5/15/2018

COMPUTATIONAL INTELLIGENCE CA - 1

SUBMITTED TO

Dr. Zhu Fangming

INSITITUTE OF SYSTEMS SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

PREPARED BY

BALAGOPAL UNNIKRISHNAN (A0178398E)

KENNETH RITHVIK(A0178448M)

MADAN KUMAR MANJUNATH(A0178237W)

RIA VIJAY NAGPAL (A0178257R)

MASTER OF TECHNOLOGY IN KNOWLEDGE ENGINEERING

BATCH KE-30 (2018)

Table of Contents

1. Executive Summary

1.1. Objective

The objective of this report is to understand, analyze and implement different types of neural network and their architectures by solving the two-main important machine learning tasks of classification and regression.

1.2. Scope

A multi class image data set is selected for building an image classification model and for regression task, a relative location of the CT slices is being prediction. Ensemble of the neural network models in each task is performed to get the best fit model.

2. Project Plan

We will be following the CRISP-DM Model for building the different neural network models for this project.

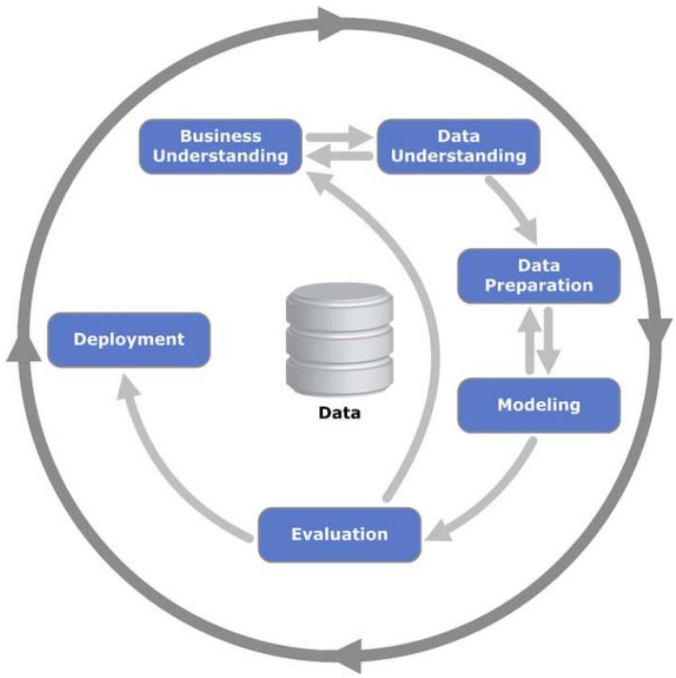

Figure 1 – CRISP-DM methodology

The following steps have been carried out to discover knowledge from the data:

- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation

3. IMAGE CLASSIFICATION MODEL

3.1. COLLECT INITIAL DATA

The source of this image classification dataset is from Shopee-IET Machine Learning Competition[1] from Kaggle website. The initial training/test sets provided by Shopee, were classified into 18 different categories. Taking the project timelines into consideration we selected random 5 categories from the lot and proceeded.

3.2. DESCRIBE DATA

The data set consists of multiple images belonging to various categories of clothing. The five clothing categories which was considered for building our model are:

1) Baby Hat
2) Baby Pant
3) Baby Shirt
4) Women Casual Shoes
5) Women Long Sleeves Top

3.3. EXPLORE DATA

Before proceeding to build model, we output some random images from each of the selected category and got the rough idea of the images from the data set. Also, we checked the frequency of images in each category and made sure that the classes are balanced for both train and test data.

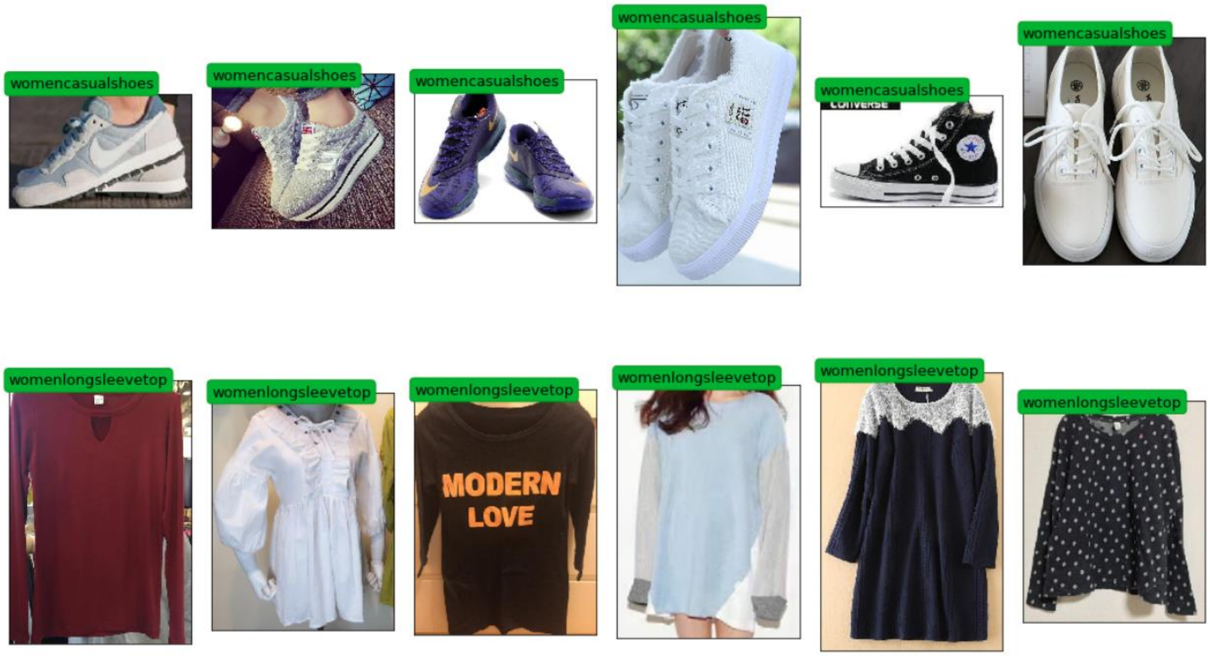

Figure 2 – EDA on Image Dataset

3.4. Data Pre-processing

The images in both train and test set were normalized from a scale of 0-255 to a scale of 0-1. Also augmented the data by

1) Horizontally flipping
2) Zooming
3) Shifting the width and height range

the images to increase the quantity of images for training our model.

3.5. Modelling

We used different architectures of neural networks for building the classification model.

3.5.1. Convolutional Neural Networks (CNN)

Keras framework which is a high level deep learning framework that supports TensorFlow backend, was chosen to build the CNN model. It was chosen because it is easy to prototype using keras and provides many callbacks and API for pretrained models.

