CM4107 CW Part1: Comparative Study

**Darie-Dragos Mitoiu**

School of Computing, RGU, Aberdeen, UK

*Abstract*—This paper contains an analysis of the accuracy and error rate produced by three standard machine learning algorithms when considering classification as the method of problem solving for a hand-writing dataset, these algorithms are an Artificial Neural Network (ANN) model, a K-Nearest Neighbour (kNN) model and a combination or hybrid model of the two mentioned models (ANN and kNN). The data that will be analysed in this comparative study it is a hand-writing data called “mnist”, this analysis has the goal to recognize the hand writing elements present in the dataset by making use of the algorithms in cause, the recognition will be done via a classification approach.

# Comparative Study setup

The aim or ultimate goal of this comparative study is to evaluate the accuracy and error rate produced by three machine learning algorithms, these algorithms are the Artificial Neural Network (ANN), the K-Nearest Neighbour and the last but not the least a hybrid algorithm created from the initial Artificial Neural Network algorithm and the K-Nearest Neighbour algorithm.

The dataset that has been chosen for this study it is a hand-writing dataset called mnist, this dataset it is presented in two forms, the first form of the dataset it is a training type of dataset and the second form it is a testing type of dataset. During the data exploration process of the mnist training and testing datasets, it has been concluded that the mnist training dataset presents 60000 rows or records and 785 columns or features while the mnist testing dataset presents 10000 rows or records and 785 columns or features. Both the training and testing mnist datasets do not present any columns or feature names, the content present in the datasets being numerical values representing the graphical representation of specific characters, the characters in cause require a row conversion to a 28x28 matrix in order to be read, in this particular case, the characters being numerical values from 0 to 9. The mnist training and testing data sets were presented in a ready state for analysis with minimal pre-processing operations required as the data was already split in training and testing datasets. Before the analysis of the mnist datasets, some pre-processing steps were performed, these steps were the minimization of the training and testing datasets to a number of 1500 records for the training dataset and to a number of 100 records for the testing dataset, this was done in order to reduce the processing time required in order to train the models in cause. The Hyper-Parameters chosen for this study are the number of epochs, the batch size and the learning rate. The epoch hyper-parameter values used in this comparative study are the following numerical values: 10, 50, 100, 150, 200. The batch size hyper-parameter values used in this comparative study are the following: 10, 50, 100, 200, 500, 1500. The learning rate hyper-parameter values used in this comparative study are the following: 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0.

In this comparative study, the three machine learning algorithms covered in this study (Artificial Neural Network, K-Nearest Neighbour, Artificial Neural Network – K-Nearest Neighbour Hybrid) will be evaluated using the mnist dataset by interchanging one hyper-parameter per session, this behaviour will mean that when a hyper-parameter of the three selected for this study (epochs, batch size, learning rate) the other two hyper-parameters will remain static.

# Neural Network

## Neural Network Hyperparameters

## This comparative study will focus on three neural network hyper-parameters, these hyper-parameters are the number of epochs, the batch size number and the learning rate number of the neural network. The number of epochs defines or represents the training iterations performed on the entire mnist dataset, the batch size number defines or represents the number of samples to work through in a single epoch before the internal network parameters are updated and last but not the least the learning rate or the step size number defines or represents the amount of weight adjustment during the training of the neural network. The number of epochs it is associated with how well the neural network will perform on a specific dataset, increasing the number of epochs of a neural network it could lead to an increase of the model’s accuracy and a reduction of the error rate which would lead to a minimization of inaccurate results, but this is not always the case, as the number of epochs it is increased to much, this it could lead to inaccurate results due to overfitting the model, the overfitting will also mean that the model will most likely provide inaccurate results for other data that may be encountered other than the training dataset. The batch size it is associated with the number of epochs where a single epoch makes use of the batch data when training the neural network. The learning rate ranges between values of 0.0 and 1.0, a high learning rate may cause a divergent behaviour of the model but this is not always the cause, in most cases a high learning rate may produce good results, while a low learning rate will prevent the training process to progress in an acceptable manner.

## Visualisation of Results

Create a graph to visualize the impact of each hyperparameter on training. You can generate your graph either with Excel or programmatically with matplotlib.

