UNSW
AUSTRALIA

University of New South Wales

COMP9417 Project - MURA DataSet Classification

Name	zID
Daniel Al Mouiee	z5114185
Aydin Ercan	z5117952

August 11, 2019

Contents

List of Figures

List of Tables

1 Introduction

The MURA (MUsculoskeletal RAdiographs) data-set is a collection of x-rays conducted on patients who have experienced various fractures, breaks and injuries to their bones. These images depict various states of the patients' bone health ranging from healthy, unscathed bones to severe breaks requiring bone amendment measures. The idea of machines and autonomous systems complementing clinicians work is not new, in fact there exists many implementations both clinically and industrially, more specifically in *"Medical Imaging Analysis"*[1].

The ability to reduce tedious workload for clinicians and physicians for them to efficiently contribute to more meaningful tasks has been a major objective for Autonomous development. Architectures such as Convolutional Neural Networks (**CNNs**) have revolutionised image classification tasks and are positively influencing the medical field. As a result of this, this project was dedicated to building an artificially intelligent system capable of distinguishing x-rays of health bones from those of severely affected fractures and breaks.

This project explored the implementation of a CNN to distinguish the different classes of bone x-rays whilst providing it as little information as possible. This will rely on the network to learn the distinct features x-rays of different bones and different states of these bones respectively. To provide a baseline for further investigations, we implemented another 2 classification algorithms (namely K- Nearest Neighbours (**KNN**) and Random Decision Forests (**RDF**)) to assess the performance of the CNN and evaluate its use as opposed to simpler architectures.

2 Methodology

2.1 Data-set - Pre-processing and feature analysis

Due to the large quantity of images for each body part included in the MURA data-set and resource constraints (2 member group, implementation of other algorithms), only the images of Elbow and Wrist bones were considered for this project.

The raw images in MURA contain many **unnecessary artifacts** that will negatively affect the performance of the CNN. A range of "noisy" features were eliminated from the images, which are:

- Writings on the images which instruct radiologists and physicians of the bone's characteristics (eg. large 'L' or 'R' letters depicting whether it is a left or right limb, meta-data to do with the patients medical history). Such information is not informative for the CNN to judge whether the X-ray is if a normal or abnormal bone. On the contrary, it will actually increase bias.
- Images which contain multiple X-rays in 1. We attempted to not overload the network with too much information as we were already asking a lot from the network.
- Images in which the grey-scale is inverted. Since most of the X-rays are black and white in nature, where the soft tissue is represented with darker colours and higher density parts reflect brighter (or whiter) rays. So, to include images in which the grey-scale is inverted, this will again confuse the network further.

Hence, the images were cropped to eliminate the unnecessary data in the images. Generally, the images were cropped to include a segment of bone that will most likely contain a break, fracture and/or plates to amend the bone. Additionally, we only want to show the network images that contain the segment only and not include a large receptive field (eg. skin, additional bones not to do with the intended body part being analysed). This strategy has proven to be effective in other medical image analysis tasks.

The features which are desired for the network to learn are:

- Amount of asymmetric bright light emitted from the bone. This indicates that higher density materiel is present in the bone segment (usually metal plates indicting a break). An easy and essential feature for the network to utilise.
- The shape and symmetry of the bone's structure. This feature is harder to analyse as grey-scale images do not portray significant changes effectively like RGB images do.

The data was split into training and validation sets (ratio 4:1), with approximately 150 normal elbow, 150 abnormal elbow, 100 normal wrist and 100 abnormal wrist which were cleaned and filtered.