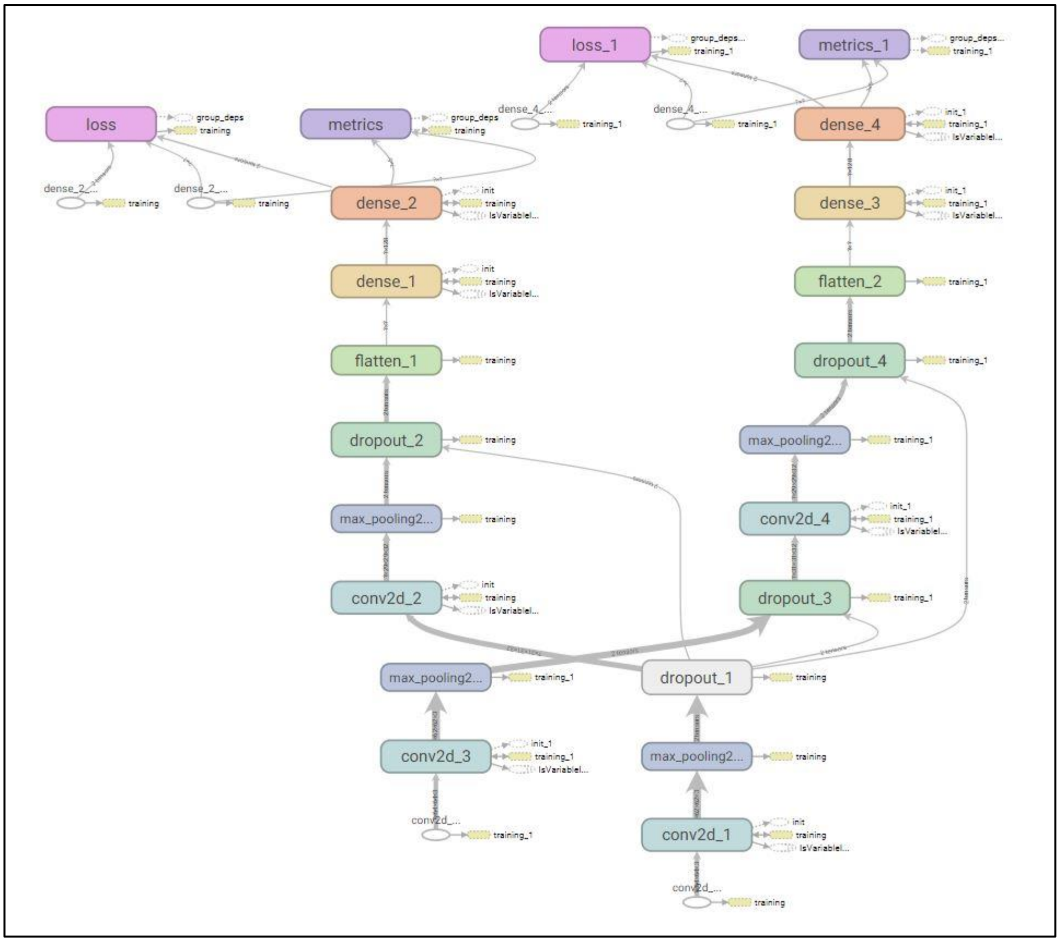

Figure 3 – Structural Design of CNN

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 62, 62, 32)        896
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 31, 31, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 31, 31, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 29, 29, 32)        9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 14, 14, 32)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 6272)              0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               802944
_________________________________________________________________
dense_2 (Dense)              (None, 5)                 645
=================================================================
Total params: 813,733
Trainable params: 813,733
Non-trainable params: 0
_________________________________________________________________
```

Figure 4 – CNN Model Summary

Convolution 2d Layer – Accepts a 2d input image of 64X64 dimension. There are 32 filter feature neurons present in this layer and the dimensions of the feature image is 3x3, the activation function used is ReLu.
Pooling Layer – Receives input from the convolution 2d Layer and performs maxpooling to down scaling of the input image with the help of 2X2 filter.
Dropout Layer – Randomly sets 20% of the total input set values to 0.
Convolution 2d Layer – Receives a 2d input image from the dropout layer. There are 32 filter feature neurons present in this layer and the dimensions of the feature image is 3x3, the activation function used is ReLu.
Pooling Layer – Receives input from the Convolution 2d Layer and performs down scaling of the input image with the help of 2X2 filter.
Dropout Layer – Randomly sets 20% of the total input set values to 0.
Flatten Layer – Performs conversion of the 2d input images into 1d feature set values needed for the next fully connected layers.
Dense Layer – Added a couple of dense layers. The first one is fully connected layer consisting of 128 neurons and ReLu as the activation function. The second dense layer is a **f**ully connected layer consisting of 5 neurons each representing a clothing category with the activation function used being SoftMax.

3.5.2. Residual Network (ResNet50)

The ResNet is an artificial neural network (ANN) of a kind that builds on constructs known from pyramidal cells in the cerebral cortex. ResNet does this by utilizing skip connections to jump over some layers.

The ResNet model summary is as below
Flatten Layer – Performs conversion of 2D input image to a 1D feature space data.
Dense Layer – It is a fully connected layer comprising of 512 neurons based on the ReLu activation function.
