model.

Type	Patch Size/Stride	Input Size
Conv	3x3/1	224x224x3
Conv	3x3/1	224x224x64
Max Pool	2x2/2	224x224x64
Conv	3x3/1	112x112x64
Conv	3x3/1	112x112x128
Max Pool	2x2/2	112x112x128
Conv	3x3/1	56x56x128
Conv	3x3/1	56x56x256
Conv	3x3/1	56x56x256
Max Pool	2x2/2	56x56x256
Conv	3x3/1	28x28x256
Conv	3x3/1	28x28x512
Conv	3x3/1	28x28x512
Max Pool	2x2/2	28x28x512
Conv	3x3/1	14x14x512
Conv	3x3/1	14x14x512
Conv	3x3/1	14x14x512
Max Pool	2x2/2	14x14x512
Flatten	-	25088
FC	-	4096
FC	-	4096
Softmax	Classifier	1000

Table 2.1: Structure of VGG16 sourced from Simonyan & Zisserman (2015)

These findings led Simonyan & Zisserman (2015) team to win the ImageNet challenge 2014 in 1^{st} and 2^{nd} place in the localisation and classification tracks.

2.6.2 MobileNet

The trend for CNNs have been to make them deeper and more complicated as shown with the VGG architecture. A problem created by this is that there was no focus to make CNNs more efficient with respect to size and speed (Howard et al. (2017)). By making these types of networks it would potentially allow for use with self driving car and augmented reality due to the need to be able to carry out computations with a limited platform (Howard et al. (2017)).

MobileNet allows for the developers to create a network to match the requirements

of their system by adjusting two parameters: a width parameter α and resolution multiplier ρ. Furthermore, MobileNet uses depthwise separable convolutions. Depthwise seperable convolutions splits into two layers: a layer for filtering and a layer for combining. These work by applying a kernel to the input channels following with a *1 x 1* convolution based on the number of output channels. For example, if the number if the number of input channels was 8 and output was 16, a kernel for instance, *3 x 3*, would pass over each of the input channels creating 8 feature maps. Then these feature maps are passed over by a *1 x 1* kernel 16 times. Then these are added together resulting in *8x3x3 + 8x16x1x1*. By using depthwise seperable convolutions, roughly one-eighth of the computations would be needed compared to a standard convolution (Chen & Su (2018)).

Table 2.2 shows the structure of MobileNet.

When using the MobileNet architecture, there are two parameters which can be used to adjust the size of the network. These are α and ρ. α adjusts the width of the network. If the value is less than one, then the number of filters in each layer is proportionally decreased. If it is larger than one, then it is proportionally increased. If the value is one, the default number of filters are used. ρ controls the resolution multiplier, this is applied to the input image meaning that the internal representation of every layer in the network is subsequently reduced by the same multiplier. Reducing both of these values will give a smaller network with a reduced computation but again it comes with a trade off in accuracy.

MobileNet has 4,253,864 parameters and has over 32x less parameters than the VGG16 model.

2.6.3 InceptionV3

The Inception architecture was originally designed so that it could perform well under strict constraints of memory and computational budget (Szegedy et al. (2016)). By creating such a network, it would mean that it would be possible to use the Inception architectures in big-data scenarios.

A problem that was found with the older architectures was that due to how it was

Type / Stride	Filter Shape	Input size
Conv / s2	3x3x3x32	224x224x3
Conv dw / s1	3x3x32 dw	112x112x32
Conv / s1	1x1x32x64	112x112x32
Conv dw / s2	3x3x64 dw	112x112x64
Conv / s1	1x1x64x128	56x56x64
Conv dw / s1	3x3x128 dw	56x56x128
Conv / s1	1x1x128x128	56x56x128
Conv dw / s2	3x3x128 dw	56x56x128
Conv / s1	1x1x128x256	28x28x128
Conv dw / s1	3x3x256 dw	28x28x256
Conv / s1	1x1x256x256	28x28x256
Conv dw / s2	3x3x256 dw	28x28x256
Conv / s1	1x1x256x512	14x14x256
5x Conv dw / s1 Conv / s1	3x3x512 dw 1x1x512x512	14x14x512 14x14x512
Conv dw / s2	3x3x512 dw	14x14x512
Conv / s1	1x1x512x1024	7x7x512
Conv dw / s2	3x3x1024 dw	7x7x1024
Conv / s1	1x1x1024x1024	7x7x1024
Avg Pool / s1	Pool 7x7	7x7x1024
FC / s1	1024 x 1000	1x1x1024
Softmax / s1	Classifier	1x1x1000

Table 2.2: Structure of MobileNet sourced from Howard et al. (2017)

designed, it is difficult to adapt the network so that it can be used in new cases whilst maintaining its efficiency. Szegedy et al. (2016) found that if the capacity of the Inception model needs to be increased, such as a simple change such as doubling the number of the filter sizes could result in the increase of computational cost and the number of parameters by four times.

To tackle this, Szegedy et al. (2016) hypothesised some general design principles to improve the Inception architecture. These are:

1. Avoid representational bottlenecks, especially early on in the network.

2. Increase the number of activations per tile to allow for more disentangled features. This should result in the network to train faster.

3. Use spatial aggregation to lower the dimensional embeddings. For example,

before the use of a 3x3 convolution, reduce the dimension of the input representation before the spatial aggregation.

4. Balance the width and depth of the network. By balancing the network, optimal performance should be achieved. These ideas have to be used judiciously in ambiguous situations only as it is not straight forward to use them to improve the quality of the network.

Table 2.3 shows the structure of the Inception-V3 model. In the model there are three different 'Inception' modules in which each module performs a series of different operations in which an input is sent to the module and a series of layers are used in parallel which are then concatenated together to produce an output.

Type	**Patch Size/Stride** or remarks	**Input Size**
Conv	3x3/2	299x299x3
Conv	3x3/1	149x149x32
Conv padded	3x3/1	147x147x32
pool	3x3/2	147x147x64
conv	3x3/1	73x73x64
conv	3x3/2	71x71x80
conv	3x3/1	35x35x192
3xInception	As in figure 2.5	35x35x288
5xInception	As in figure 2.6	17x17x768
2xInception	As in figure 2.7	8x8x1280
pool	8x8	8x8x2048
linear	logits	1x1x2048
softmax	classifier	1x1x1000

Table 2.3: Structure of Inception-V3 sourced from Szegedy et al. (2016)

Figures 2.5, 2.6, 2.7 shows the different modules that are used in the Inception-V3 model. Inception module 1 shown in figure 2.5 follows a similar design to which the VGG architecture followed. The inception module used in the previous model used a *5 x 5* kernel and in this version three *3 x 3* kernels. The second Inception module in figure 2.6, replaces previously used *7 x 7* kernels with *1 x n* and *n x 1*. In Szegedy et al. (2016) testing of the Inception-V3 model, n = 7 was used resulting in *1 x 7* and *7 x 1* kernels. The third module shown in figure 2.7 was created to promote high dimensional representations in the image.

