**Convolutional Neural Network in Hand-writing Recognition**

**What is CNN and why do we choose CNN?**

The Convolutional Neural Network is a type of ANN that is primarily used in the field of computer vision. CNN classifies and recognizes images of objects and texts because it demonstrates excellent results for input data feature extraction, classification, and pattern recognition. A CNN consists of a layer of data, a hidden layer, and a layer of output. CNN's secret layer consists of extracting input data features and classifying extracted features (Lee, Park and Sim, 2020).

**Implementation strategies**

1. Data analysis

MNIST database is a training set of 60,000 examples and a test set of 10,000 examples. It is a subset of NIST 's accessible larger package. The digits in a fixed-size picture have been size-normalized and centred. (LeCun, Cortes and Burges, 2020).

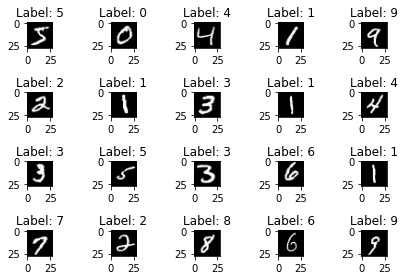
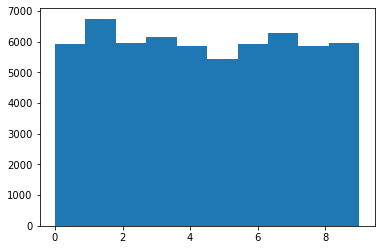


Figure 1 shows the first 20 samples in MNIST dataset

Before splitting the dataset, we have to consider whether MNIST dataset has data imbalance which can result in bad performance of the model. So, I visualize the distribution of each labels in training data shown below. The classes of training data have been evenly distributed which avoided data imbalance.

Fig 2 The histogram of MNIST dataset

1. Data preparation
2. Split MNIST dataset

In this assignment, I imported the dataset by importing Python library named ‘mnist’ from ‘keras.dataset’. Then, I split the dataset into ‘x\_train’, ’y\_train’ and ‘x\_test’, ‘y\_test’, where training set contains 60,000 samples and testing set contains 10,000 samples.

1. Reshape ‘x\_train’ and ‘x\_test’

According to Professor Yann LeCun, the images in MNIST dataset were in a scale of 28X28. However, the data of the image was in one dimension vector containing 784 values. So, I reshaped all images back to 28X28 scale with singe channel since the images were all grey scaled.

1. Normalization

A Neural Network is among the best practices for training to normalize the data to achieve a mean close to 0.0. In general, normalizing the data accelerates learning and leads to faster convergence (Stöttner, 2020). In order to increase the performance, ‘x\_train’ amd ‘x\_test’ were normalized by dividing them with 255, since the range of each pixels was from 0 to 255.

1. Convert the class of the testing set into a binary matrix

This process is to convert the classes of ‘y\_train’ and ‘y\_test’ which are 10 digits number from 0 to 9 to binary matrix, so that the model can avoid a problem that leads the model to assume the higher number of the class has, thee better class is.

To be more specific, the class ‘1’ will be converted into [1,0,0,0,0,0,0,0,0].

1. Building Convolutional Neural Network

In this stage, I will be talking about how I developed the architecture of my CNN model and parameter tuning.

1. The architecture of my CNN model

The lecture slide and other influential research/article provided in reference session have contributed significant effort on inspiring my own CNN architecture. My architecture is mainly developed upon LeNet-5 architecture.

**IN -> Convolutional layer 1-> MaxPool 1-> Convolutional layer 2-> MaxPool 2-> fully connected 1-> fully connected 2-> OUT**

The convolutional layer 1 contains 32 kernels with size 5 by 5 and ‘relu’ as the activation function. Next layer is max pooling layer 1 with size 2 by 2. The convolutional layer 2 contains 64 kernels with size 3 by 3 and activation function ‘tanh’. Max pooling layer 2 is the size of 2 by 2 following with the fully connected layer with 128 neurons, activation function ‘tanh’ and dropout rate 0.3. Fully connected 2 contains 64 neurons, ‘tanh’ activation function and the last one is output layer.