Dropout Layer – Used to set the value of 50% of the input set values to 0 to avoid overfitting of the model.
Dense Layer – Consists of 512 neurons fully connected and uses ReLu for its activation function.
Dropout Layer – 50% of the current input set values are assigned a zero value.
Dense Output Layer – The output layer is a fully connected layer consists of 5 neurons having softmax activation function.

3.5.3. Capsule Neural Network (CapsNet)

CapsNet a machine learning system is a type of ANN which adds a structure called capsule to the CNN and reuses the output from multiple capsules to form a more stable representation of higher order capsules.

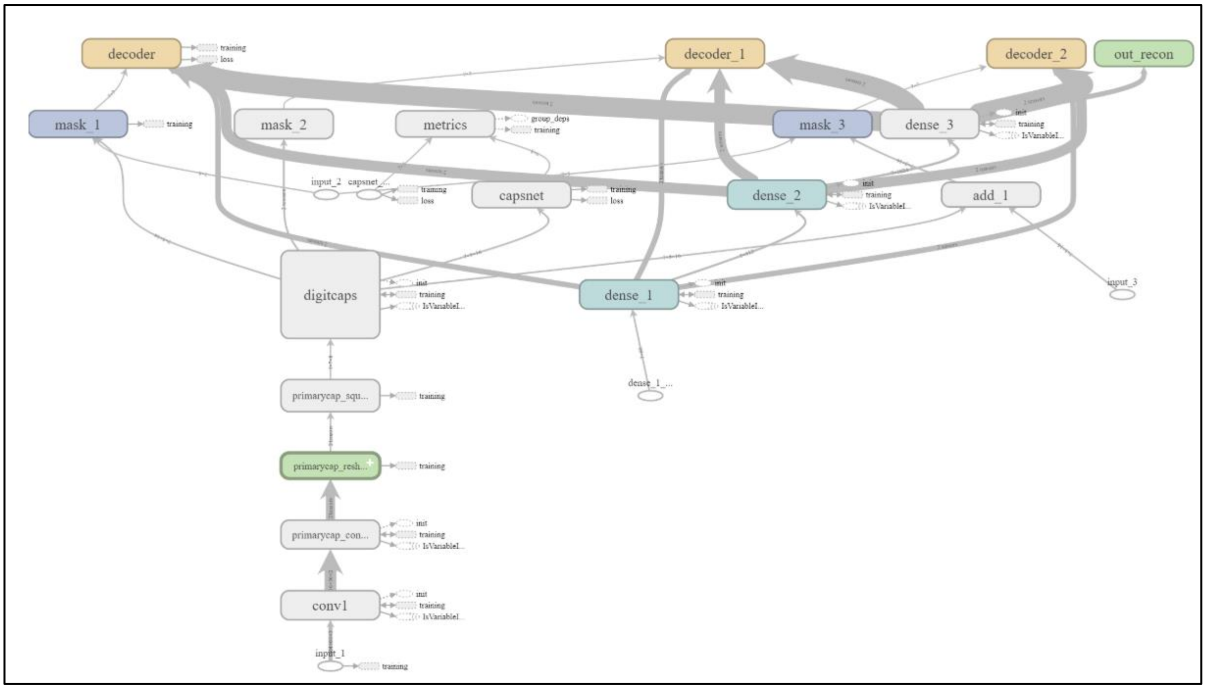

Figure 5 – Structure of CapsNet Model

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 64, 64, 3)	0	
conv1 (Conv2D)	(None, 56, 56, 256)	62464	input_1[0][0]
primarycap_conv2d (Conv2D)	(None, 24, 24, 256)	5308672	conv1[0][0]
primarycap_reshape (Reshape)	(None, 18432, 8)	0	primarycap_conv2d[0][0]
primarycap_squash (Lambda)	(None, 18432, 8)	0	primarycap_reshape[0][0]
digitcaps (CapsuleLayer)	(None, 5, 16)	11796480	primarycap_squash[0][0]
input_2 (InputLayer)	(None, 5)	0	
mask_1 (Mask)	(None, 80)	0	digitcaps[0][0] input_2[0][0]
capsnet (Length)	(None, 5)	0	digitcaps[0][0]
decoder (Sequential)	(None, 64, 64, 3)	13161984	mask_1[0][0]

Total params: 30,329,600
Trainable params: 30,329,600
Non-trainable params: 0

Figure 6– CapsNet Model Summary

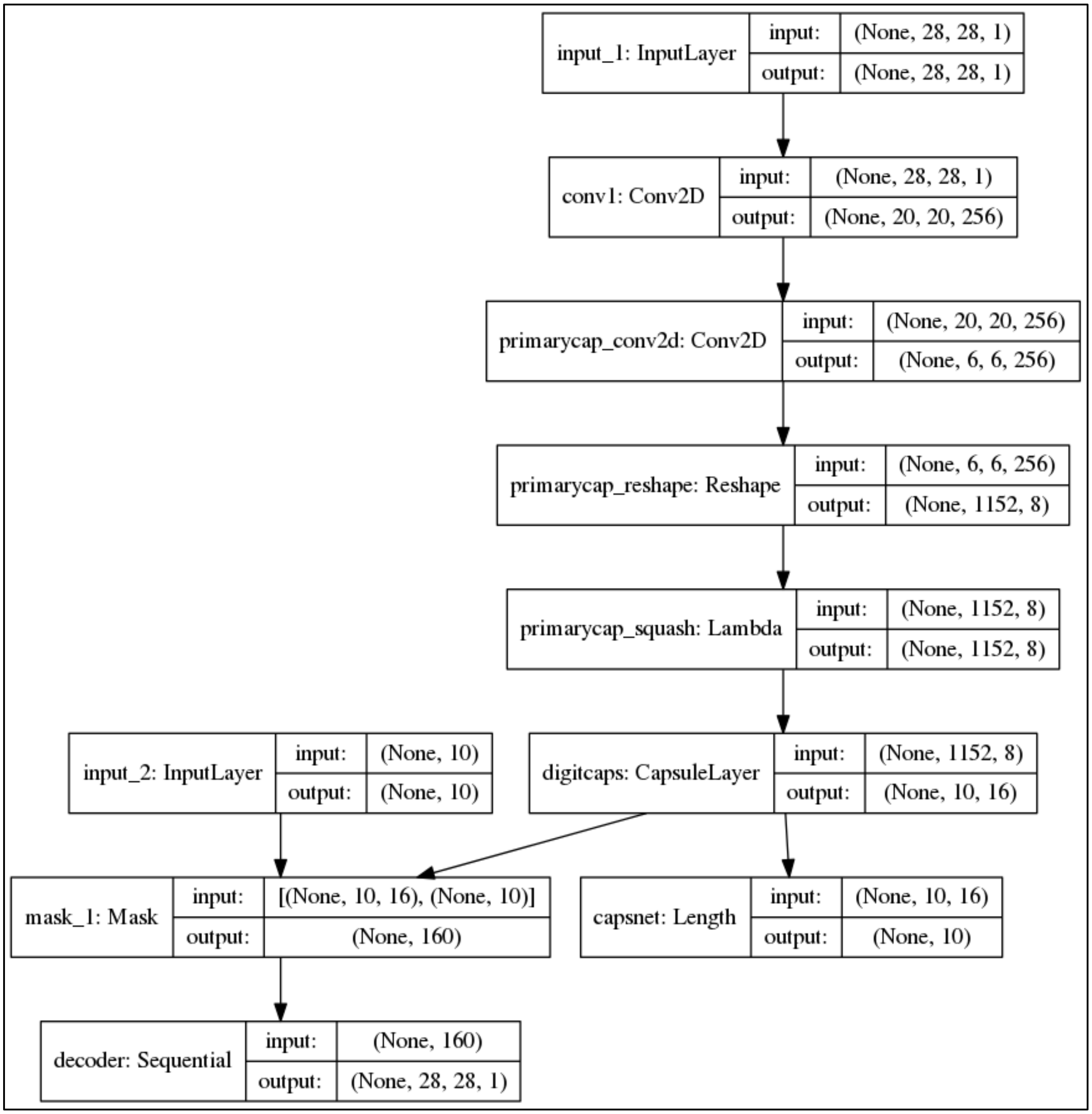

Figure 7 – CapsNet model layers

3.5.4. Classification Ensemble Model

Ensemble modelling methods are hybrid machine learning algorithms that combine two or more different machine learning models to produce a more accurate result that is better than all the other individual models. This approach combines the predictive power of individual models and are frequently placed at the top of Kaggle machine learning competitions.

Ensemble modelling methods can be used to decrease variance (bagging) and bias (boosting) as well as improve predictions (stacking).

We have performed two types of ensemble modelling for this problem:

1) **Majority Vote Ensemble Model**: This is one of the simplest techniques of ensemble modelling where the mode of the predictions of each model is used as the final prediction of the ensemble.

2) **Mean Ensemble Model**: This is another simple ensembling method where we take the average of the prediction probabilities for each class in a single data row and use the maximum value of averaged probabilities to predict the output class.

3.6. Evaluation

3.6.1. Convolutional Neural Networks (CNN)

The confusion matrix is used to evaluate the performance of the classification model on as set of validation data for which the true results are known. The accuracy for the designed CNN model is 86.25%.