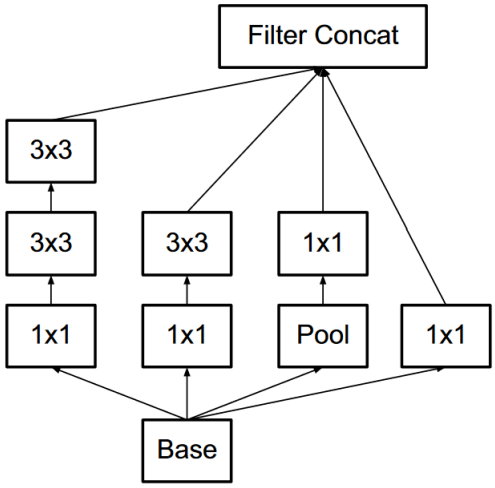


Figure 2.5: Inception Module 1 sourced from Szegedy et al. (2016)

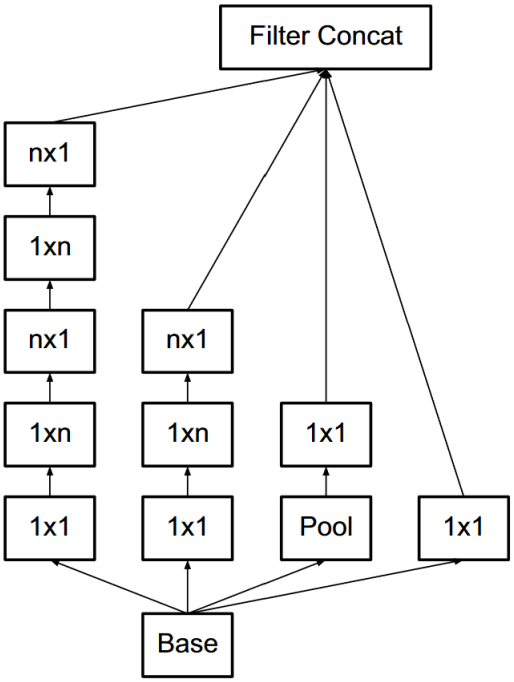


Figure 2.6: Inception Module 2 sourced from Szegedy et al. (2016)

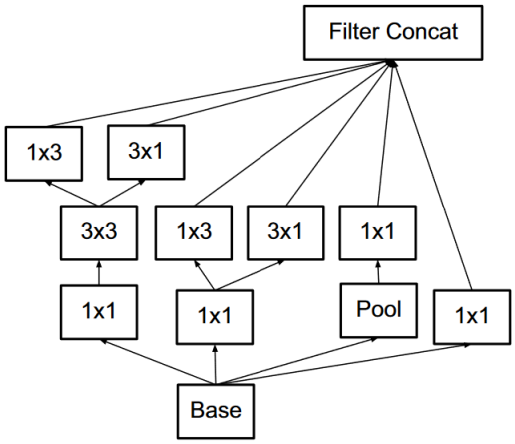


Figure 2.7: Inception Module 3 sourced from Szegedy et al. (2016)

Overall, the Inception-V3 architecture has a total of 23,851,784 parameters. Compared to VGG16 it has nearly 6x less parameters allowing it to be used on a much wider variety of systems.

2.6.4 Uses of Transfer Learning with Medical Images

There have been many studies which have researched the use of transfer learning in the field of medical imaging and the studies have achieved good results.

Transfer Learning with Digital Radiography (DR) Images

DR is challenging to classify compared to other medical images (Alsabahi et al. (2018)). There have been many attempts to solve this issue, however, this research has not performed well. Alsabahi et al. (2018) proposed the use of transfer learning using the Inception-V3 model that has been trained on the ImageNet dataset. Following this they proposed the network would be fine tuned and trained so that it could utilised for the DR dataset.

A DR image was chosen for the dataset because it is the most recommended type of 2-dimensional data for detecting lung cancer. The images used were resized and a segmentation of the image took place so that a binary image was created. This enabled the lung fields to be clearly seen. The use of the Inception-V3 model with the DR

dataset achieved a training accuracy of 83.39% and a validation accuracy of 79.10% (Alsabahi et al. (2018)).

Transfer Learning with Mammograms

Yemini et al. (2018) addresses the problem of mass detection for breast cancer in mammograms. Computer-aided diagnosis (CAD) schemes have the potential for improving breast cancer diagnosis performance. A CNN was proposed using transfer learning from the Inception-V3 architecture. However, it was found that the dataset available is too small for the use of Inception-V3 and that there was a class imbalance, where there was more negative patients than positive patients.

To solve this, artificial images were created from the negative images with a tumour placed randomly on the breast. Furthermore, a decaying weighting matrix multiplied by a sigmoid function as used so that there was a smooth transition between the background and the placed tumour. Another method used was to augment the images to balance the amount of normal and malignant images when training. Using the Inception-V3 architecture with artificially inserted tumours achieved an AUC of 0.78 and using augmentation with the Inception-V3 architecture achieved an AUC of 0.86 (Yemini et al. (2018)).

Transfer Learning with Ultrasound Videos

Chen et al. (2015) investigated the use of transfer learning for developing automatic locating of fetal abdominal standard planes (FASP) from ultrasound videos. To help improve the performance of their CNN, the use of transfer learning was investigated. Initially, a multi-layered base CNN was trained from the ImageNet dataset. Then they finetuned and trained additional fully connected layers that have been added so that it can be used with the FASP dataset. This was so that every frame in the video could be identified as either FASP or non-FASP. Chen et al. (2015) achieved high accuracy, precision, recall and F1 results of 0.904, 0.908, 0.995 and 0.950 respectively.

A problem that was found however, is that I takes approximately one minute for the system to locate the FASP from one video. Therefore, it is not real time. It was also discussed that the dataset could be improved by randomly corrupting the training data with acoustic shadows, thus potentially improving the robustness of the CNN (Chen et al. (2015)).

Transfer Learning with Ultrasound Images

The classification of ultrasound images for the diagnosis of congenital abnormalities of the kidney urinary-track (CAKUT) in children is a challenging task. Zheng et al. (2018) explored the use of transfer learning to investigate if this problem could be solved. The model chosen was the imagenet-caffe-alex model. Before any training took place, segmentation and normalisation techniques were used on the images. A graph-cut method was used to segment the kidneys. An average Dice index of 0.96 was achieved when compared to manual segmentation (Zheng et al. (2018)).

Three models were trained to compare three different datasets. The datasets were left kidneys, right kidneys and both kidneys. These were trained and compared. It was found that the model using the dataset containing the left kidneys achieved an accuracy of 0.84 ± 1.6, the right kidneys achieved an accuracy of 0.81 ± 1.8 and using both kidneys achieved an accuracy of 0.87 ± 2.1 (Zheng et al. (2018)).

2.7 Data Augmentation

A problem which many neural networks face is that if there is not a huge amount of data available for training a CNN. The CNN can struggle to learn from the data as it can easily cause over fitting and not generalise well. To attempt to tackle this data augmentation can be used. Keras provides a function 'Image Generator' which allows for images to be taken from the directory and be augmented so that a new randomly generated image is used every epoch. During this investigation there will be changes made do the generator using different techniques to change the image.

The new images are created from artificial data points being added to the original image by applying different transformations. Transformation techniques that will be investigated are: horizontal flipping, rotation, brightness range, re-scaling and the zoom range.

2.7.1 Image Augmentation in Keras

Horizontal Flipping

Horizontal flipping is a simple translation that will create a new image where the image is flipped so that it would look like the ultrasound image has been taken from another

perspective. Vertical flipping however, will not be used as the structure of the image would be significantly different as the layer of skin would be at the bottom of the image. [insert image here]

Rotation Range

The rotation range parameter will randomly rotate the image up to a degree range given. This will allow for images generated that will effectively look like the ultrasound has been taken at different angles.

Brightness Range

This augmentation is used by giving a range of values for changing the brightness shift. Experimenting with this will change the pixel value in the image by a random number in the given range.

Re-scaling

For this parameter each pixel is multiplied by a given value. In this experiment the value is 1/255 so that the pixels will range from 0 to 1. Some networks may benefit from this.

