**Group 6 Report                                                                                                      7/22/2019**

**Group member NetID**

FangJia Li            Sagar Moozhikulam Tze Yang Chen                Akib Hossain

Fxl160430           sbm170130          txc172930           axh174930

**Introduction and problem description**

         This report will be focusing on reflecting what we have learned on machine learning onto this project. Regarding this project, we will differentiate layers on the neural network and observe its impact on the results. In other words, change the number of layers and numbers of different units per layer while trying to optimize the results.

**Related work**

In class, we have practice a similar practice regarding metrics learning. We will be implementing that method onto this dataset. According to the article (Beniwal, 2018), also provided an example and a clearer algorithm using Neural Network for digit recognition. Dr. Koehler (Koehler, 2018) also have a similar work on a related topic which we have taken influence from.

**Dataset**

         This dataset consists of two data files: test.csv and train.csv. Each data file contains hand-drawn image of digits from zero to nine. Each image is a combination of 28\*28 metrics. Each metric has a pixel-value associated with it. The goal is to predict which number (0~9) the metric is representing.

The training data set has 785 columns. The first column is the digit that the user drew, and the rest of the columns are the pixel-values of the image. The testing data only has the columns that contain the pixel-values of the image.

There are a total of 42,000 28\*28 training data, and a total of 28,000 28\*28 testing data. However, only the training data will be used for obtaining data.

The better the accuracy the better.  However, we expect 90 % correction rate to be considered “successful”

**Pre-processing techniques**

Since the dataset has been separated into training data and testing data. However, 10-fold validation will be used to determine the average accuracy. A prediction table (.csv file) will also be produced. Python 3 will be the language used in this project, with keras, pandas, and numpy library used. Furthermore, we use jupyter and anaconda for our coding environment.

**Proposed solution and methods**

Algorithm provided in the article (Beniwal, 2018) will be implemented in this project.

1.     Weight Initialization

2.     Forward Propagation

3.     Cost Computation

4.     Backwards Propagation

5.     Minimizing Cost-Weight function

         This Project will be testing different models to see which is the optimal of all.

 However, since we are using pre-existing libraries, we will focus mainly on the different combinations of layers and units. We have 3 different models which have the same total number of ‘units’. Model A, 3 layers with 128 units per layer. Model B, 6 layers with 64 units per layer. Model C, 12 layers with 32 units per layer. We are expecting Model C to have the best learning outcome (highest testing score) since they have more units per layer.

**Results**

The results will be in a table as below.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Model A, 3\*128 | Model B 6\*64 | Model C 12\*32 |
| Test Acc(%) | 96.40 | 92.51 | 80.83 |
| Validation Acc(%) | 96.48 | 95.29 | 85.33 |
| Total time (Sec) | 90 | 91 | 101 |

**Conclusion**

According to our results above, it is clear that only Model A and Model B were able to score over 95 % correct rate even though all three models has the same total number of units. In addition, Model A also spend less time on learning. Thus, meaning Model A has the highest learning-effort ratio. However, Model B gives similar results as well. Meanwhile, Model score way less with 85% Accuracy.

Initially,We suspect that the reason behind the contracts from Model C to Model A is due to data overfitting. When adding more layers than it needs, the model extracts too many features from this dataset, which lowers the efficiency and accuracy for this dataset. Later on We learned that the reason for low performance is due to Model C losing gradient, since the training score for Model C is also low.

**Contribution of team members**

Coding: Fangjia Li,Tze Yang Chen

Documentation: Fangjia Li, Tze Yang Chen

Proofread: Sagar Moozhikulam, Akib Hossain

**References**

Beniwal, H. (2018, December 27). Handwritten Digit Recognition using Machine Learning. Retrieved from https://medium.com/@himanshubeniwal/handwritten-digit-recognition-using-machine-learning-ad30562a9b64

Koehler G. (2018, December 27). MNIST Handwritten Digit Recognition in Keras.   
Retrieved from <https://nextjournal.com/gkoehler/digit-recognition-with-keras>