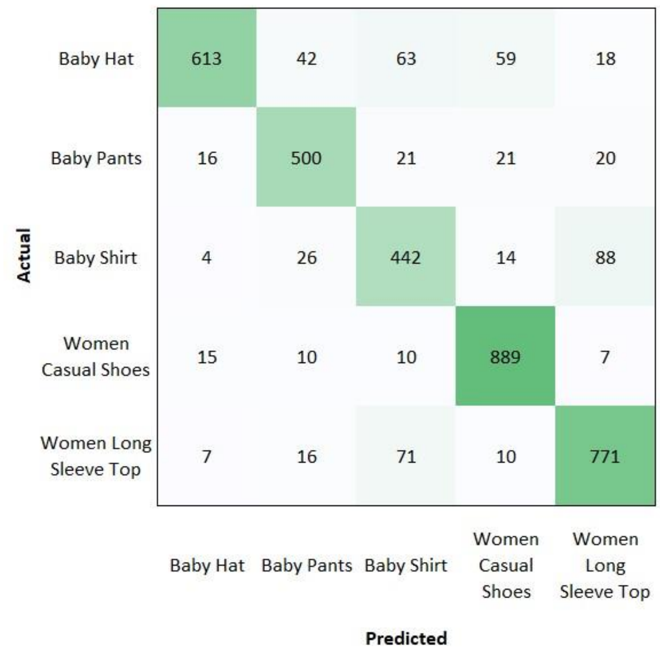

Figure 8 – CNN Confusion Matrix

It is observer from the confusion matrix that majority of the items are correctly predicted. The major misclassification among the rest can be with

- Baby Shirt misclassified as Women Long Sleeve Top
- Women Long Sleeve misclassified as Baby Shirt

Parameters Category	Precision	Recall	F1-Score	Support
Baby Hat	0.94	0.77	0.85	795
Baby Pants	0.84	0.87	0.85	578
Baby Shirt	0.73	0.77	0.75	574
Women Casual Shoes	0.90	0.95	0.92	931
Women Long Sleeve Top	0.85	0.88	0.87	875
Avg/Total	0.86	0.86	0.86	3753

Table 1– CNN Evaluation Matrix Table

The logs generated while building the model was loaded onto the tensor board. The below plots were generated from there.

- Training Accuracy and Validation Accuracy:

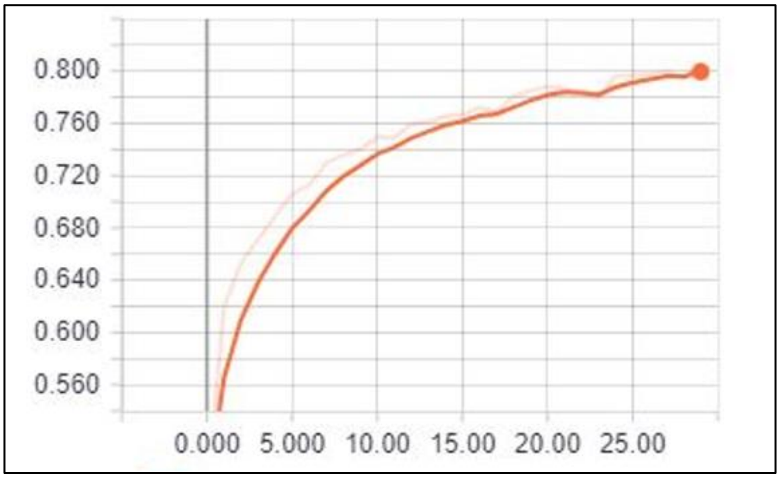

Figure 9 – Training Accuracy

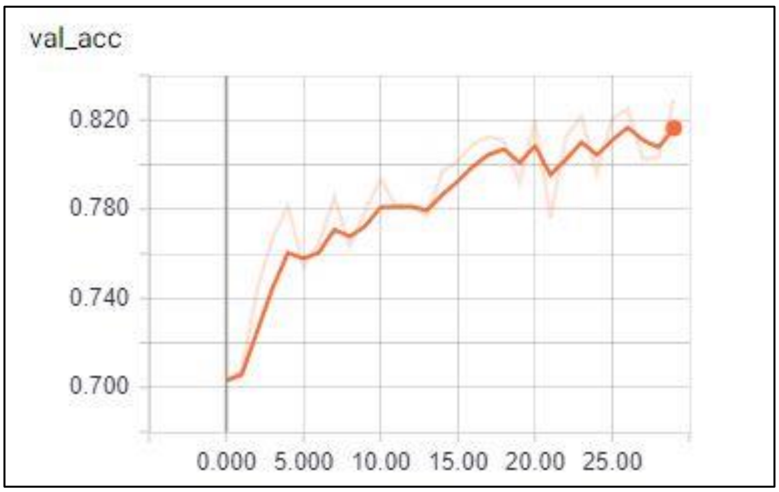

Figure 10 – Validation Accuracy

- Training Loss and Validation Loss:

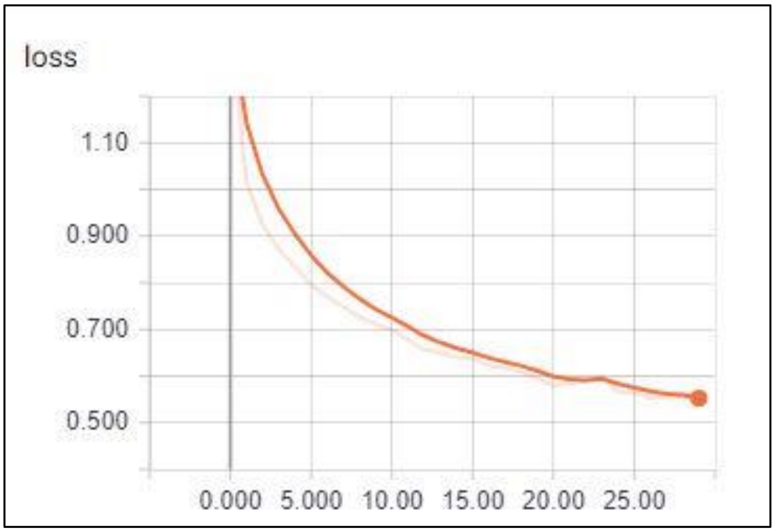

Figure 11 - Training Loss

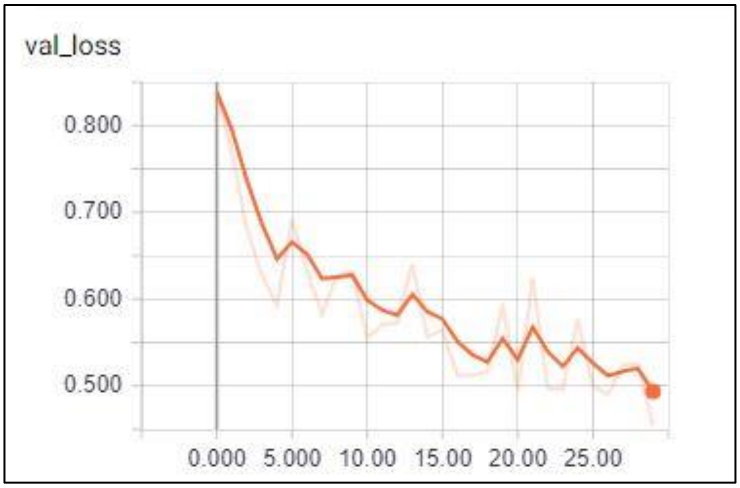

Figure 12 - Validation Loss

Initially during training there is a huge increase in the accuracy which is expected. However, accuracy starts to increment in small amounts just after few epochs. This is usual characteristics while training a neural network when it tries to find the local minima in the multi-dimensional vector space.

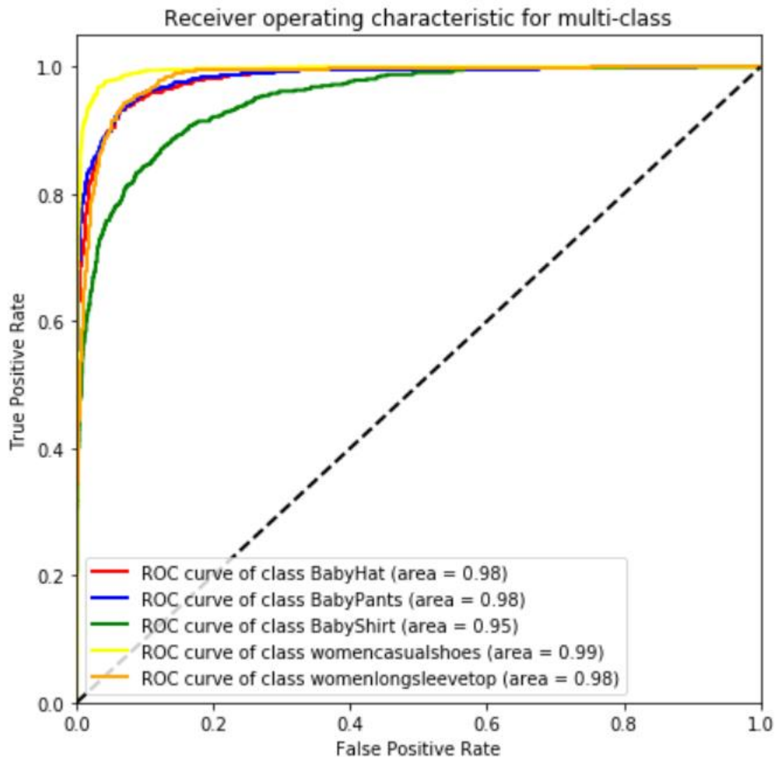

Figure 13 – ROC for CNN Model

From this ROC curve it is seen that the data points are in the upper left side of the plot which shows that our model is performing optimally. This also indicates that there is a high true positive rate and a low false positive rate.

3.6.2. Residual Network (ResNet50)

The accuracy for the designed ResNet model is <u>96.98%.</u>

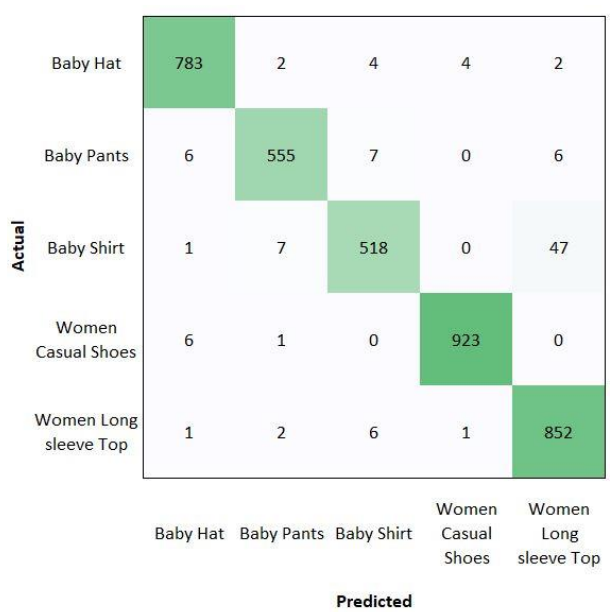

Figure 14 – Confusion Matrix for ResNet Model

From the confusion matrix of the ResNet model it is observed that majority of the classifications are correct. The misclassification is comparatively more in classifying a baby shirt.

Parameters	Precision	Recall	F1-Score	Support
Category				
Baby Hat	0.98	0.98	0.98	795
Baby Pants	0.98	0.97	0.97	574
Baby Shirt	0.95	0.90	0.93	573
Women Casual Shoes	0.99	0.99	0.99	930
Women Long Sleeve Top	0.94	0.98	0.96	872
Avg/Total	0.97	0.97	0.97	3744

Table 2 – ResNet Evaluation Matrix Table

3.6.3. CAPSULE NETWORK (CAPSNET) [3]

The accuracy for the designed ResNet model is <u>65.41%.</u>

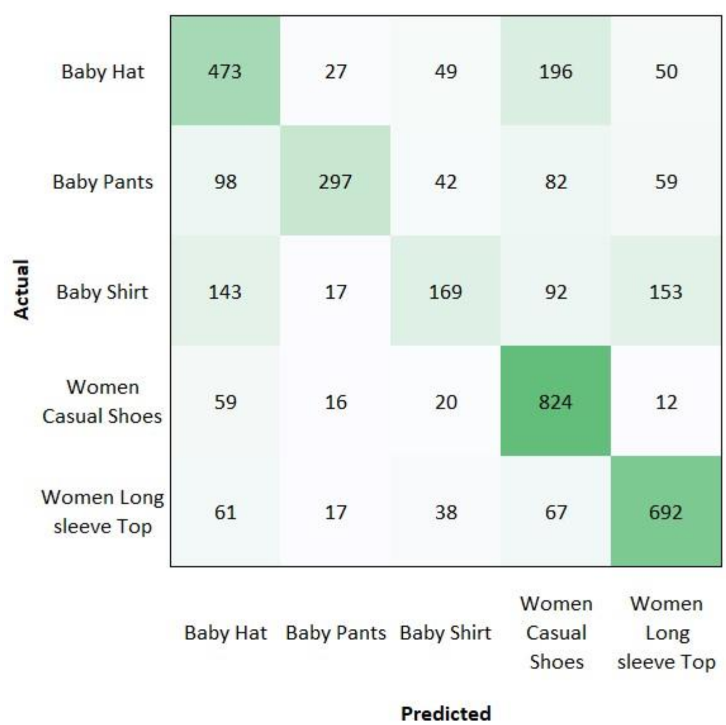

Figure 15 – Confusion Matrix of CapsNet Model

From the confusion matrix of the CapsNet model it is observed that there is a greater percentage of misclassification. The major misclassification among the rest can be with