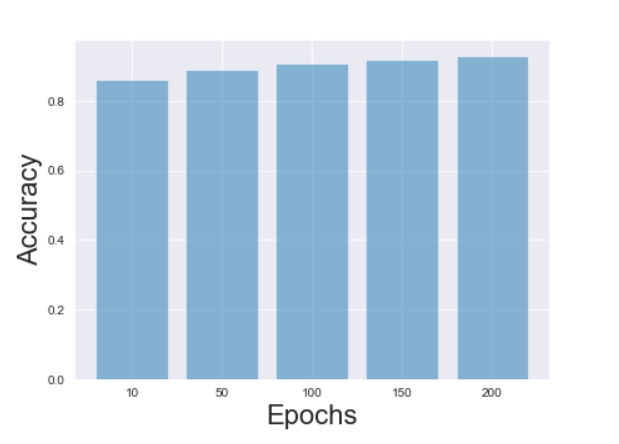


Fig. 1. Artificial Neural Network Accuracy based on number of Epochs

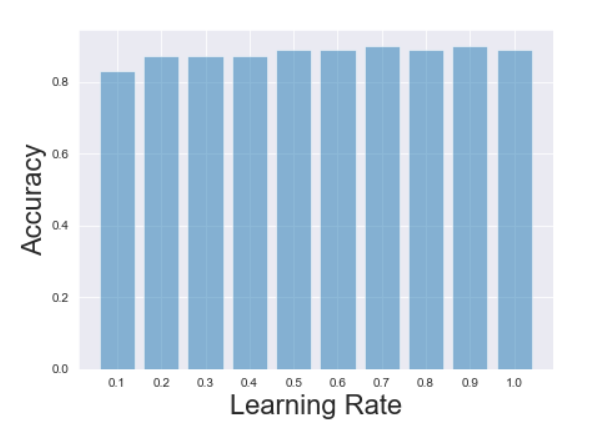


Fig. 2. Artificial Neural Network Accuracy based on Learning Rate

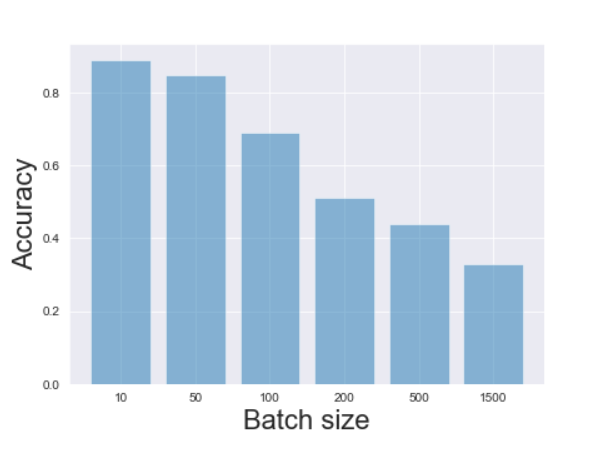


Fig. 3. Artificial Neural Network Accuracy based on Batch Size

## Discussion of Results

The results of the Artificial Neural Network (ANN) model presented on the left side show the performance of the artificial neural network represented based on the accuracy of the model and not the mean squared error rate. The training of the model was performed using 1500 records of the msnist dataset and the test of the model was done using 100 mnist records from the mnist testing dataset. The hyper-parameter called epochs used for this comparative study was used in a series approach by making use of multiple instances of the Artificial Neural Network (ANN), each instance having its unique epoch value, in this comparative study, the following epochs values were used: 10, 50, 100, 150, 200. As it can be seen in the Fig. 1, a low number of epochs will result in an influence over the model’s performance, the influence being a poor one, decreasing the model’s accuracy and increasing the model’s error rate, in this case the low value of epochs it is 10, which presents the lowest accuracy value, the accuracy value being 0.86. We can see that an increase in the number of epochs can lead to a higher accuracy value for the model and a decreased error rate, in this case, the highest epochs value being 200, the use of this epochs value will result in an accuracy value of 0.93, which it is the highest value from the five values compared in this study, nevertheless, the increase in the accuracy is not substantial comparing to a signification lower value of epochs which is the epochs value of 100, this epochs value will result in an accuracy of 0.91, which is a relatively close accuracy rate to the accuracy produced by the double of this value. The hyper-parameter called learning rate or step size used for this comparative study, just as the epochs hyper-parameter was used in series approach by making use of multiple values of learning rates, this learning rates values being: 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 and 1.0. As it can be seen in the Fig. 2, a low number of learning rate or step size will result in low accuracy for the Artificial Neural Network (ANN), the lowest learning rate being 0.1 in this specific case, the accuracy produce by the 0.1 learning rate was 0.84, which it is the lowest of the ten values compared, the highest accuracy rate produced was 0.90, this accuracy rate being produced by making use of the learning rate 0.7, this means that an increase in the learning rate or step size will result in an increase of the model’s accuracy. The hyper-parameter called batch size used in this comparative study, similar to the previous hyper-parameters (epochs and learning rate) was used in a series approach, the values analysed in this comparative study being: 10, 50, 100, 200, 500 and 1500. As it can be seen in the Fig. 3, a low batch size value will result in a higher accuracy for the Artificial Neural Network, while a high batch size value will result in a decrease of the model’s accuracy and an increase in the model’s error rate, as the batch size value increase the accuracy decreases. The comparative study was performed using the three hyper-parameters mentioned as interchangeable values, when the epochs were tested, the batch size value was equal to 20 and the learning rate equal to 0.3, while when the learning rate was tested, the epochs value was equal to 20 and the learning rate remained at the value of 0.3 when the epochs and batch size were tested.

# k-nearest neighbour

## k-NN Hyperparameters

The machine learning algorithm called the k-Nearest Neighbour it is a supervised machine learning algorithm that can be used for both classification and regression problems, the classification approach being the most common for this algorithm, when referring to a “supervised machine learning algorithm” it means that the algorithm requires a class label for the input data in order to provide predictions on the testing data once the training of the model it is completed using the class label in cause. The K-Nearest Neighbour algorithm has the role to estimate the number of data points for a specific instance in order to allow the categorisation of the instance in cause to a specific group of instances, in other words, similar instances are close to each other forming a specific group, this being the logic behind the K-Nearest Neighbour machine learning algorithm. The hyper-parameter k used by the k-Nearest Neighbour algorithm refers to the number of instances that establish a specific group of neighbour instances, where an instance can be added by making use of the unweighted or weighted voting process. An example that could describe the role of the hyper-parameter called k of the k-Nearest Neighbour algorithm would be in the Fig. 4, where the value of k equal to 10 represents the number of neighbour where a specific instance could be added to based on the voting process, if the instance would pass the voting process, meaning the neighbours will present similar characteristics as the instance in cause, it will be added to the specific group of instances, once this is done the group would present 11 instances instead of 10. In a k-Nearest Neighbour with unweighted voting, the process of voting for a potential new candidate instance will be done based on the class majority in the specific group of instances. In a k-Nearest Neighbour with a weighted voting, the process of voting will be done via the distance between the potential new candidate instance and the group of instances and the similarity of the features between the candidate instance and the group of instances.

