CAB420 assignment 2

Liam Hulsman-Benson, n9960392

Elliot Mcphearson,

Jayden Tye,

***Introduction/Motivation:***

clearly motivate your project, and describe the research question, and how it relates to previous works that have been done in this area.

***Related Work:***

briefly describe a small number of relevant existing approaches, and the objective of your work.

***Data:***

clearly describe the data set, any pre-processing, and the data split into training/validation/test.

A set of coloured images taken at a resolution of 1920X1080. The dataset is then cropped to only extract the hand region, this results in images of 400X400 pixels.

The downloaded data consisted of multiple folders:

* Original\_frames: The unprocessed RGB 400X400 pixel images of the hand region.
* Greyscale\_frames: The original images in greyscale.
* Binary\_frames: Processed original images with true or false if the specific pixel is part of the hand. Hand is white in the images.
* rotated\_scale\_grayscale\_frames: Greyscale frames rotated and scaled, increases the data set.
* Original Videos: the original 1920X1080 videos which the original frames were extracted from.

Only 2 of the listed datasets were used, the original frames and the greyscale frames. The original frames were used for deep model, whereas the greyscale images were used for the SVM and Tree models, The predeveloped greyscale frames from the dataset were selected as there was less noise in the background,Figure 1 .

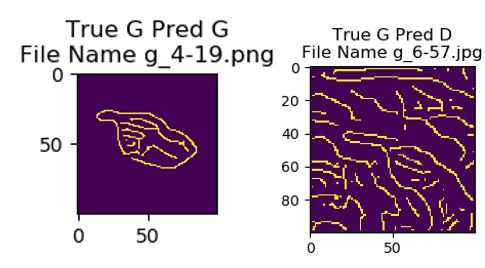


Figure :Right, Predeveloped greyscale image from dataset, Left, self-converted greyscale image. both run through edge detection.

Elliot needs to write this part about which frames you choose and why

The label data for the deep learning model differed slightly than the two other models, the labels were converted to categorial to use classification for the model instead of regression. Classification was selected as it was the clear choice for the dataset and problem. Regression may have worked with decreased results and more training required; this is speculation however as a regression model was not tested. The non-deep learning models both just used an integer ranging between 0 to 24, inclusive. 24 was the number of classes as z and j were absent from the dataset as they were reserved for dynamic gestures.

The greyscale images were selected over the binary frames as it allowed for more features to be extracted from the images whilst using edge detection and a function for the conversion to the binary frames did not need to be made. If the binary images were used for this only the outline of the hand would be captured, whereas using the greyscale images allowed for the outlines of overlapping fingers and marks on the hands to be captured and processed. Due to the non-deep learning models not working well with the large number of pixels, feature extraction was used to reduce the processing required for the non-deep learning models. mention more why they are bad at it. Feature extraction was achieved by using the “skimage.feature.canny” function. This returns an image the same dimensions as the input picture, however the values are 1 and 0s indicating if each pixel is a part of an edge. The three variable inputs, sigma, low threshold and high threshold were experimented with to ensure the output images contained features of the hand and to ensure tight a tight outline of the hand, Figure 1. When passing these images into the non-deep neural networks the results must be flattened to work with the methods, this does not affect the results, however.

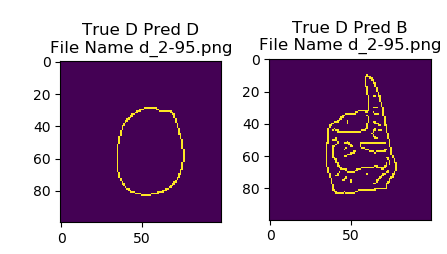


Figure 2: Left, a bad representation of the edge detection on the hand. Right, a good representation of the edges of the hand.

The dataset was split into a training and testing set, the order of the images was randomly shuffled to ensure a split of data when loaded in. With the shuffled list the first 70% of images became the training set whereas the remaining images became the testing set. The ratio of images could be changed with a single input. Figure 2 shows the training and testing class representations, as each class in each set has a decent amount of data for each class, the above splitting method of data could be used.