Zoom Range

The zoom range will create a new image that has been zoomed in by a random amount between a range. A float or a range can be given. If a float is given the range will be [1-float, 1+float]

2.7.2 Experiments with Data Augmentation to Improve Performance

Data Augmentation with Facial Expressions

Tong et al. (2019) explores facial expression recognition however one of the obstacles faced was that there was a small amount of data available. Deep convolutional neural networks generally require a large amount of training data to achieve ideal results. It was discussed that there was two key problems with deep facial expression recognition which are: over-fitting due to a lack of sufficient training data and that facial expression changes are subtle and changeable. The first-order information is insufficient to provide more discriminant information (Tong et al. (2019)).

Data augmentation techniques could be a solution to the first problem. The techniques that have been explored are to apply a jitter to the image, then follow up with horizontal flipping, and then erasing parts of the image. These techniques were applied to the training data. To create testing data, flipping and shifting the image was applied.

Two datasets were explore. They were real-world affective faces database (RAFDB) and the statistic facial expression in the wild (SFEW). It was found that data augmentation was very effective to improve the classification performance. For the RAFDB dataset the accuracy rate was increased by 1-7% and for the SFEW dataset there was an increase of 16-32% (Tong et al. (2019)). There was a larger increase with the SFEW dataset due to it being a smaller dataset.

Data Augmentation with Medical Thermography

Ornek & Ceylan (2019) compared different traditional transformations for data augmentation for medical thermography. A problem that was discussed using convolutional networks was that there needed to be balanced and sufficient data to produce ideal results. This is a problem in which medical imaging faces due to there being a small amount of data available. The traditional augmentation techniques that were explored where rotating, mirroring, zooming, shearing, histogram equalisation, colour changing, sharpening, blurring, brightness enhancement and contrast changing.

The experiment consisted of using a variety of different techniques that was combined such as using contrast changing, brightness changing and sharpening. The results from the experiment showed that using image augmentation can produce an increase in accuracy, sensitivity and specificity. However, it was also found that some combinations could also decrease the performance of the CNN.

The control results with no augmentation achieved an accuracy of 73.55%, sensitivity of 76.36% and specificity of 70.73%. The combination of rotation changing, mirroring and zooming got an accuracy of 64.26%, sensitivity of 69.77% and specificity of 58.76%. Whereas using contrast changing, sharpening and blurring achieved 99.84% accuracy, 99.85% sensitivity and 99.82% specificity (Ornek & Ceylan (2019)).

Data Augmentation with CT scans

Another investigation to the use of augmentation for medical images was conducted by Namozov & Cho (2018) to improve the performance of CNNs due to there being a small amount of data available. Training a deep CNN from scratch is a complicated task due to there being a small amount of medical data available.

The types of augmentation that was experimented with was horizontal and vertical flipping and rotation. The parameter set for rotation was 90°. This enabled the original training data to increase from 250 images to 1000 images. The images were trained using a deep CNN inspired by the VGG-Net.

When the network was trained for the original dataset, a training accuracy of 85.723% and a testing accuracy of 72.61% was achieved. The dataset containing augmented images achieved a training accuracy of 93.157% and testing accuracy of 92.25% (Namozov & Cho (2018)).

2.8 Generative Adversarial Networks (GANs)

Goodfellow et al. (2014) found that using generative models and discriminatory models alone did not provide the results needed discover hierarchical models that represent probability distributions over data. They found that using discriminatory models provided good result but generative models have not due to having a difficulty of approximating many intractable probabilistic results (Goodfellow et al. (2014)).

This led to the proposal of an adversarial nets framework. This is where the generator is placed against an adversary; this would be the discriminator. The generators job is to create an image in which the discriminator thinks its a real image. By putting these two networks against each other and it forces each of the models to train and get better than the other model until the synthetic image generated are indistinguishable from a real image. For example, a forger would create fake money which a banker has to tell the difference of if it fake or real. If it is fake, then the forger would go back and try to create a better copy. If the banker thinks it is real, then the banker would go and learn how to spot better copies of fake money. This is repeated until both the banker is good at spotting fake money and the forger is good at creating fake notes.

2.8.1 Improvements of GANs

The use of GANs to generate synthetic images with fine detail is still very difficult due to the data needed to train a GAN. Usually, a GAN would be trained from scratch due to the need to balance the discriminator and the generator. If the discriminator used weights transferred from another model, it may hinder the ability for the generator to create diverse images if there is too much negative feedback. To tackle this problem, Baek et al. (2019) created a GAN that uses a pretrained network resulting in two discriminators being used along with a novel energy function. DenseNet was used as the transferred model for one of the discriminator (Baek et al. (2019)). It was found that when the DenseNet model was used with a standard GAN loss function and learning algorithm that the generator was only creating specific images that the discriminator would classify as real. This is a problem as the aim of using GANs to create more data for a CNN is that there needs to be a variety of images created that are generated. If only specific images are generated then it will not help performance of the networks.

Baek et al. (2019) algorithm works by initially training the generator and the untrained discriminator together first, and then use both the discriminators later. This is so that initially the general knowledge of generating samples is created, and then the generation ability can be improved using the pretrained discriminator.

2.8.2 Uses of GANs

Using GANs to Create Tissue Data

Zhang et al. (2018) used GANs to create synthetic data of tissue and achieved an accuracy of 98.83%. The aim of the project is to solve the problem of not having enough data to train a network to improve the ability to recognise human thyroid tissues.

DCGANs use the typical concept of a GAN which is to use a generator and a discriminator network. A CNN is implemented in both of the networks. The main aim of the generator is to create a fake image which is mapped from random noise. The discriminator calculates the effectiveness of the generator by predicting if the image is fake or not. A probability will be created and the aim of the generator is to get the probability as low as possible for detecting the synthetic images. Due to both of these networks competing against each other, it creates a Min Max game for each network to achieve the best result. Zhang et al. (2018) created an extra 600 thyroid images and

600 non-thyroid images and achieved an accuracy of 98.83%. Furthermore, if more synthetic images were used then there was an increase in accuracy.

2.8.3 Using GANs to Create Intravascular Ultrasounds (IVUS)

Tom & Sheet (2018) investigated the use of a GAN inspired approach to generate realistic IVUS. This was achieved by employing a simulation of ultrasound imaged using a physics based simulator with tissue echogenicity maps (Tom & Sheet (2018)). Then a GAN generates low resolution images from the simulated images to create a speckle map. The image created was of size 64x64. The next step was to take the output image of the first GAN and then the second GAN generates realistic high resolution images by adding realistic artifacts. The images generated were of size 256x256.

The results of this experiment was calculated using a Visual Turing Test. Each evaluater was presented with 20 pairs of images in which one was real and the other was synthetic. The presenter then had to choose which one of the image in the pair was the real image. There was 260 pairs used and 147 of those were correctly identified giving a 56% chance the real image was chosen.

Chapter 3

Methodology and Results

3.1 Data

The dataset explored contains ultrasound images of different muscles from a patient. The original size of the image is 614x820. This image contains the ultrasound image as well as other information such as the muscle scanned and information about the ultrasound itself such as the frequency used. Figure 3.1 shows two of the original images.