## Visualisation of Results

## Provide a single graph, to compare weighted and unweighted kNN at different values of k on your datasets.

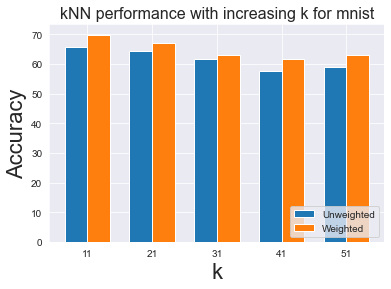


Fig. 4. K-Nearest Neighbour Accuracy based on the number of k

## Discussion of Results

The results of the K-Nearest Neighbour (kNN) model presented on the left side show the performance of the K-Nearest Neighbour represented based on the accuracy of the model and not the mean squared error rate. The training of the model was performed using 327 records of the msnist dataset and the test of the model was done using 73 mnist records from the mnist testing dataset, the testing data being not present in the training process, the model has been tested on data that it was not seen before by the model. The hyper-parameter called k represents the number of instances that create a specific group, all the stances in the group having the same class label. As it can be seen in the Fig. 4, as the number of k or the number of instances that establish a specific class label group increases, the accuracy of the K-Nearest Neighbour (kNN) decreases and the error rate of the model increases for both unweighted and weighted voting process. It is clear that as the number of k increases the accuracy of model decreases for both unweighted and weighted voting process, nevertheless, this decrease it is not a significant one for the five values of k analysed in the Fig. 4, an exception being the last value of k = 51, where the decrease of the accuracy it is easily notable, the decrease being in the interval of 10-15%, from k = 11, where the accuracy it is 70% for weighted voting, the k = 51 presents an accuracy of approx. 60% for the weighted voting process. As it can be seen in Fig. 4, there is a relatively small difference in accuracy produced by the unweighted and weighted voting process, the weighted voting having the best performance in all five cases, an exception being the k = 31, where the difference it is almost inexistent.

# Adding new training data

## Strategy for using rotated data

In order to create the new training data from the mnist dataset by making use of the rotation function provided for the Artificial Neural Network (ANN) Model and the K-Nearest Neighbour (KNN) Model, two data manipulation approaches have been performed on the mnist training dataset, the Artificial Neural Network (ANN) model requiring its own approach and the K-Nearest Neighbour model respectively in order to satisfy the data input requirements for each algorithm. The approach taken for the Artificial Neural Network model was to load the training mnist dataset as a data frame containing only 3500 records from the mnist training comma separated file (csv) which is significantly larger than 3500 records, once the data has been loaded, a degree of rotation of 35 has been defined, an array list which has the role to contain the new images once rotated has been also defined as an empty array list, once the degree and new data array list container have been created, an iteration was performed on the mnist training data frame, for each iteration on the training data frame the class label of each record has been retained in a variable called “label”, the class label in the mnist training dataset it is the first element in each row, so at index position 0, after the class label has been retained, the rest of the content for each row (without the class label) in the mnist training dataset have been also retained in two variables after the rotation function has been applied on the record without the label, these two variables presenting the instances returned from the rotate image function, which it is a function that will return rotated images clockwise and anti-clockwise to a specific degree, after the rotation has been performed and the new instances have been returned, the addition of the class label needs to be added to the record in order to perform the training of the models later on, this was done using the numpy insert function, once the insertion of the class label has been performed, the clockwise and anti-clockwise rotated images have been added to the new training data array. The approach taken for the K-Nearest Neighbour model was very similar to the approach taken for the Artificial Neural Network (ANN), the only difference being the conversion of the new training data array to a numpy array in order to allow the separation of the class label and the content of each record in the newly created training data, the conversion was necessary in order to allow the K-Nearest Neighbour to perform the training process.

## Results with added training data

The results of the Artificial Neural Network (ANN) model presented on the bottom side show the performance of the artificial neural network represented based on the accuracy of the model and not the mean squared error rate. The training of the model was performed using 3500 records of the rotated msnist dataset and the test of the model was done using 100 mnist records from the mnist testing dataset. The hyper-parameter called epochs was used for this comparative study in a series approach by making use of multiple instances of the Artificial Neural Network (ANN), each instance having its unique epoch value, in this comparative study, the following epochs values were used: 10, 50, 100, 150, 200, these epoch being associated with the optimal value of learning rate being equal to 1.0 and the optimal batch size which was equal to 10. As it can be seen in the Fig. 5, using a significantly larger amount of training data compared to the initial Artificial Neural Network (ANN) model with the mentioned epochs values and the optimal learning rate and batch size hyper-parameters did not present any improvement in the accuracy of the model for none of the five epochs values compared, instead a decrease in the accuracy of the model of approx.. 10% it is present for all five epochs values, this decrease in the accuracy may be caused by the rotation degree used for the newly created training data, the degree of rotation used was equal to 35, a reduction of this value may result in an increase of the model’s accuracy, while an increase in the rotation degree may result in the decrease of the accuracy and in an increase of the error rate.