2.2 CNN Implementation

The model's design is nothing extraordinary in terms of the essential components. It is made up of:

1. Convolutional layers: 3 layers (namely *conv_1*, *conv_2*, *conv_3*) of filter sizes 32x32 for *conv_1*, *conv_2* and 64x64 for *conv_3*.
2. Fully Connected layers: 2 layers (namely *full_con_1*, *full_con_2*)
3. Max Pooling layers: 3 layers (namely *pool_1*, *pool_2*, *pool_3*) in between the convolutional layers. These are:

    - *convo_1* -> *pool_1* -> *convo_2*
    - *convo_2* -> *pool_2* -> *convo_3*
    - *convo_3* -> *pool_3* -> *full_con_1*
4. Flattening layer: 1 layer (namely *flat*) between *conv_3* and *full_con_1*

Included in the **Appendix** is a list of hyper parameters chosen for the CNN. This aspect of the network was not explored thoroughly as the objective of this project was to investigate the additional predictive power of CNN in comparison to simple classifiers.

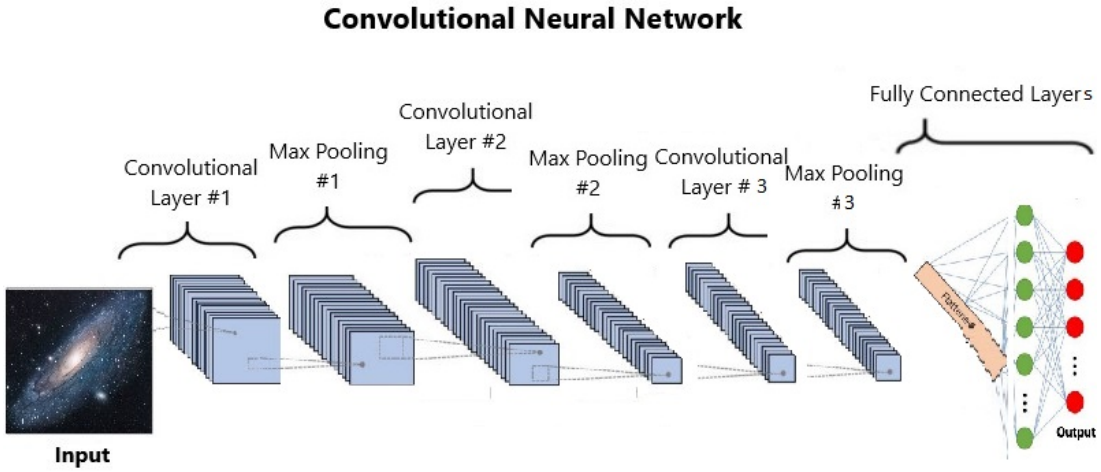

Figure 1: High Level view of the CNN's architecture

2.3 RDF Implementation

The random forest model was implemented using the **sklearn** python library. The cost and accuracy values were acquired through the use of the "accuracy_score" and "log_loss" methods in the **sklearn.metrics** library, which were then showcased through the use of line graphs from the **matplotlib.pyplot** library from python.

The method and code used to pre-process the image data however was acquired from an online source (Reference 2). The image data was translated into both a feature vector and a colour histogram (explained below), through the functions 'image_to_feature_vector' and 'extract_color_histogram'. The labels for the related images was set through identifying whether the image path to the images contained "negative" or "positive" as each image was in a folder relevant to its label.

For predicting and testing results, we applied two separate forms of data created from the image data itself. The first data represented the raw pixel intensity of the images, looking at how intense certain pixels on the image are and making deductions off of that (feature vector). The second data represented the colour histogram, looking at the overall color of the image and making deductions based off of that. A training and testing set was made for both of these types of data, followed by a testing prediction for both.

For the testing, we applied each data type with 7 different random forest classifiers. Each of these classifiers had a different number of trees applied to it (10, 20, 50, 100, 200, 500, and 1000). This was intended to provide

more detailed information on the overall accuracy and cost of the model. The outcome of the testing the wrist images is displayed in section 4.2.1 below.

2.4 KNN Implementation

The KNN model was also implemented using the sklearn python package as well as utilising the 2 functions 'score' and 'log_loss' to acquire the accuracy and cost data. The method and code used for data pre-processing was also the same as the one used in the Random Forest model.