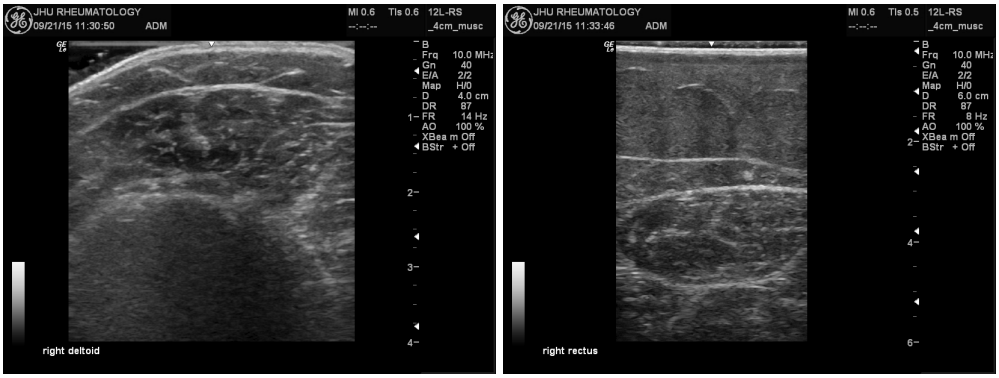


Figure 3.1: Original muscle ultrasound images

In this experiment, only the image of the ultrasound is needed therefore the image will be cropped. The dataset will then contain two different sizes of images. They are 476 x 476 and 318 x 476, figure 3.2 shows the two images that have been cropped. To feed these into the neural network, they need to be resized so that they are both the same. There will be two datasets created from this which will consist of the images

being sized 224 x 244 and 299 x 299. This is due to MobileNet and VGG16 that will be using images sized 224x224 and Inception-V3 which uses images sized 299 x 299. The models that will be trained from scratch will use the images sized 224x224.

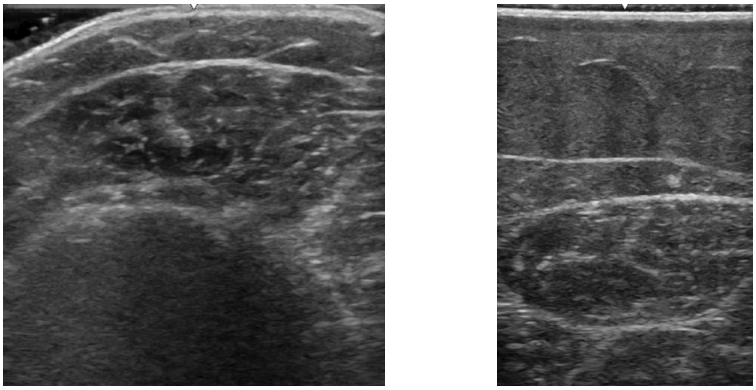


Figure 3.2: Cropped muscle ultrasound images

3.2 Configuration

During this experiment there will be some parameters which will be the same throughout testing. The learning rate will be set to $1x10^{-4}$ using the Adam optimiser. During the running of the CNN, some callbacks will be used such as early stopping and reducing the learning rate. Both of these will monitor the validation loss. The learning rate will be adjusted by a factor of 0.1 with a patience of 20 meaning that after 20 epochs of no improvement the learning rate will be changed. The CNN will stop training if the validation loss has no improvement after 30 epochs. The code for the callbacks is shown in figure B.3 in Appendix B. Furthermore, weightings will be applied to the classes due to there being a class imbalance of having more myositis patients than healthy patients, this would help reduce the network favouring myositis over healthy patients. The weighting for myositis will be one and the weighting for healthy patients will be set to *number of myositis patients/number of healthy patients*.

For each experiment run, there will be 8-fold cross validation using the patients, it was decided that due to there being a class imbalance of 47 myositis patients and 33 normal patients there will be 6 myositis patients and 4 normal in the testing set so that the ratio is similar to that of the original dataset, however for the eight split contain

the rest of the patients that have not been tested. The rest of the data will be using in training which will be split with a ratio of 80:20 for training and validation respectively.

It was chosen to split the patients so that at any point in a fold, the patients images are not in both training and testing. This was so that there is no chance of having the images mixed between the datasets have any artificial improvement to the CNN because if this was used in a real scenario, the patients images to be classified would not be used in training, therefore it is important that the patient's data is not mixed through training and testing of the models investigated.

3.3 Training from Scratch

Initially, there will be two CNNs that will be trained from scratch. They will both be trained and tested using the original dataset with cross validation.

The first CNN trained from scratch is based of a classic CNN in which it is a series of convolutional layers followed by pooling layers and ending with some fully connected layers. Figure B.1 in Appendix B shows the layers in the first CNN trained from scratch.

The second CNN trained from scratch is purely convolutional so that the only fully connected layer is the layer used for classification. Also there are no spatial dimensions. This was tested out of curiosity to find what results that might be found. Figure B.2 in Appendix B shows the layers in the second CNN trained from scratch.

Table 3.1 shows the results from both the CNNs trained from scratch. The first CNN performed very poorly compared to the second one. There is a difference of 11.4% accuracy between the first and second CNN tested. Also there was a difference of 0.369 in F1 measure. The results of this show that these networks performed quite poorly and was not good at classifying either myositis or healthy patients.

Data/augmentation	Accuracy	Sensitivity	Specificity	F1 Score
CNN 1	0.373	0.134	0.708	0.171
CNN 2	0.487	0.531	0.436	0.540

Table 3.1: Training from Scratch Results

3.4 Transfered Models with Augmented Data

The three models that will be tested are: VGG16, MobileNet and Inception-V3. These models will be tested individually and then with a variety of augmentation techniques. These will then be compared to find which produces the best results.

The augmentations to be used are horizontal flipping, a rotation up to: 10°, 20°, 30°, 40°, a brightness range of 0-0.5, 0.5-1.0 and 1.0-1.5, rescaling between 0 and 1 and a zoom range of up between 0.9 and 1.1, 0.7 and 1.3, and between 0.5 and 1.5.

3.4.1 VGG16

The first model to be tested is the VGG16 model. Originally, there are two fully connected layers with 4096 neurons per layer. However, due to the large memory requirements needed for the network, it was decreased to 1024 neurons in the first node and 512 neurons on the second node. A dropout of 0.5 was used as well to help reduce overfitting. Furthermore, the final fully connected layer was changed so that it can be used for this classification problem, the amount of neurons was changed from 1000 to 1.

Table 3.2 shows the results of using the VGG16 network to classify the images using the normal dataset and then using a variety of augmentations. When using the normal data, an accuracy of 63.0% was achieved with an F1 score of 0.367. The augmentation that provided the best increase of results was rescaling the pixel values between 0 and 1. Doing so achieved an accuracy of 72.9% with an F1 score of 0.639. By applying this augmentation the performance of the network improved significantly. However, it has to be noted that the other augmentations applied to the data did not improve the performance, but in fact it decreased the performance.

Data/augmentation	Accuracy	Sensitivity	Specificity	F1 Score
Normal	0.630	0.380	0.992	0.367
Horizontal Flip	0.609	0.344	0.994	0.338
Rescale	**0.729**	0.939	0.426	0.639
Rotation 10°	0.594	0.322	0.992	0.317
Rotation 20°	0.611	0.350	0.994	0.341
Rotation 30°	0.606	0.341	0.992	0.334
Rotation 40°	0.595	0.320	0.998	0.316
Brightness Range 0.0-0.5	0.614	0.428	0.919	0.411
Brightness Range 0.5-1.0	0.619	0.375	0.973	0.364
Brightness Range 1.0-1.5	0.614	0.354	0.993	0.345
Zoom Range 0.1	0.608	0.341	0.999	0.333
Zoom Range 0.3	0.593	0.316	0.998	0.314
Zoom Range 0.5	0.599	0.324	1.000	0.321

Table 3.2: VGG16 Results

3.4.2 MobileNet

The next model to be tested will be the MobileNet model. When this model was used the final fully connected layer was changed from 1000 neurons to 1. Furthermore, a dropout layer of 0.5 was used to slow the learning as the network was overfitting the data very quickly.

Table 3.3 shows the results of using the MobileNet network to classify the images using a normal dataset followed by a variety of augmentations. When using the normal data, an accuracy of 77.7% was achieved with an F1 score of 0.816. The augmentation that proved the best result was using a zoom range of 0.3, which provided an accuracy of 81.5% with an F1 score of 0.839. By randomly changing the zoom of the images an increase of 3.8% in accuracy was achieved.