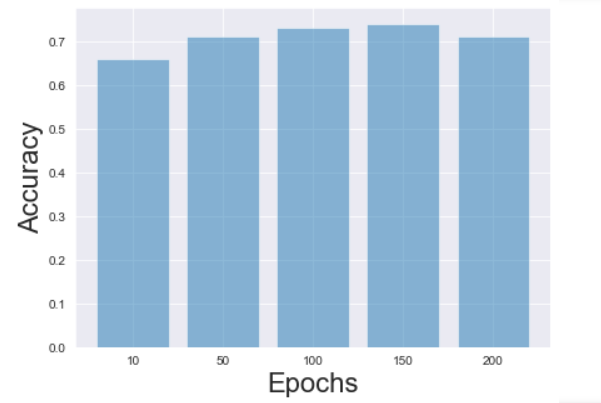


Fig. 5. Artificial Neural Network using rotated training data

The results of the K-Nearest Neighbour (kNN) model presented on the bottom side show the performance of the K-Nearest Neighbour using the rotated training mnist dataset represented based on the accuracy of the model and not the mean squared error rate. The training of the model was performed using 500 records of the rotated training msnist dataset and the test of the model was done using 100 regular mnist records from the mnist testing dataset, the testing data being not present in the training process, the model has been tested on data that it was not seen before. The hyper-parameter called k was used for this comparative study in a series approach by making use of multiple instances of the K-Nearest Neighbour (kNN), each instance having its unique k value, in this comparative study, the following k values were used: 11, 21, 31, 41, 51, these k values being associated with the Euclidean way of calculating the distance between the K-Nearest Neighbour instances. As it can be seen in the Fig. 6, using a larger amount of training data compared to the initial K-Nearest neighbour (kNN) model with the mentioned k values and the Euclidean calculation method did not present any improvement in the accuracy of the model for none of the five k values compared, instead a decrease in the accuracy of the model of approx. 10-15% it is present for all five k values, this decrease in the accuracy may be caused by the rotation degree used for the newly created training data, the degree of rotation used was equal to 35, a reduction of this value may result in an increase of the model’s accuracy, while an increase in the rotation degree may result in the decrease of the accuracy and in an increase of the error rate.

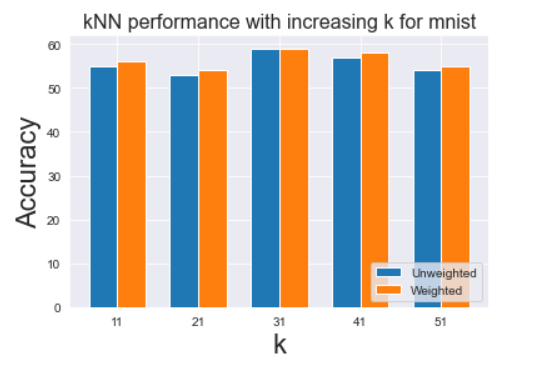


Fig. 6. K-Nearest Neighbour using rotated training data

# Adding your own handwriting

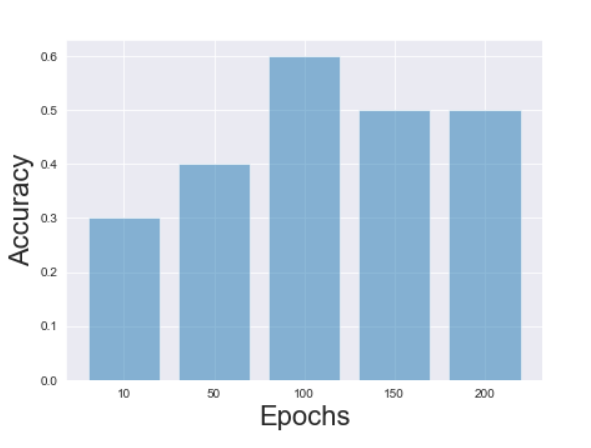
## Creating my own handwriting charachters

In order to create the handwriting images, the piece of software called Adobe Photoshop was used, this software it is a commercial product but it can be used as a trial product, an alternative to this software would be the Windows Application called Paint or the third-party software called GIMP, using these programs would allow the same result when considering the creation of handwriting numbers as 28x28 PNG files. The first step in the creation of the handwriting images was to create a blank image with a size of 500x500, this size was selected in order to allow the drawing of the numbers in a clear manner, as 28x28 it is a small resolution, it would be difficult to perform any drawing and to be accurate on this small resolution.