- Baby Hat is misclassified as Women Casual Shoes
- Baby Shirt is misclassified as Women Long Sleeve Top and Women Long Sleeve Top
- Apart from the above two the misclassification of the images in other classes are also relatively higher.

Parameters Category	Precision	Recall	F1-Score	Support
Baby Hat	0.57	0.59	0.58	795
Baby Pants	0.79	0.51	0.62	578
Baby Shirt	0.53	0.29	0.38	574
Women Casual Shoes	0.65	0.89	0.75	931
Women Long Sleeve Top	0.72	0.79	0.75	875
Avg/Total	0.65	0.65	0.64	3753

Table 3 – CapsNet Evaluation Matrix Table

3.6.4. CLASSIFICATION ENSEMBLE MODEL

Majority Vote Ensemble Model:

When we performed majority vote ensemble modelling using the custom CNN, ResNet pre-trained on imagenet, Xception pre-trained on imagenet and CapsNet we got an accuracy of 96.78%. This is better than the accuracy of any of the individual models. From the confusion matrix we can still see that baby shirts are being predicted as women's long sleeve top as they look similar. But this still improves over the prediction power of any individual model.

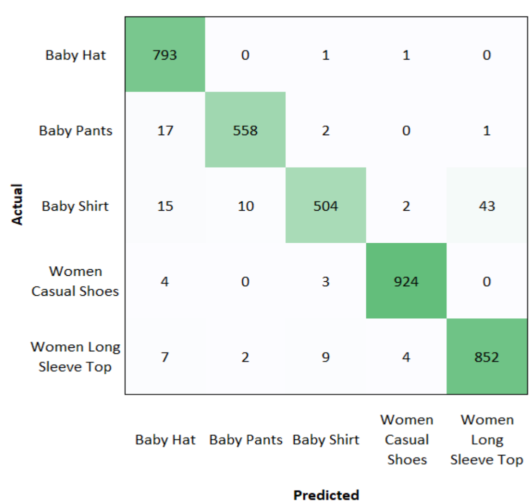

Figure 16 – Majority Vote Confusion Matrix

Parameters Category	Precision	Recall	F1-Score	Support
Baby Hat	0.95	1.00	0.97	795
Baby Pants	0.98	0.97	0.97	578
Baby Shirt	0.97	0.88	0.92	574
Women Casual Shoes	0.99	0.99	0.99	931
Women Long Sleeve Top	0.95	0.97	0.96	874
Avg/Total	0.97	0.97	0.97	3752

Table 4 – Majority Vote Evaluation

Mean Ensemble Model:
Mean ensemble model of the same base models further improves our accuracy to 98.27%. We are also achieving near perfect values of precision, recall and f1-score for all classes. We are also achieving perfect values of precision and sensitivity for predicting women's casual shoes.

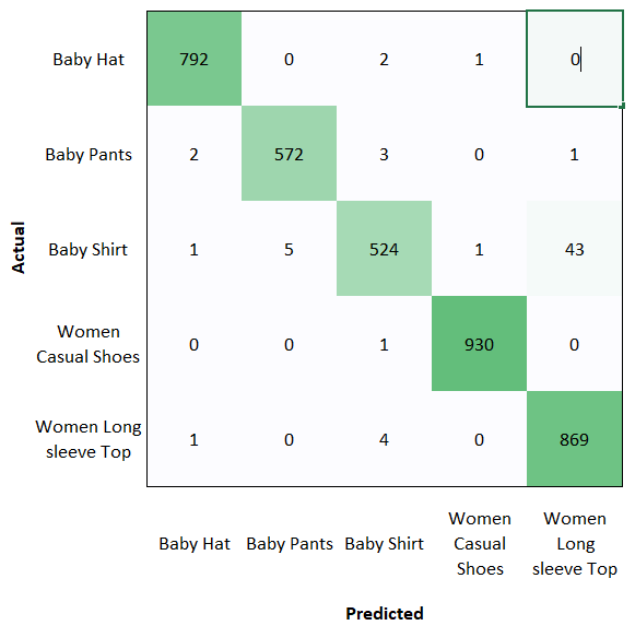

Figure 17 – Mean Ensemble Confusion Matrix

Parameters Category	Precision	Recall	F1-Score	Support
Baby Hat	0.99	1.00	1.00	795
Baby Pants	0.99	0.99	0.99	578
Baby Shirt	0.98	0.91	0.95	574
Women Casual Shoes	1.00	1.00	1.00	931

Women Long Sleeve Top	0.95	0.99	0.97	874
Avg/Total	0.98	0.98	0.98	3752

Table 5 – Mean Ensemble Evaluation Matrix

Model	Accuracy
RESNET50	0.969882729211
XCEPTION	0.945895522388
CAPSNET	0.653251599147
CNN_CUSTOM	0.861407249467
MAJORITY VOTING	0.967750533049
MEAN ENSEMBLE	0.982675906183

Table 6 – Ensemble Results

4. Regression Model

The regression model is built to predict the relative location of the CT slices on axial axis of the human body.

4.1. Collect Initial Data

This regression dataset was found from the UCI Machine Learning Repository[2]. It consists of 384 features extracted from CT images.

4.2. Describe Data