3.4.3 Inception-V3

The final model to be tested is the Inception-V3 model. Before this model, was trained, some changes had to be made to the model. The final fully connected layer was changed from 1000 neurons to 1. Furthermore, in this model, a dropout layer of 0.6 was used to slow the learning as the network was overfitting the data very quickly.

Table 3.4 shows the results of using the Inception-V3 network to classify the images

Data/augmentation	Accuracy	Sensitivity	Specificity	F1 Score
Normal	0.777	0.989	0.476	0.816
Horizontal Flip	0.786	0.998	0.476	0.857
Rescale	0.785	0.956	0.547	0.820
Rotation 10°	0.775	0.966	0.506	0.836
Rotation 20°	0.792	0.989	0.511	0.846
Rotation 30°	0.781	0.989	0.487	0.854
Rotation 40°	0.768	0.970	0.480	0.844
Brightness Range 0.0-0.5	0.772	0.993	0.458	0.862
Brightness Range 0.5-1.0	0.782	0.993	0.475	0.855
Brightness Range 1.0-1.5	0.774	0.962	0.508	0.833
Zoom Range 0.1	0.793	0.996	0.504	0.852
Zoom Range 0.3	**0.815**	0.998	0.556	0.839
Zoom Range 0.5	0.790	0.999	0.491	0.856

Table 3.3: MobileNet Results

using a normal dataset followed by a variety of augmentations. When using the normal data, an accuracy of 77.6% was achieved with an F1 score of 0.824. The augmentation that proved the best result was using a random rotation of up to 30°, which provided an accuracy of 81.3% and an F1 score of 0.837. By using random rotations, an increase of 3.7% in accuracy was achieved.

Data/augmentation	Accuracy	Sensitivity	Specificity	F1 Score
Normal	0.776	0.986	0.477	0.824
Horizontal Flip	0.794	0.996	0.505	0.851
Rescale	0.802	0.992	0.533	0.842
Rotation 10°	0.799	1.000	0.516	0.851
Rotation 20°	0.772	0.972	0.491	0.843
Rotation 30°	**0.813**	0.995	0.553	0.837
Rotation 40°	0.802	0.997	0.523	0.847
Brightness Range 0.0-0.5	0.779	0.978	0.497	0.844
Brightness Range 0.5-1.0	0.788	1.000	0.487	0.857
Brightness Range 1.0-1.5	0.793	1.000	0.500	0.855
Zoom Range 0.1	0.767	0.962	0.490	0.835
Zoom Range 0.3	0.787	0.997	0.491	0.857
Zoom Range 0.5	0.791	0.999	0.490	0.855

Table 3.4: Inception-V3 Results

3.5 Discussion of Results

Overall, the best model was the MobileNet model using an augmentation of a zoom range of between 0.7 and 1.3. This achieved the best result which was an accuracy of 81.5% with an F1 score of 0.839. Inception-V3 using the augmentation random rotations of up to 30°achieved a similar accuracy of MobileNet which was 81.3%. It was found that in every dataset tested, both MobileNet and Inception-V3 was better at classifying a myositis image as myositis then it was at classifying a healthy image as healthy. Interestingly, VGG16 followed this trend until the pixels were rescaled between 0 and 1 in which it was better at classifying healthy patients as healthy than myositis patients as having myositis.

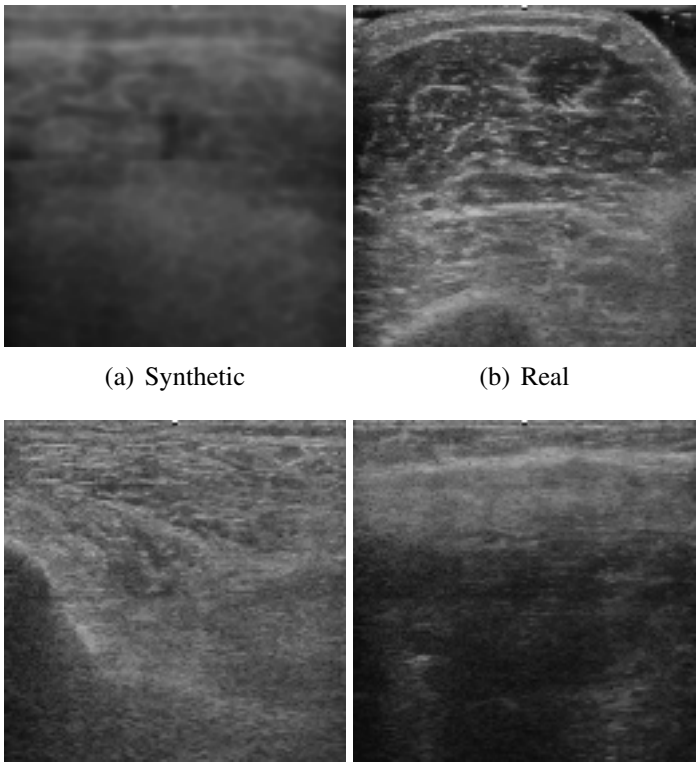
Compared to Burlina et al. (2017) results, an increase of accuracy of 5.3% was achieved. However, it may not be a fair comparison of results. A problem which is found is that Burlina et al. (2017) cross validated their results of 70% training, 10% validation and 20% testing. But have made no mention that the patients will be separated so that they do not appear in both the training and testing at one time. In the testing conducted in this paper, it was ensured that there was no patient appearing in both the training and testing at one time. By mixing this data, it could artificially increase the performance of the CNN.

Overall, it can be agreed that further work needs to be done to be able to help the network distinguish the difference between myositis and healthy patients better.

3.6 Generative Adversarial Networks (GANs)

GANs were experimented with to try to generate some new images to enhance the dataset. CNNs tend to perform much better when there is more data available to it. The idea behind using GANs is to experiment and see what images can be produced. Figure 3.3, shows a synthetic image generated from the GAN compared to some real images. The image created is the only the image created by the network which could be an indicator that the discriminator is learning too quickly compared to the generator or the generator is learning too quickly compared to the discriminator. It is important that an equilibrium is maintained of the loss for both the discriminator and the generator. If there is a spike in the loss then this would lead to the modal collapse and the network

would not recover from this. This network ran for 1500 epochs before it was decided to stop the training. Also, the images were resized to 128 due to being unable to use the full size images due to the large memory requirements. Further improvements will be discussed later in Chapter 4.



(a) Synthetic (b) Real

(c) Real (d) Real

Figure 3.3: Synthetic image generated by GAN compared to real images

Chapter 4

Conclusions and Future Work

This project investigated the use of CNNs with a patients ultrasound to diagnose the disease myositis. Different models were tested along with a variety of augmentations. Furthermore, GANs were experimented with to attempt to use them to build a larger dataset for training to provide more images which could help improve the performance of the networks.

Overall, there was good performing CNNs tested in this experiment. Both MobileNet and Inception-V3 achieved good results, however, there were some problems found. Both of these networks gave excellent results classifying the disease myositis but when classifying a healthy image it did not perform as well. A reason for this could be the class imbalance in the dataset. Using a weighting alone may not be enough but instead, when augmenting the data for example, ensure that there are equal amounts of data generated per class. This would help reduce the problem of an imbalanced dataset. Another option could be to look into the main features of what makes a myositis image, and what makes a healthy image, and focus on those features to make them stand out more in the images, helping the network create better feature maps. Using segmentation could help as well, as it will allow for the network to be trained on the important parts of the image and so that unimportant parts of the image can be ignored.