The second step in the creation of the handwriting numbers was the drawing of the number using the tool provided by the Adobe Photoshop called “Brush Tool”, this tool was used with a drawing size of 25, this size allowing the readability of the drawing once the image would be scaled down, this tool was used over the Pen Tool also provided by the Adobe Photoshop as the Brush Tool allows more versatility when considering the drawing line size. The third step in creation of the handwriting numbers was to draw the numbers using the brush tool and the fourth step and the last step was to scale down the image from a 500x500 size to a 28x28 by making use of the “Image Size” feature of Adobe photoshop, once this was done the image was saved as a png file in the folder called “handwriting”. Once the handwriting images have been created, the images have been converted into the proper format for the Artificial Neural Network by making use of the functions called “get\_my\_test\_data” and “map\_target\_to\_output\_layer”, the first function was used in order to load the images and the second function was used to convert the images to the adequate format. Before replacing the mnist test data for the test method of the Artificial Neural Network model, the utility function called “mean\_squared\_error” have been modified in order to calculate the length of the new handwriting test data by changing the use of the size attribute used before to using the build-in function called “len” in order to calculate the length of the test data array. After these steps have been performed the newly test data have been given to the testing method of the Artificial Neural Network and the testing has been performed successfully.

## Testing on my own handwriting

Some of the observations that must be mentioned after the testing process was completed would be that there was noticeable decrease in the accuracy of the model and an increase in the error rate produced by the model. Using the hyper-parameter called epochs for the evaluation of the model, five epochs values were used as in the first Artificial Neural Network, associated with a batch size equal to 20 and a learning rate equal to 0.3, exactly the same values are in the first Artificial Neural Network as it can be seen in the Fig. 1. As it can be seen in the Fig 7 there is a 10-20% decrease in the model’s accuracy comparing to Fig. 1.



# Hybrid

## Combining ANN and k-NN

Using the optimal parameters from previous parts of the coursework; develop a hybrid system which improves data representation. The idea is that you make use of the hidden layer activation for each training instance and use that as input into the kNN.

## Visualisation of Results

A table should be provided, comparing accuracy of the hybrid system against the kNN and neural network. This will demonstrate that the improvement in representation over the course of training is due to improved representation as gained from the network. You can present the results for each of your datasets using a table (or graph). You can test with and without the new data added in section IV.

## Discussion of Results

##### References

Provide references in the following format:

1. Jason Brownlee (2018, July 20). *Difference Between a Batch and an Epoch in a Neural Network* [Online]. Avaiable: <https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/>
2. Jason Brownlee (2016, April 15). *K-Nearest Neighbour* [Online]. Avaiable: <https://machinelearningmastery.com/k-nearest-neighbors-for-machine-learning/>
3. Jason Brownlee (2019, January 25). *Understand the Impact of Learning Rate on Neural Network Performance* [Online]. Avaiable: <https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/>
4. Jason Brownlee (2020, August 21). *How to Control the Stability of Training Neural Networks With the Batch Size* [Online]. Avaiable: <https://machinelearningmastery.com/how-to-control-the-speed-and-stability-of-training-neural-networks-with-gradient-descent-batch-size/>
5. Onel Harrison (2018, Sep. 10). *Machine Learning Basics with the K-Nearest Neighbors Algorithm* [Online]. Available: <https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761>
6. Wiratunga, N., 2020. Artificial Neural Nets (ANNs), CM4107 [Powerpoint Presentation]. Artificial Neural Nets (ANNs). Advanced Artificial Intelligence. Robert Gordon University. School of Computing. 15 October. Available: [http://campusmoodle.rgu.ac.uk/mod/resource  
   /view.php?id=3687426](http://campusmoodle.rgu.ac.uk/mod/resource/view.php?id=3687426)
7. Wiratunga, N., 2020. K Nearest Neighbour, CM4107 [Powerpoint Presentation]. K Nearest Neighbour. Advanced Artificial Intelligence. Robert Gordon University. School of Computing. 29 October. Available: <http://campusmoodle.rgu.ac.uk/mod/resource/view.php?id=3687437>
8. Wiratunga, N., 2020, ANN\_2020. Aberdeen. Robert Gordon University
9. Wiratunga, N., 2020, kNN. Aberdeen. Robert Gordon University
10. G. Eason, B. Noble, and I.N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529-551, April 1955. (*references*)

Fig. 7. Arttificial Neural Network using custom handwriting testing data