The training and testing for the model was similar to Random Forest as well. However, for testing and predicting, rather than creating new random forest classifiers with differing numbers of trees, KNN focused on creating new models with differing numbers of neighbours. Four different numbers of neighbours was used for testing (10, 20, 50, and 100). The outcome of testing the wrist images is displayed below in section 4.2.2 below.

2.5 Experimentation

1. The data-sets for "Elbow" and "Wrist" x-rays were pre-processed, filtered (as described above) and split into "normal" and "abnormal" classes. By far, this was the most time consuming process of the project and limited the number of "pre-processed" images used for training and validation.

2. The development of a python3.6 scripts to train the CNN using TensorFlow1.1 library. This was done with the assistance from an online tutorial to assist with aspects of data pre-processing and a prediction script for testing on an unseen data-set.

3. The collection of different metrics to evaluate the performance of the CNN using TensorBoard.

4. The development of simple classifiers (**KNN** and **RDF**) to evaluate the additional predictive power the CNN offers.

3 CNN Results

Below, is a summary of the Convolutional Neural Network's training and Validation accuracy and cross entropy scores. These were collected using TensorBoard and by averaging out multiple iterations of training (highlighted in the background in a faint colour):

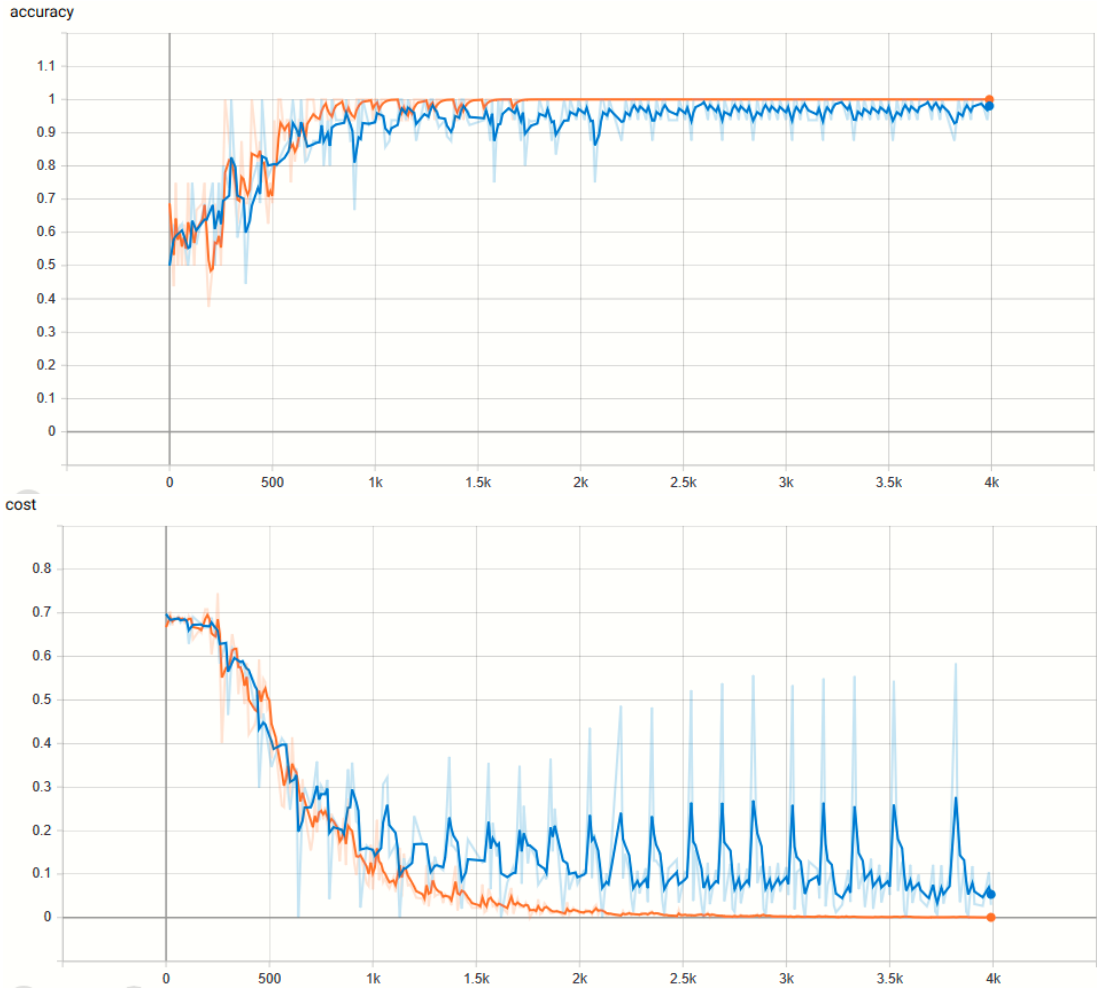

Figure 2: CNN Accuracy and Cross Entropy scores on the Wrist DataSet, Orange on Training set, Blue on Validation Set *(Normal Vs Abnormal)*