For future work, there are many different avenues that can be explored which could help improve the performance of the network. The first obvious step would be to acquire more data. Preferably, if the new data acquired contained more patients and fewer images per patient, rather than a small number of patients with a lot of images per patient, it could potentially allow for the model to be able to generalise more, this

could also help stop some of the overfitting that was taking place.

There are some other problems found, that are with the dataset itself. There is a large gap in age from aged 23 to 84, there is not an equal amount of male to female patients and furthermore, the duration of the disease in each patient could not be controlled. This ranged from 6 months to 360 months.

There are up to seven different muscles scanned per patient. Each muscle has a different structure which could potentially of hindered the performance of the CNN, especially with having a small dataset and some muscles appearing more often than others. Ideally, there would be the same amount of images per muscle in each class. A problem with the current network is that it has to be able to identify the muscle type and then the class. To make this easier for the network, training several networks for each muscle type could potentially lead to better results. This would mean that the clinician would have to input what muscle is being scanned so the correct network can be used. However, currently this is not feasible. Due to the disease being rare, it would be difficult to create datasets that are large enough the can effectively train the network.

Another problem when using the networks trained with this dataset is that it could face problems when it is used by other clinics with different equipment that use different parameters such as the frequency and the depth used, to combat this, a dataset could be created by using different machines that use similar parameters which could help the network generalise more allowing it for widespread use.

When training the GAN, some difficulties arose. Training a GAN takes a long time making it difficult to monitor, especially on older hardware. During the training there was a point in which the loss for the generator increased rapidly and maintained a high loss. This meant that a modal collapse occurred and it would require some hyperparameter changes.

Due to the time limit, the GAN was only able to be trained for a significant amount of time once which resulted in one image being created and training using the full sized image took a long time, instead the images were resized to 128 x 128.

For future work, there needs to be some adjustment made to the network to allow

for the generator and the discriminator to be trained at a similar speed. Also, training the GAN using the full sized image may be beneficial as there would be no feature maps lost when resizing. Another change that would be made is that the images in the dataset are vastly different and the myositis and healthy patients are mixed. Figure 3.3 shows three real images in which the structure is quite different. To solve this, Conditional GANs could be used so that classes could be given such as a myositis image of the right deltoid which would create images that are similar to that.

Another method that could be used to improve performance is the use of saliency maps. A saliency map contains a pixels unique quality, this can create an image that is more meaningful.

An example in which saliency maps have been used to improve a CNNs performance with ultrasound images was conducted by Kumar et al. (2016) which uses two CNNs. One CNN is trained on the normal images, and another CNN is trained on the saliency images. The outputs of these are concatenated together before classification. When this was compared to a state of the art method, the performance increased significantly.

Gradient-weighted Class Activation Mapping (Grad-CAM) could be used which could be very helpful for the clinician. This is a form of explainable AI which aims to help show the user why the network has made its decision. The idea for this is to have a ‘heatmap’ which shows the clinician where on the image is being used for classification. This could help the clinician so that their choice of diagnosis can be made faster as the Grad-CAM could show them an area of the image where the muscle fibres indicate myositis.

References

Alsabahi, Y. A. L., Fan, L. & Feng, X. (2018), Image classification method in dr image based on transfer learning, *in* '2018 Eighth International Conference on Image Processing Theory, Tools and Applications (IPTA)', pp. 1–4.

Baek, J., Yoo, Y. & Bae, S. (2019), 'Adversarial learning with knowledge of image classification for improving gans', *IEEE Access* **7**, 56591–56605.

Burlina, P., Billings, S., Joshi, N. & Albayda, J. (2017), 'Automated diagnosis of myositis from muscle ultrasound: Exploring the use of machine learning and deep learning methods', *PLOS ONE* **12**(8), 1–15.
URL: *https://doi.org/10.1371/journal.pone.0184059*

Chen, H., Ni, D., Qin, J., Li, S., Yang, X., Wang, T. & Heng, P. A. (2015), 'Standard plane localization in fetal ultrasound via domain transferred deep neural networks', *IEEE Journal of Biomedical and Health Informatics* **19**(5), 1627–1636.

Chen, H. & Su, C. (2018), An enhanced hybrid mobilenet, *in* '2018 9th International Conference on Awareness Science and Technology (iCAST)', pp. 308–312.

Duan, Y., Zhong, J., Shuai, G., Zhu, S. & Gu, X. (2018), Time-scale transferring deep convolutional neural network for mapping early rice, *in* 'IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium', pp. 1136–1139.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. (2014), Generative adversarial nets, *in* 'Advances in neural information processing systems', pp. 2672–2680.

Guo, T., Dong, J., Li, H. & Gao, Y. (2017), Simple convolutional neural network on image classification, *in* '2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)(', pp. 721–724.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. & Adam, H. (2017), 'Mobilenets: Efficient convolutional neural networks for mobile vision applications', *CoRR* **abs/1704.04861**.
URL: *http://arxiv.org/abs/1704.04861*

Kumar, A., Sridar, P., Quinton, A., Kumar, R. K., Feng, D., Nanan, R. & Kim, J. (2016), Plane identification in fetal ultrasound images using saliency maps and convolutional neural networks, *in* '2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)', pp. 791–794.

Namozov, A. & Cho, Y. I. (2018), An improvement for medical image analysis using data enhancement techniques in deep learning, *in* '2018 International Conference on Information and Communication Technology Robotics (ICT-ROBOT)', pp. 1–3.

Ornek, A. H. & Ceylan, M. (2019), Comparison of traditional transformations for data augmentation in deep learning of medical thermography, *in* '2019 42nd International Conference on Telecommunications and Signal Processing (TSP)', pp. 191–194.

Simonyan, K. & Zisserman, A. (2015), Very deep convolutional networks for large-scale image recognition.
URL: *https://arxiv.org/abs/1409.1556*

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. & Wojna, Z. (2016), Rethinking the inception architecture for computer vision, *in* '2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)', pp. 2818–2826.

Temdee, P. & Uttama, S. (2017), Food recognition on smartphone using transfer learning of convolution neural network, *in* '2017 Global Wireless Summit (GWS)', pp. 132–135.

Tom, F. & Sheet, D. (2018), Simulating patho-realistic ultrasound images using deep generative networks with adversarial learning, *in* '2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)', pp. 1174–1177.

Tong, X., Sun, S. & Fu, M. (2019), 'Data augmentation and second-order pooling for facial expression recognition', *IEEE Access* **7**, 86821–86828.

Wang, S., Mu, X., He, H. & Xu, S. (2018), Specific objective recognition based on

convolutional neural network transfer learning, *in* '2018 2nd IEEE Advanced Information Management,Communicates,Electronic and Automation Control Conference (IMCEC)', pp. 270–273.

Yemini, M., Zigel, D. Y. & Lederman, D. D. (2018), Detecting masses in mammograms using convolutional neural networks and transfer learning, *in* '2018 IEEE International Conference on the Science of Electrical Engineering in Israel (ICSEE)', pp. 1–4.

Zhang, Q., Wang, H., Lu, H., Won, D. & Yoon, S. W. (2018), Medical image synthesis with generative adversarial networks for tissue recognition, *in* '2018 IEEE International Conference on Healthcare Informatics (ICHI)', pp. 199–207.

Zheng, Q., Tastan, G. & Fan, Y. (2018), Transfer learning for diagnosis of congenital abnormalities of the kidney and urinary tract in children based on ultrasound imaging data, *in* '2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)', pp. 1487–1490.

Appendix A

Appendix 1

START HERE - Basic Information

This form must be completed for all student projects.

Before you proceed

Some activities inherently involve increased risks or approval by external regulatory bodies, so a proportional ethics review is not recommended and a full ethical review may be required.