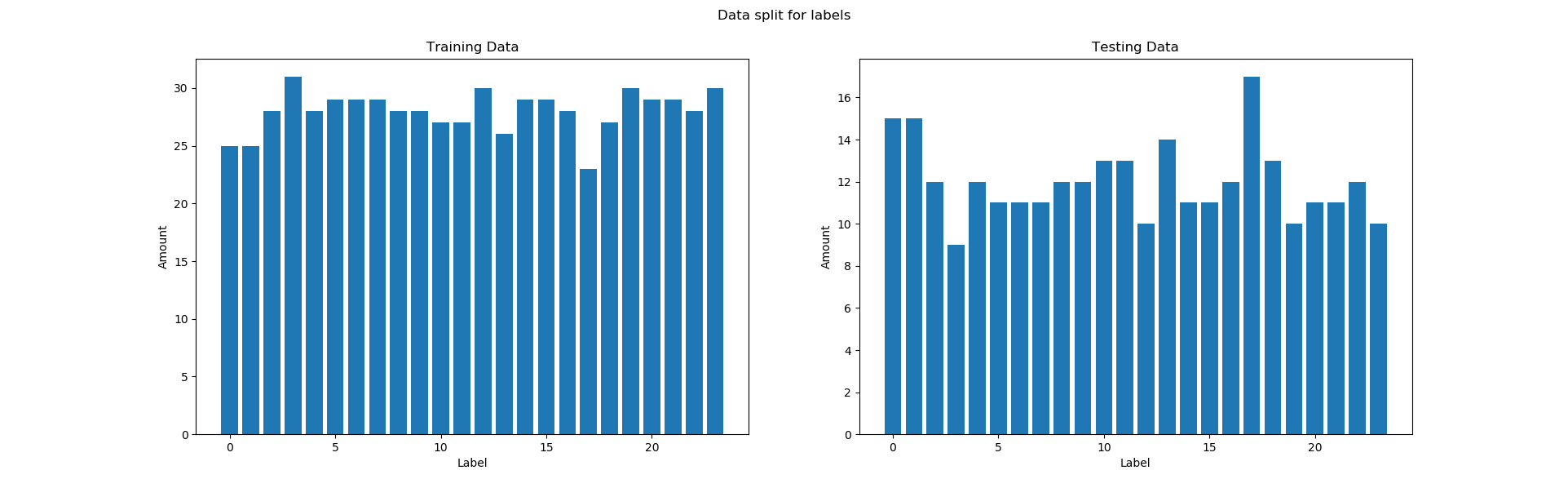


Figure 3: The class representation for the training and testing sets.

***Methodology:***

Clearly explain 2/3 algorithms that you used with citations to the literature. Please note that your project ideally should extend the existing approaches. You dont need to propose a novel algorithm, but you might be looking into approaches that have not previously investigated on your dataset. Note also that your three approaches should be somewhat different. For example, rather than simply using three feed-forward neural networks for a classification task, you could perhaps use (depending on the task) one feed-forward network, one GAN, and a non-deep learning method.

***Deep:***

Test with edge detected images

***Decision Tree:***

The tree method was arguably the easiest method to implement and was the model with the shortest initialisation and training time. This in turn resulted in the model producing the worst accuracy of the 3 models. The model received the edge detection images, these were selected over the RGB or greyscale images due to the reasons stated above.

The specific model was a random forest classifier, this model produces better accuracies than a single tree due to the number of trees used in the forest. The classifier works by using multiple trees which have little to no similarity between each other, these trees are then all executed to produce a prediction for a class. The class with the highest number of predictions is the overall prediction for the entire forest model. This method works well as many models predicting will on average produce the correct result, as any errors from a single tree will be averaged out, Figure 3. This is assuming the data and features given to the model are appropriate.



Figure 4: Random forest classifier

The random forest classifier has a range of parameters that can be adjusted to improve the accuracy of the predictions. A range of values and parameters were selected to estimate the optimal model, Figure 5. Gridsearch was used to find the optimal values in the list, gridserch works by testing every value from the list and finding the best performing model based on accuracy, the model performs the training and testing using k-fold validation. As this tuning was just an estimate, slightly more accuracy may be possible. The tuned model ended up consisting of the following hyperparameters, {'bootstrap': True, 'max\_depth': 20, 'max\_features': 6, 'min\_samples\_leaf': 3, 'min\_samples\_split': 8, 'n\_estimators': 1000}. This model received an accuracy of 47%, whereas the untuned model only achieved an accuracy of 13%, showing the implementation of tuning the model can greatly affect the model.

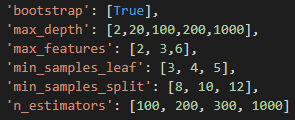


Figure : Random forest hyperparameter optimisation values.

***SVM:***

The SVM was the second fastest to initialise and train, with a large majority of the Decision tree code working for the SVM. This again resulted in the 2nd worst accuracy, sitting firmly in between the Decision tree and the deep network. Again, the model receives the edge detected images, the same as the decision tree.

A range of setting were used to estimate the appropriate parameters for the model, Figure 4. These settings were used in grid search to find the optimal values. Best found tuned parameters from the given list was the following, {'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}. Without the tuned parameters the model was achieving an approximate accuracy of 41%, after the tuning the accuracy increased to 50%, showing the increase in accuracy that can achieved by simply tuning the model.



Figure 6: SVM hyper parameters.