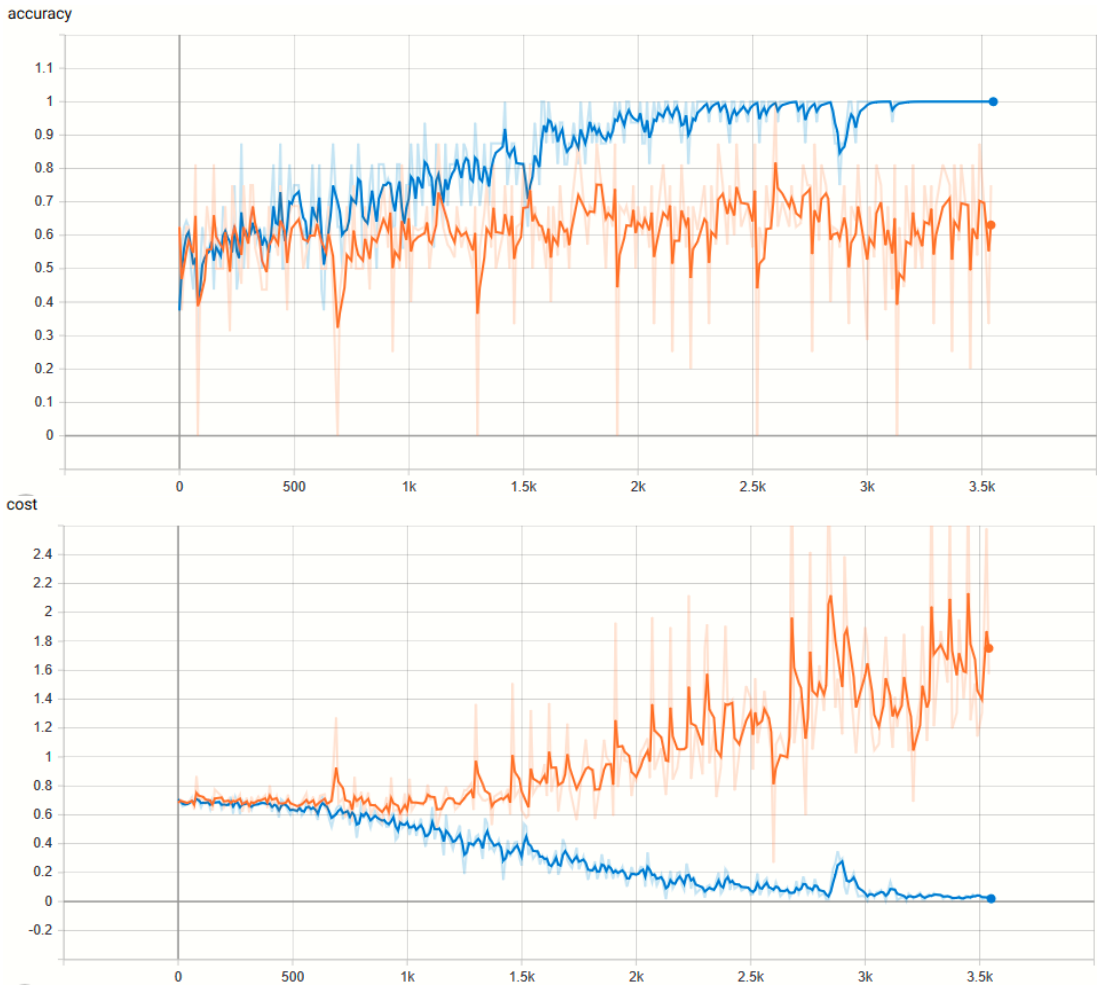

Figure 3: CNN Accuracy and Cross Entropy scores on the Elbow DataSet, Blue on Training set, Orange on Validation Set *(Normal Vs Abnormal)*

The tables below are summaries of running the trained CNN on unseen prediction sets of data and measuring the accuracy, precision and recall (where we assume positive cases are *Normal* x-ray images):

Accuracy (%/100)	**Precision (%/100)**	**Recall (%/100)**
0.909	0.741	0.833

Table 1: Average Accuracy, Precision and Recall values on an unseen Wrist data-set

Accuracy (%/100)	**Precision (%/100)**	**Recall (%/100)**
0.818	1.000	0.666

Table 2: Average Accuracy, Precision and Recall values on an unseen Elbow data-set

4 Discussion

4.1 CNN Performance

For a simple design, the model was able to identify abnormal x-rays from those that are normal with high accuracy for the Wrist data-set. The validation accuracy, however, did not eventually converge to a maximum, nor the validation cost to a minimum, suggesting that the model could have been too simple. Adding additional convolutional layers may have assisted in reducing the cross entropy scores and allowing it to finally stabilise at a minimum value. Despite the sub-optimal validation scores, the table values suggest that, on an unseen testing set, the model is capable of achieving high scores of accuracy, precision and recall.

Interestingly, the model performed better on the wrist data-set than on the elbow. Reasons for this could be that the wrist data-set was refined better where the desired features were captured better. Another potential reason could be that the elbow X-rays are harder to analyse than the wrist. For example, the wrist x-rays were mostly taken where the joint between the forearm and wrist was not flexed (straight forearm and wrist) whereas the elbow joint was flexed in some images. This change in joint angles may have contributed to the model's ability to distinguish between the different classes for the elbow data-set.

4.2 Comparing CNN to other models

The 2 extra models (Random Forest and KNN) were implemented using python as a means to compare our CNN's results with that of other models.

4.2.1 Random Forest Model

Below are illustrations of a Random Forest model's performance on the MURA data-set:

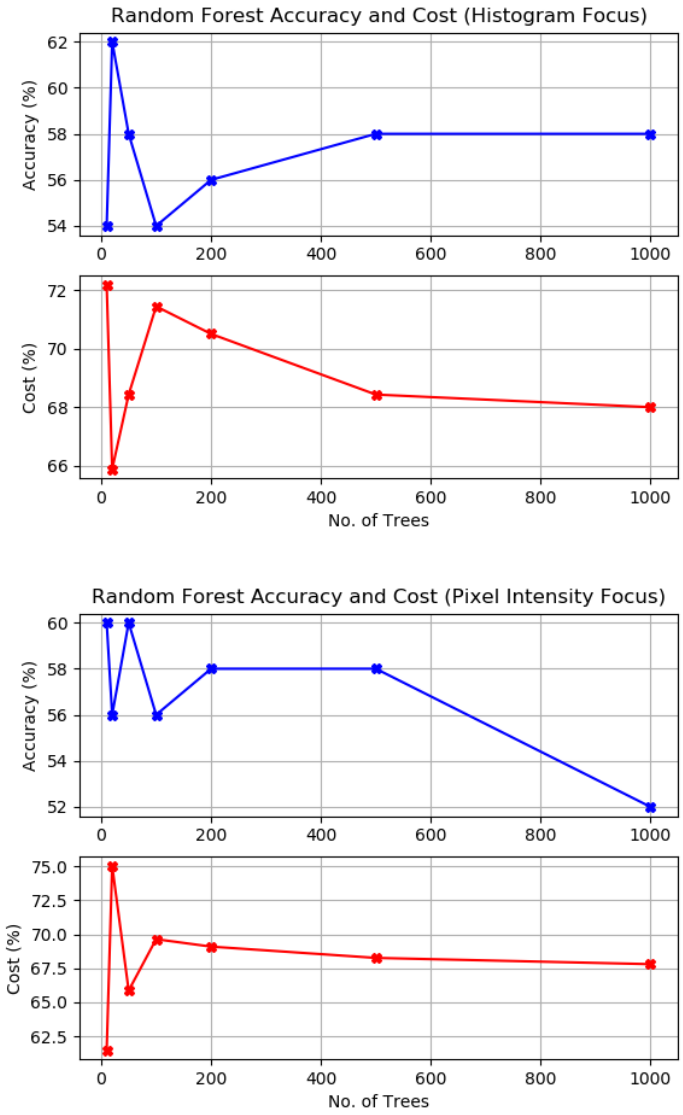

Figure 4: Random Forest Accuracy and Cross Entropy scores on the Wrist data-set *(Normal Vs Abnormal)*

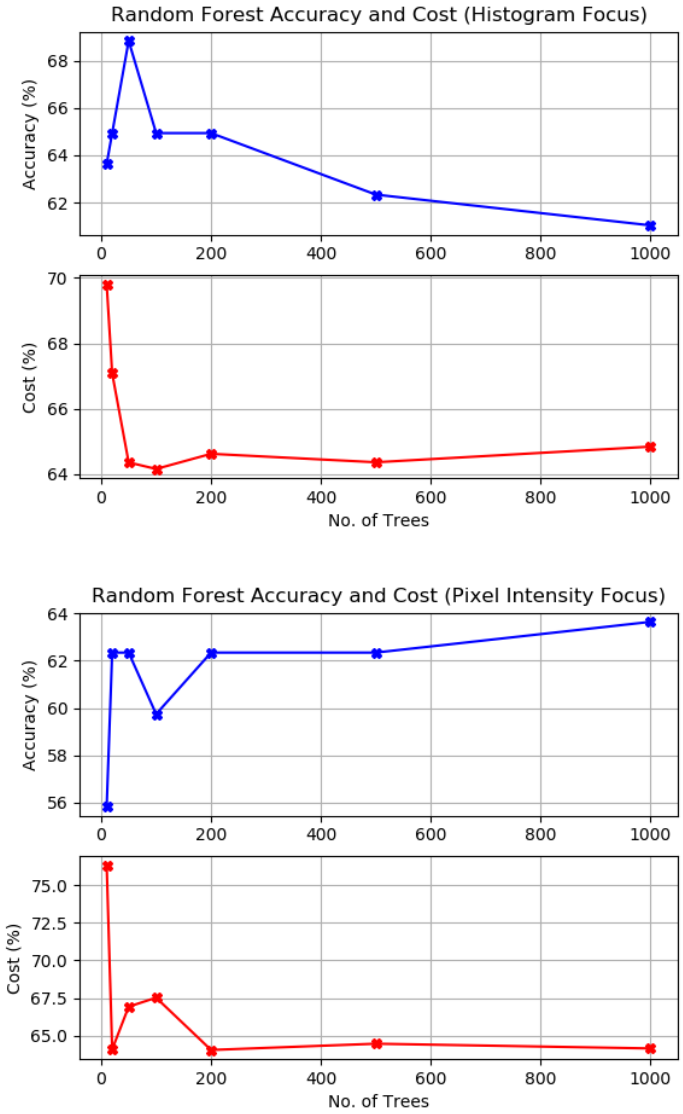

Figure 5: Random Forest Accuracy and Cross Entropy scores on the Elbow Data-Set *(Normal Vs Abnormal)*

The RDF model performance was mediocre with the maximum accuracy reaching around 73% on an unseen testing set. The cross entropy cost was hovering above 60% too indicting that the model isn't performing well either. The histogram focus approach performed better than the pixel intensity focus method, however increasing the number of leaf trees does not show a particular trend. The CNN definitely outperformed the RDF model in this experiment.