These may include:

i. Approval from an external regulatory body (including, but not limited to: NHS (HRA), HMPPS etc.);
ii. Misleading participants;
iii. Research without the participants' consent;
iv. Clinical procedures with participants;
v. The ingestion or administration of any substance to participants by any means of delivery;
vi. The use of novel techniques, even where apparently non-invasive, whose safety may be open to question;
vii. The use of ionising radiation or exposure to radioactive materials;
viii. Engaging in, witnessing, or monitoring criminal activity;
ix. Engaging with, or accessing terrorism related materials;
x. A requirement for security clearance to access participants, data or materials;
xi. Physical or psychological risk to the participants or researcher;
xii. The project activity takes place in a country outside of the UK for which there is currently an active travel warning issued by the authorities (see info button);
xiii. Animals, animal tissue, new or existing human tissue, or biological toxins and agents.

If any of these activities are fundamental to your project, please contact your supervisor to determine if a full application is required.

This form must be completed for each research project which you undertake at the University. It must be approved by your supervisor (where relevant) PRIOR to the start of any data collection.

In completing this form, please consult the University's ACADEMIC ETHICAL FRAMEWORK for ethical research.

A1 Please confirm that you will abide by the University's Academic Ethical Framework in relation to this project.

- (•) Yes
- ( ) No

A2 Are you submitting this application as a learning experience, for a unit which already has ethical approval? (please confirm with your supervisor)

- ( ) Yes
- (•) No

A3 Student details

Title	First Name	Surname
	[redacted]	[redacted]

Email [redacted]

A3.1 Manchester Metropolitan University ID number

18062462

A4 Supervisor

Title	First Name	Surname
Mr	Luciano	Gerber

Faculty Science and Engineering

Telephone +44 (0)161 247 1694

Email l.gerber@mmu.ac.uk

A5 Which Faculty is responsible for the project?

Science and Engineering

A6 Course title

7G6Z1015 Masters Project

A7 Project title

Machine Learning Based Approaches to Finding if a Patient has Myositis Based on Ultrasounds

A8 What is the proposed start date of your project?

14/06/2019

A9 When do you expect to complete your project?

27/09/2019

A10 Please describe the overall aims of your project (3-4 sentences). Research questions should also be included here.

To explore the use of deep neural networks (DNN) on a dataset containing ultrasound images and other features to predict if a patient has myositis.

A11 Please describe the research activity

Myositis is a disease, which affects the muscles causing them to be inflamed. Generally, an MRI scan is used to check if a patient has this. However, there can be some problems using MRI. It is costly to use, time consuming and some patients who have implants or pacemakers cannot use them. This leads to ultrasound. It is easy to use, provides an improved resolution for soft tissue scans. However, ultrasound is open to issue of the operator and interpreter bias.
Using machine-learning techniques on the produced images of the ultrasound can give us predictions that can be used to aid in diagnosing if the patient has the disease. For this project, DNNs will be experimented with to attempt to be able to accurately predict the correct classification of the image.
Initially, a simple DNN will be created to get a baseline performance. Then modifications will be made to attempt to increase the performance.
Next, the use of other features supplied with the data such as 'muscle strength' will be used to explore if this will increase accuracy.
If this work is completed in time, then it may be worth looking into using some computer vision techniques such as segmentation to see if this can help the classification process.

A12 Please provide details of the participants you intend to involve (please include information relating to the number involved and their demographics; the inclusion and exclusion criteria)

N/A

A13 Please upload your project proposal

Type	Document Name	File Name	Version Date	Version	Size
Project Protocol	TermsofReference	TermsofReference.pdf	14/06/2019	1	194.3 KB

Project Activity

B1 Are there any Health and Safety risks to the researcher and/or participants?

- ○ Yes
- ◉ No

B2 Please select any of the following which apply to your project

- ☐ Aspects involving human participants (including, but not limited to interviews, questionnaires, images, artefacts and social media data)
- ☐ Aspects that the researcher or participants could find embarrassing or emotionally upsetting
- ☐ Aspects that include culturally sensitive issues (e.g. age, gender, ethnicity etc.)
- ☐ Aspects involving vulnerable groups (e.g. prisoners, pregnant women, children, elderly or disabled people, people experiencing mental health problems, victims of crime etc.), but does not require special approval from external bodies (NHS, security clearance, etc.)
- ☐ Project activity which will take place in a country outside of the UK
- ☑ None of the above

B2.4 Is this project being undertaken as part of a larger research study for which a Manchester Metropolitan application for ethical approval has already been granted or submitted?

- ○ Yes
- ◉ No

Data

F1 How and where will data and documentation be stored?

My computer, its a publicly available dataset

F2 Will you be collecting personal data or sensitive personal data as part of this project?

- ○ Yes
- ◉ No

Insurance

F3 Does your project involve:

- ☐ Pregnant persons as participants with procedures other than blood samples being taken from them? (see info button)
- ☐ Children aged five or under with procedures other than blood samples being taken from them? (see info button)
- ☐ Activities being undertaken by the lead investigator or any other member of the study team in a country outside of the UK as indicated in the info button? If 'Yes', please refer to the 'Travel Insurance' guidance on the info button
- ☐ Working with Hepatitis, Human T-Cell Lymphotropic Virus Type iii (HTLV iii), or Lymphadenopathy Associated Virus (LAV) or the mutants, derivatives or variations thereof or Acquired Immune Deficiency Syndrome (AIDS) or any syndrome or condition of a similar kind?
- ☐ Working with Transmissible Spongiform Encephalopathy (TSE), Creutzfeldt-Jakob Disease (CJD), variant Creutzfeldt-Jakob Disease (vCJD) or new variant Creutzfeldt-Jakob Disease (nvCJD)?
- ☐ Working in hazardous areas or high risk countries? (see info button)
- ☐ Working with hazardous substances outside of a controlled environment?
- ☐ Working with persons with a history of violence, substance abuse or a criminal record?
- ☑ None of the above

Additional Information

G1 Do you have any additional information or comments which have not been covered in this form?

- ○ Yes
- ◉ No

G2 Do you have any additional documentation which you want to upload?

- ◉ Yes
- ○ No

G2.1 Please attach a copy of any other materials relevant to this application

Type	Document Name	File Name	Version Date	Version	Size
Additional Documentation	RA_Project_SoftwareDevelopment_270918	RA_Project_SoftwareDevelopment_270918.pdf	14/06/2019	1	369.0 KB

Signatures

H1 I confirm that all information in this application is accurate and true. I will not start this project until I have received Ethical Approval.

- ◉ I confirm
- ○ I do not confirm

H2 Please notify your supervisor that this application is complete and ready to be submitted by clicking "Request" below. Do not begin your project until you have received confirmation from your supervisor - it is your responsibility to ensure that they do this.

Signed: This form was signed by Luciano Gerber (L.Gerber@mmu.ac.uk) on 23/09/2019 12:06

H3 By signing this application you are confirming that all details included in the form have been completed accurately and truthfully.

Signed: This form was signed by [redacted] on 14/06/2019 10:31

The MANCHESTER METROPOLITAN UNIVERSITY
Faculty of Science and Engineering

RISK ASSESSMENT COVER SHEET

REFERENCE NUMBER: NPC / 270918 / JDE1.49

SCHOOL: Computing, Mathematics & Digital Technology

TITLE OF WORK: CMT Projects involving software development only

LOCATION OF WORK: John Dalton Building computing facilities, computers at student's own home etc.

INTENDED ACTIVITIES (attach methods sheets (e.g. standard operating practices) and work schedules to this form):

General use of computers to develop and test software. Method sheets and work schedules not applicable.