The data was retrieved from a set of 53500 CT images from 74 different patients (43 male, 31 female). Each CT slice is described by two histograms in polar space. The first histogram describes the location of bone structure in the image, the second give the location of air inclusions inside the body. Both the histograms are concatenated to form the final feature vector.

4.3. Explore Data

We analysed the data set and found that there is no missing data. The below table gives the attributes information of the dataset.

Column Number	Description
1	PatientID: Each ID represents different patient
2 - 241	Histogram describing bone structures
242 - 385	Histogram describing air inclusions
386	Reference: Relative location of the image on the human axial axis scale. Values are in the range [0,180] where 0 denotes the top of the head and 180 the sole of the feet.

4.4. Data Pre-processing

A subset with only the histogram features is selected from the raw data set and excluded the 'PatientID' and the 'Reference' variable. Also, we excluded the features in columns 59, 69, 179, 189 and 351 as they turned out to be a column with constant value. The data was pre-normalized.

Dimension Reduction Using PCA

The dimension of the subset is now 379, as it is very large we applied 'Principal Component Analysis (PCA)' as our dimension reduction technique. Post PCA it was found that the 56 principal components were found to have eigen values greater than 1 and it covered 76% of the data.

Train and Test Split

The selected 56 principal components were further used to form the training and testing datasets. It was split in the ratio 75:25 respectively using sampling.

4.5. Modelling

4.5.1. General Regression Neural Network (GRNN)

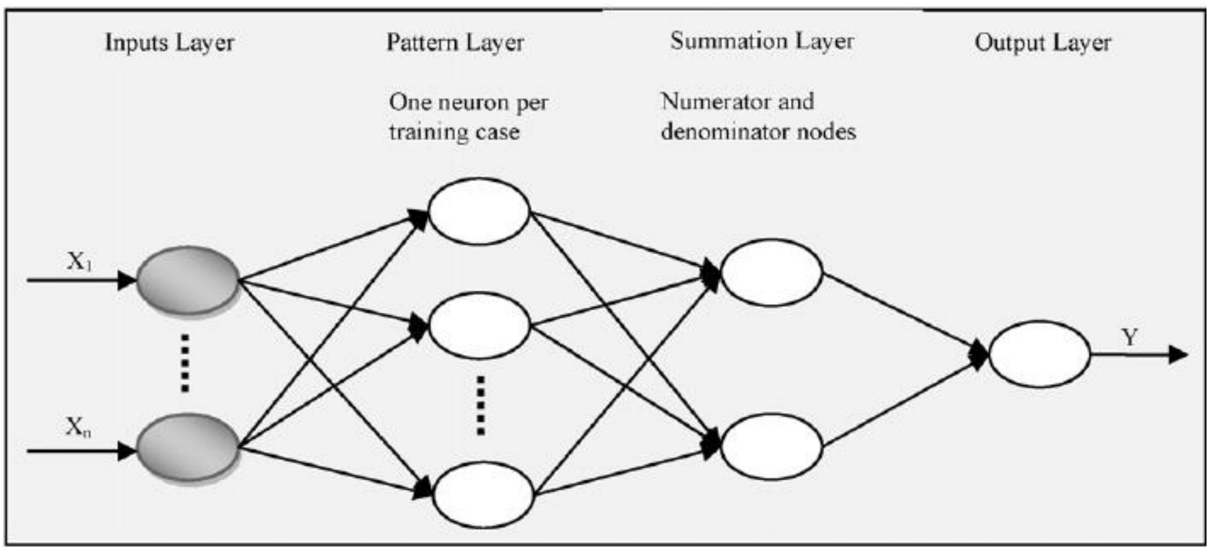

Figure 18 – Basic Structure of GRNN model

- The above diagram shows the basic structure of the GRNN model. The model was built using the GRNN package in R. It implements the algorithm proposed by Specht.
- Using K-Means data was clustered into 6 clusters. The number 6 was chosen because there was an approximate even distribution across clusters. This also determines the number of pattern nodes in the network.
- Considering the trade-off between narrow peaked and smoother surfaces the smoothing constant sigma was set to 0.5.
- Post tuning the hyperparameters the model was built on the training and the testing data.

4.5.2. MULTI-LAYER PERCEPTRON (MLP)

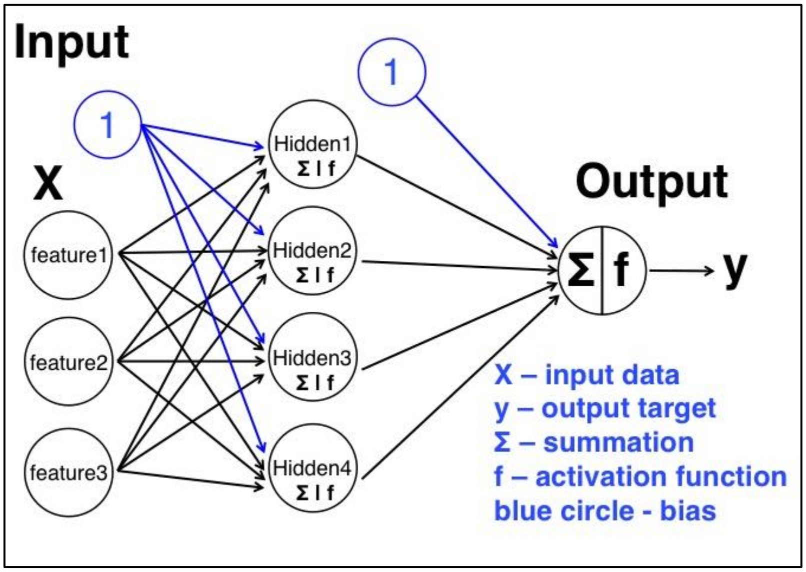

Figure 19 – Basic structure of MLP

- The ‘MLP’ function of RSNNS package in R is used for building and studying the architecture of multilayer perceptron model. MLP’s are fully connected feedforward networks.