4.2.2 KNN Model

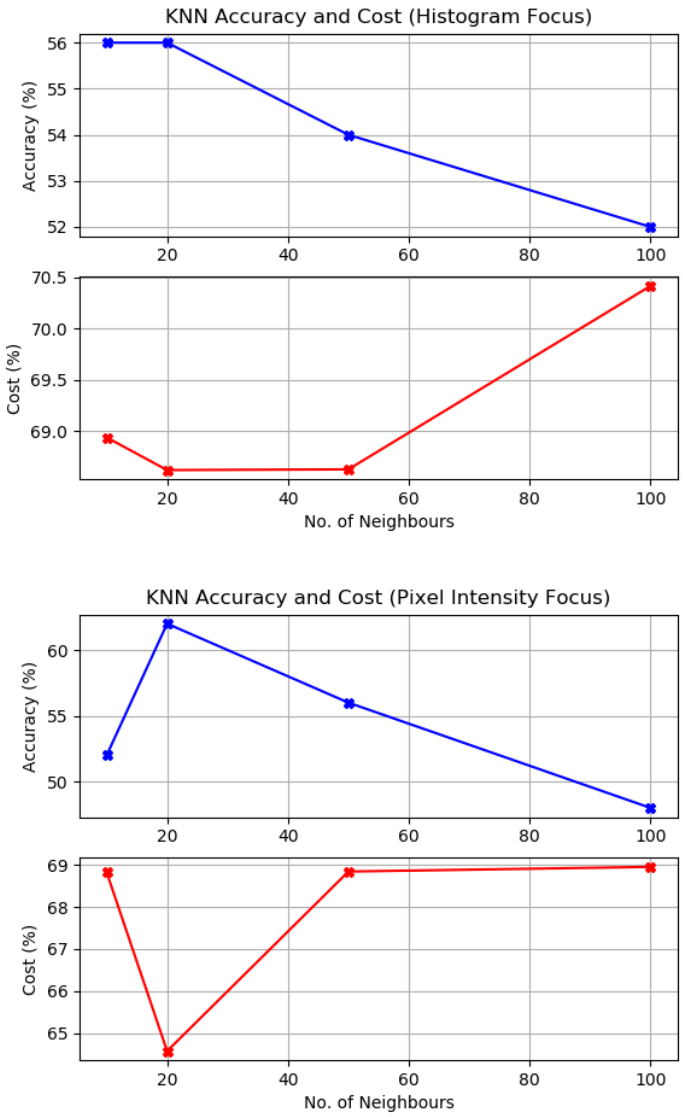

Figure 6: K- Nearest Neighbours Accuracy and Cross Entropy scores on the Wrist DataSet *(Normal Vs Abnormal)*