PERSONS AT RISK (list names of all individuals (including status e.g. staff/student), and/or unit(s) / course(s) undertaking the activity. For students please indicate course and level, for staff give contact email / phone number):

Undergraduate students.

HAZARDS (provide a summary of the hazards anticipated and attach detailed assessments with appropriate risk control methods to this form):

Repetitive Strain Injury – work related upper limb disorder

Back injury resulting from improper posture

Eye strain

Fatigue Stress

Possible risk from 240v electrical mains supply

Are these hazards necessary in order to achieve the objectives of the activity?

Yes

Hazard Rating (delete as appropriate): **Low**

HAZARDOUS SUBSTANCES/MATERIALS USED AND HAZARD CLASSIFICATION

(appropriate COSHH data sheets / risk assessments must be attached to this form**):**

ALL CONTAINERS OF HAZARDOUS SUBSTANCES SHOULD BEAR CORRECT HAZARD WARNING LABELS.

NAME OF MATERIAL *Please provide also approximate quantity and concentration if applicable.*	**HAZARD CLASS**	**HAZARD LABEL**	**DISPOSAL** *Hazardous materials must not be removed from laboratories. List disposal arrangements for* <u>all materials listed below</u> *in the location where* <u>the work will be carried out:</u>

RISK CONTROL METHODS (provide a summary of the hazards anticipated and attach detailed assessments with appropriate risk control methods to this form):

The hazards identified above are controlled by:

Facilities review when laboratories are commissioned
Induction session on H&S given to students by Technical Services Manager
School H&S information given in Student handbook
Posters in laboratories
PAT testing of equipment after three years
Annual H&S inspections

The laboratory workstations, whilst not legally required to be DSE compliant, (the continuous usage is too low to present risk) are fully compliant with current legislation. Monitors and keyboards are adjustable, chairs are adjustable and the lighting designed for both computer usage and associated reading activity. In each laboratory, there is an adjustable desk, suitable for wheelchair users, usually located in the next to the door.

Hazard Rating with Control Methods (delete as appropriate): **Low**

Will any specific **training** *be required (if* YES *give details)?* N/A

Are there any specific **first aid** *issues (if* YES *give details)?* N/A

PROCEDURE FOR EMERGENCY SHUT-DOWN (if applicable):

In the event of fire, flood or other emergency, evacuation of the laboratory would take place and the technical staff would subsequently make an assessment of the necessity of switch-off. As overall system control is vested in a separate server room, there would be little physical harm to any device in directly cutting the power to the mains for each individual lab.

Re-start of the lab may present problems of a technical nature but would not affect the personal safety or health of any individual.

IF OFF-SITE INDICATE ANY OTHER ISSUES (e.g. associated with: individual's health and dietary requirements (obtain off-site health forms for all participating individuals and indicate where this information will be located); social activities, transportation, ID requirements; permissions for access and sampling).

Not applicable – this form applies only to the laboratories listed

	NAME	**STAFF/STUDENT No.**	**DATE**
Originator	Nicholas Costen	01900261	270918
Supervisor	N/A		
Technical Manager			
Divisional / School Health and Safety Coordinator (p.p. HoS)			

DATE TO BE REVIEWED BY: September 2019

Aim

To explore the use of deep neural networks (DNN) on a dataset containing ultrasound images and other features to predict if a patient has myositis.

Learning Outcomes

To explore the use of DNN and how they can be used for image classification.

Exploring how additional features along with the images can help increase performance.

Project Description

Myositis is a disease, which affects the muscles causing them to be inflamed. Generally, an MRI scan is used to check if a patient has this. However, there can be some problems using MRI. It is costly to use, time consuming and some patients who have implants or pacemakers cannot use them. This leads to ultrasound. It is easy to use, provides an improved resolution for soft tissue scans. However, ultrasound is open to issue of the operator and interpreter bias.

Using machine-learning techniques on the produced images of the ultrasound can give us predictions that can be used to aid in diagnosing if the patient has the disease. For this project, DNNs will be experimented with to attempt to be able to accurately predict the correct classification of the image.

Initially, a simple DNN will be created to get a baseline performance. Then modifications will be made to attempt to increase the performance.

Next, the use of other features supplied with the data such as 'muscle strength' will be used to explore if this will increase accuracy.

If this work is completed in time, then it may be worth looking into using some computer vision techniques such as segmentation to see if this can help the classification process.

The data that will be used is a publicly available dataset.

References

https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0184059

https://github.com/jalbayd1/myopathy_US

Evaluation Plan

Each stage of the process will be compared using measures such as accuracy, precision and recall. For example a comparison will be made of using the images alone and using the images along with other data.

Activity Schedule

Create a basic DNN – 17/06/2019

Modify the DNN to improve the performance – 08/07/2019

Start adding additional features such as strength to improve performance – 29/07/2019

If time permits, explore the use of segmentation

Appendix B

Appendix2

```
input_shape = (224,224,3)
inputs = Input(shape=input_shape)

x = Conv2D(filters=32, kernel_size = (5,5), activation = 'relu')(inputs)
x = Conv2D(filters=32, kernel_size = (5,5), activation = 'relu')(x)
x = MaxPool2D(pool_size = (2,2), strides = 2)(x)

x = Conv2D(filters=64, kernel_size = (3,3), activation = 'relu')(x)
x = Conv2D(filters=64, kernel_size = (3,3), activation = 'relu')(x)
x = MaxPool2D(pool_size = (2,2), strides = 2)(x)

x = Conv2D(filters=64, kernel_size = (3,3), activation = 'relu')(x)
x = Conv2D(filters=64, kernel_size = (3,3), activation = 'relu')(x)
x = MaxPool2D(pool_size = (2,2), strides = 2)(x)

x=Dense(512,activation='relu')(x)
x = Dropout(0.5)(x)
x=Dense(256, activation='relu')(x)
x = Dropout(0.5)(x)
x = Flatten()(x)

preds=Dense(1,activation= sigmoid)(x)

model = Model(inputs=inputs, outputs=preds)
model.summary()
```

Figure B.1: Code showing the layers of the first CNN trained from scratch

```
input_shape = (224,224,3)
inputs = Input(shape=input_shape)

x = Conv2D(filters=32, kernel_size = (5,5), strides = 2, activation = 'relu')(inputs)
x = Conv2D(filters=64, kernel_size = (3,3), strides = 2,activation = 'relu')(x)
x = MaxPool2D(pool_size = (2,2), strides = 1)(x)
x = Conv2D(filters=64, kernel_size = (3,3), strides = 2,activation = 'relu')(x)
x = Conv2D(filters=64, kernel_size = (3,3), strides = 2,activation = 'relu')(x)
x = MaxPool2D(pool_size = (2,2), strides = 1)(x)
x = Conv2D(filters=64, kernel_size = (3,3), strides = 2,activation = 'relu')(x)
x = Conv2D(filters=64, kernel_size = (3,3), strides = 2,activation = 'relu')(x)
x = MaxPool2D(pool_size = (2,2), strides = 1)(x)

x = Flatten()(x)

preds=Dense(1,activation= sigmoid)(x)

model = Model(inputs=inputs, outputs=preds)
model.summary()
```

Figure B.2: Code showing the layers of the second CNN trained from scratch

```
model_save_name = "best_wts" + str(i) + ".h5"

earlyStopping = EarlyStopping(monitor='val_loss', patience=22, verbose=1, mode='min')

model_save = ModelCheckpoint(model_save_name, save_best_only=True,
    monitor='val_loss', mode='min')

reduce_lr_loss = ReduceLROnPlateau(monitor='val_loss', factor=0.1,
    patience=10, verbose=1, min_delta=1e-4, mode='min')
```

Figure B.3: Code showing the callbacks used