- Using K-Means data was clustered into 6 clusters. The number 6 was chosen because there was an approximate even distribution across clusters. This also determines the number of input nodes in the network.
- The training input and its corresponding target values are the main input parameters to this function.
- The other hyperparameters which was tuned are
  1) Number of hidden units: 10
  2) Number of iterations to learn: 1000
  3) Initialization Function: Randomize_Weights is used within the intervals (-0.1, 0.1). All weights and bias are initiated by this function with distributed random values.
  4) Learning Function: Standard Back Propagation is used with a learning rate of 0.2. The learning rate specifies the gradient descent step width.
  5) Activation function for all hidden units: Act_Logistic, this applies the sigmoid function to the weighted sums.
  6) LinOut: It is set to ‘True’ so that the identity function is set as the activation function.

4.5.3. Radial Basis Function (RBF)

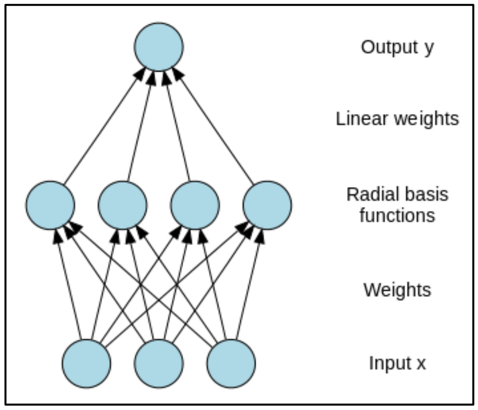

Figure 20 – Basic Structure of RBF

- The 'RBF' function of RSNNS package in R is used for building and studying the architecture of radial basis function network. It performs linear combination of 'n' basis functions that are symmetric radially around a prototype.
- The training input and its corresponding target values are the main input parameters to this function.
- Using K-Means data was clustered into 6 clusters. The number 6 was chosen because there was an approximate even distribution across clusters. This also determines the number of input nodes in the network.
- The other hyperparameters which was tuned are
  1) Number of hidden layers: 20
  2) Number of iterations to learn: 1000
  3) Initialization Function: RBF_Weights is used within the input and hidden layers (0, 1, 0, 0.01, 0.01).
  4) Learning Function: Radial Basis Learning is used with learning parameters set to (1e-8, 0, 1e-8, 0.1, 0.8).
  5) LinOut: It is set to 'True' so that the identity function is set as the activation function.

4.5.4. Regression Ensemble Model

Mean Ensembling

Takes the mean of the predicted outputs by each model.

Stacking

In stacking we predict the output of the training data using the model with which it was trained. We then use all the predicted outputs of each model as training data for another supervised machine learning algorithm and use the original target variable values as the target variable for this algorithm. In our problem we used simple MLP as the supervised machine learning algorithm. We then supply with the predicted output of the test data of all the individual models and this MLP gives us the ensembled output.

4.6. EVALUATION

4.6.1. GENERAL REGRESSION NEURAL NETWORK (GRNN)

The Root Mean Square Error (RMSE) and R-Squared metrics are used to evaluate a regression model.