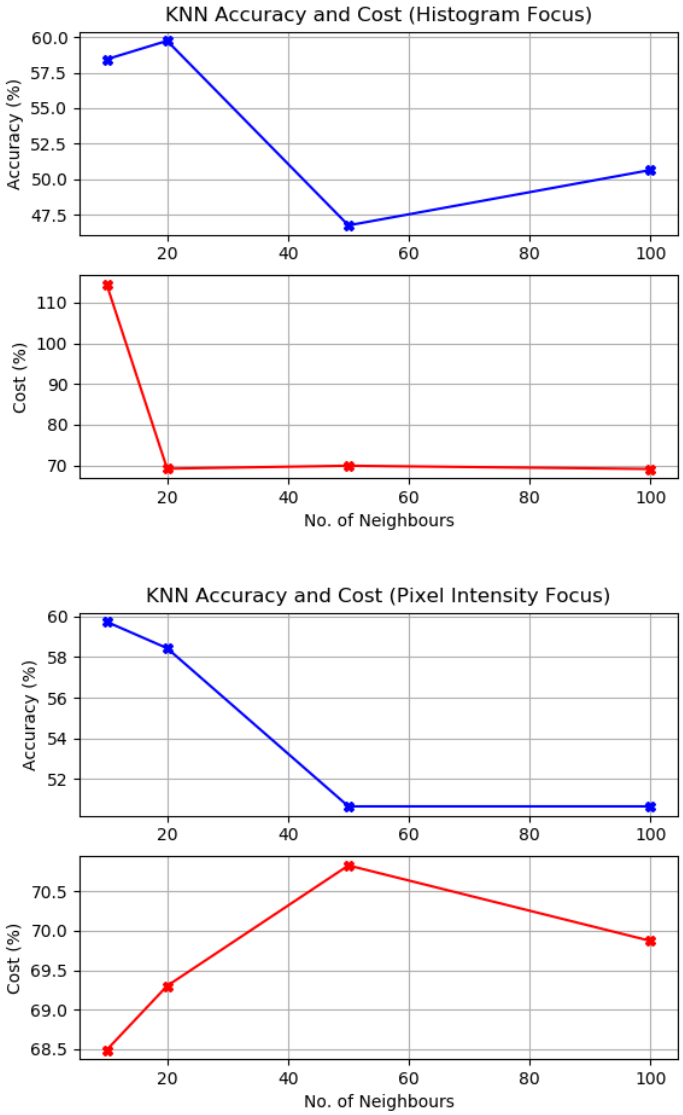

Figure 7: K- Nearest Neighbours Accuracy and Cross Entropy scores on the Elbow DataSet *(Normal Vs Abnormal)*

The same can be said about the KNN model, with a maximum accuracy 62% and much higher cross entropy scores. RDF actually outperforms the KNN model, hence being over-powered by the CNN as well. Interestingly, the Pixel Intensity Focus approach performed better than the Histogram focus (opposing the RDF model) with performance decreasing when the number of neighbours increased in the KNN model.

4.3 Visualisation of CNN Activation Layers

Below are visualisations of the layer activation present in the 3rd and final convolutional layer of the CNN model:

Figure 8: Visualisation of third convolutional layer on the features of (a) *Normal Elbow* bone X-Ray, (b) *Abnormal Elbow* bone X-Ray

Figure 9: Visualisation of third convolutional layer on the features of a (a) *Normal Wrist* bone X-Ray, (b) *Abnormal Wrist* bone X-Ray

It is clear that the 3rd layer begins to localise certain features and is rather dense and "blobby" for the final convolutional layer. This suggests that there is still valuable information that can be gained by increasing the number of convolutional layers as the fully connected layers need to do with these filters which are not specialised enough. Due to project constraints, we did not achieve the goal of optimising the CNN design as it is evident that it is in need of refinement and we set this as a goal for future projects.

5 Conclusions

The CNN effectively distinguishes between normal and abnormal x-ray images of the wrist and elbow. In comparison to the baseline models of the KNN and RDF, the CNN achieves higher levels of accuracy. However, the CNN design could have been optimised when analysing the layer activations and validation set metrics. Future plans for this project include exploring optimisation options for the CNN and fine-tuning hyper-parameters. In conclusion, medical imaging automation is a great field to explore and will continue to produce promising results with the ultimate goal of helping clinicians and patients.

6 References

1 V. Gulshan, *Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs*, 2016.

2 https://www.pyimagesearch.com/2016/08/08/k-nn-classifier-for-image-classification/

7 Appendix

7.1 CNN's Hyper-parameters

- Adam optimiser
- Learning rate = $10^{-}4$
- Activation function : Rectified Linear Unit (ReLU)
- Number of Epochs: 4000
- Batch size : 16