The SVM model was selected as it provided a decent midpoint between the decision tree and the deep learning model. The model still requires the feature detection of the decision tree but allows for a much higher accuracy. The higher accuracy produced by the model can be attributed to the SVMs ability to linearly separate data that would not usually be linearly separated. This is achieved by representing the data in a higher dimension where a linear separation line can be drawn, Figure 4. This allows for the model to produce nonlinear boundaries in the original dimensional space, an advantage over the decision tree model, Figure 5.

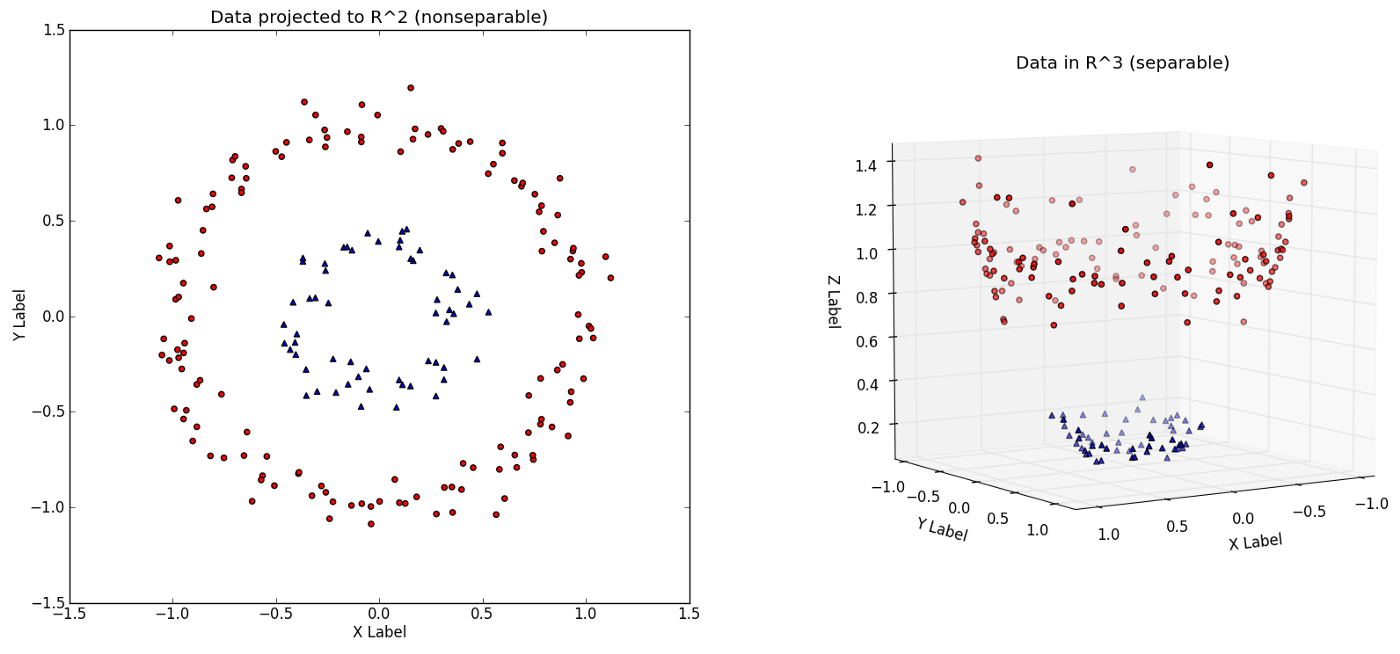


Figure 7: SVM Higher dimension representation.

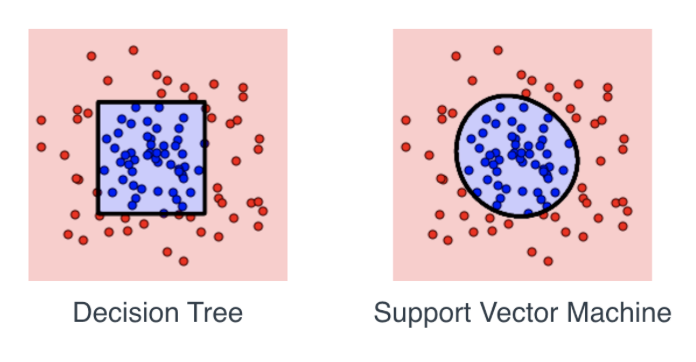


Figure 8: Decision tree vs SVM decision boundary

***Evaluation and Discussion:***

Present the results of all your approaches clearly and compare them with existing approaches. Discuss why your methods are working better/worse than the existing approaches.

***Conclusions and Future Works:***

Clearly explain if the experiments match the objectives, the advantages/shortcomings of the proposed approached, and if any changes are required/ plans you have for the future investigations.