Metrics	Value
RMSE	0.1999268
R-Squared	0.9999203

The below plot is got by plotting the predicted against the actual values of the GRNN model. It also reads that there is a high accuracy of the predicted values.

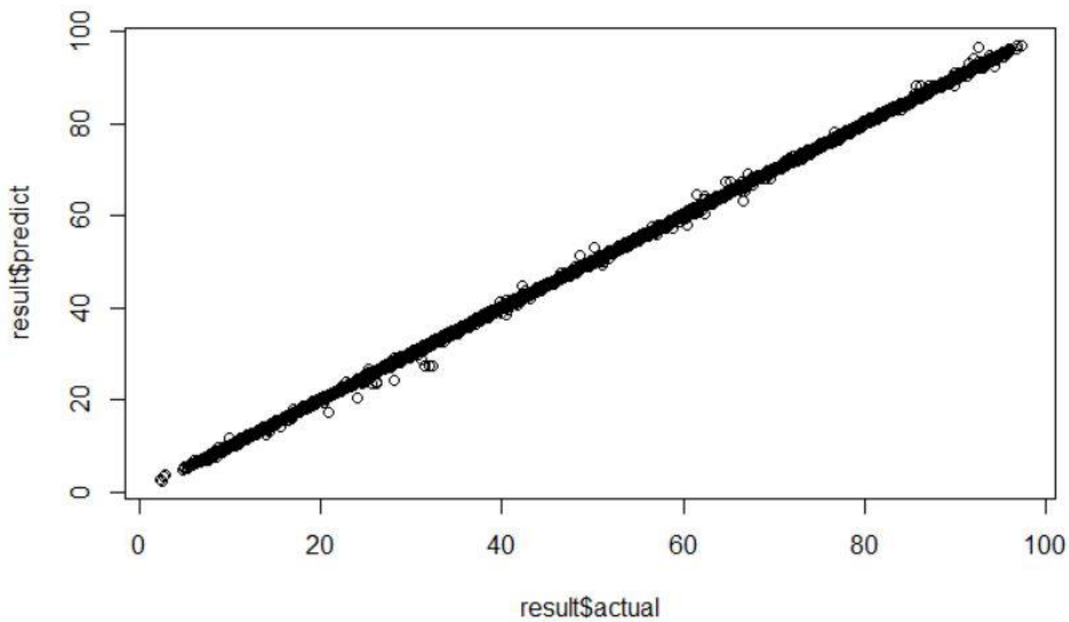

Figure 21 – Actual vs Predicted

4.6.2. MULTI-LAYER PERCEPTRON (MLP)

The Root Mean Square Error (RMSE) and R-Squared metrics are used to evaluate a regression model.

Metrics	Value
RMSE	0.0103210967873
R-Squared	0.808796324494

The below plots show the iterative training and test error of the MLP neural network model.

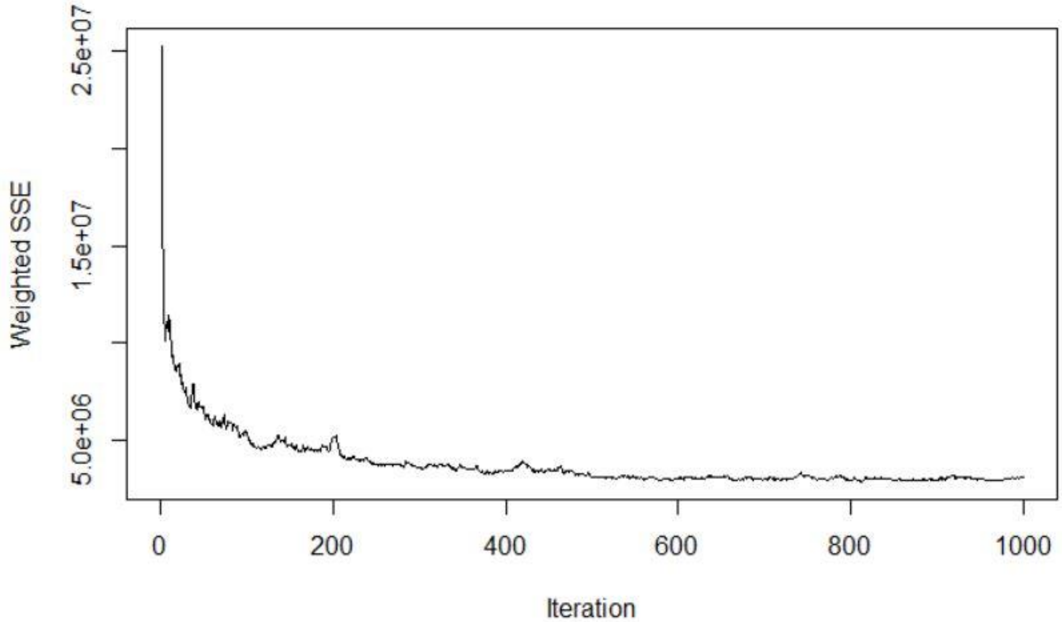

Figure 22 – Iterative Error Plot for MLP

4.6.3. Radial Basis Function (RBF)

The Root Mean Square Error (RMSE) and R-Squared metrics are used to evaluate a regression model.

Metrics	Value
RMSE	0.0106966198534
R-Squared	0.801839564767

The below plots show the iterative training and test error of the RBF neural network model.

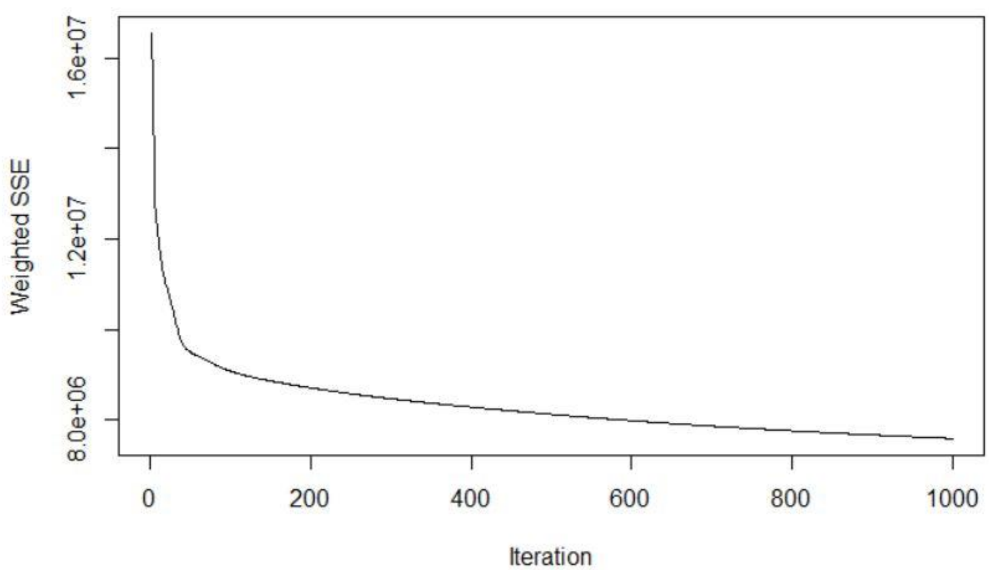

Figure 23 – Iterative Error Plot for RBF

4.6.4. Regression Ensemble Model

Stacking ensemble model proved to be better than the mean ensemble in this problem.

Model	MSE	R2_Score
MLP	0.0103210967873	0.808796324494
RBF	0.0106966198534	0.801839564767
MEAN ENSEMBLE	0.00845186510476	0.843424811702
STACKING ENSEMBLE	0.00729725486593	0.864814565717

Table 7 – Regression Ensemble Results

5. Tools Used

To build the above mentioned neural network models we have made use of the following software and packages:

- R
  - GRNN
  - RSNNS
  - hydroGOF
- Python
  - TensorFlow
  - Keras
  - Scikit Learn

6. Acknowledgement

We would like to thank Kaggle and UCI for publishing their data on their website.

7. Conclusion

In this assignment, we have analysed and used convolutional neural network, residual neural network, capsule neural network and ensemble neural network to classify clothing images into 5 classes from IET-Shopee challenge on Kaggle website. On the other hand, we also used generalize neural network, multilayer perceptron neural network and radial basis function neural network to perform regression to predict the relative location of the CT slices. We conclude that

1) Mean ensemble has the best model performance with an accuracy of 98.27% accuracy on the clothing image classification dataset.
2) Stacking ensemble has the best model performance in predicting the relative location of the CT slices with an R2 score of 0.86.

8. References

1. Kaggle Competition: https://www.kaggle.com/c/shopee-iet-machine-learning-competition

2. UCI Data Repository: https://archive.ics.uci.edu/ml/datasets/Relative+location+of+CT+slices+on+axial+axis
3. Dynamic routing between capsules: https://arxiv.org/pdf/1710.09829.pdf