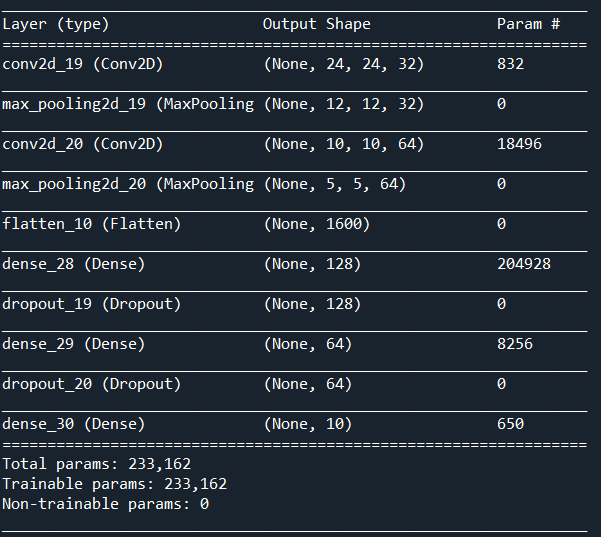


Fig 4: Architecture of the CNN model

1. Fit the model and results

In this stage, I fit the model described above with the training dataset, batch size 64 and epochs 10. After training, the model is validated by testing dataset with accuracy 99.21% on testing dataset which contains 10,000 observations.

1. Parameter Tuning

In this stage, Grid search has been applied on finding the best training model from various activation functions (‘relu’, ’tanh’), batch size (64, 96,128) and epochs (10, 15, 20). Since the size of testing data is 10,000 which is one seventh of the total dataset, the cross-validation splitting strategy is selected as 7 which is same as the training and testing dataset. As the table shown below, after 35-hours searching, the best model highlighted in blue is the case 9 with batch size 96, activation function ‘relu’, epochs 20 and results in 99.14% in Grid search testing set.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case | Batch Size | Activation function | Epochs | Average test accuacy | Standard Deviation |
| 1 | 64 | relu | 10 | 0.9903 | 0.0009 |
| 2 | 64 | relu | 15 | 0.9908 | 0.0013 |
| 3 | 64 | relu | 20 | 0.9906 | 0.0011 |
| 4 | 64 | tanh | 10 | 0.9903 | 0.9908 |
| 5 | 64 | tanh | 15 | 0.991 | 0.0013 |
| 6 | 64 | tanh | 20 | 0.9911 | 0.0016 |
| 7 | 96 | relu | 10 | 0.9908 | 0.0008 |
| 8 | 96 | relu | 15 | 0.9912 | 0.0006 |
| 9 | 96 | relu | 20 | 0.9914 | 0.0011 |
| 10 | 96 | tanh | 10 | 0.9902 | 0.0012 |
| 11 | 96 | tanh | 15 | 0.9907 | 0.0010 |
| 12 | 96 | tanh | 20 | 0.9912 | 0.0012 |
| 13 | 128 | relu | 10 | 0.99 | 0.0008 |
| 14 | 128 | relu | 15 | 0.9913 | 0.0008 |
| 15 | 128 | relu | 20 | 0.9907 | 0.0009 |
| 16 | 128 | tanh | 10 | 0.9892 | 0.0017 |
| 17 | 128 | tanh | 15 | 0.9905 | 0.0011 |
| 18 | 128 | tanh | 20 | 0.9911 | 0.0017 |

1. Evaluation and conclusion

The best model which is the case 9 will be evaluated by accuracy score and confusion matrix. The accuracy score is 99.41% on MNIST validation dataset, while the model with the original parameters only delivers accuracy 99.21%.

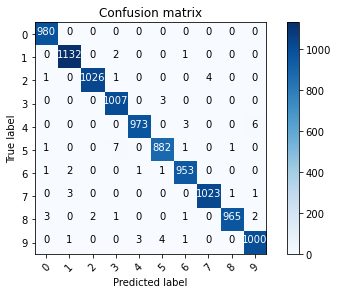
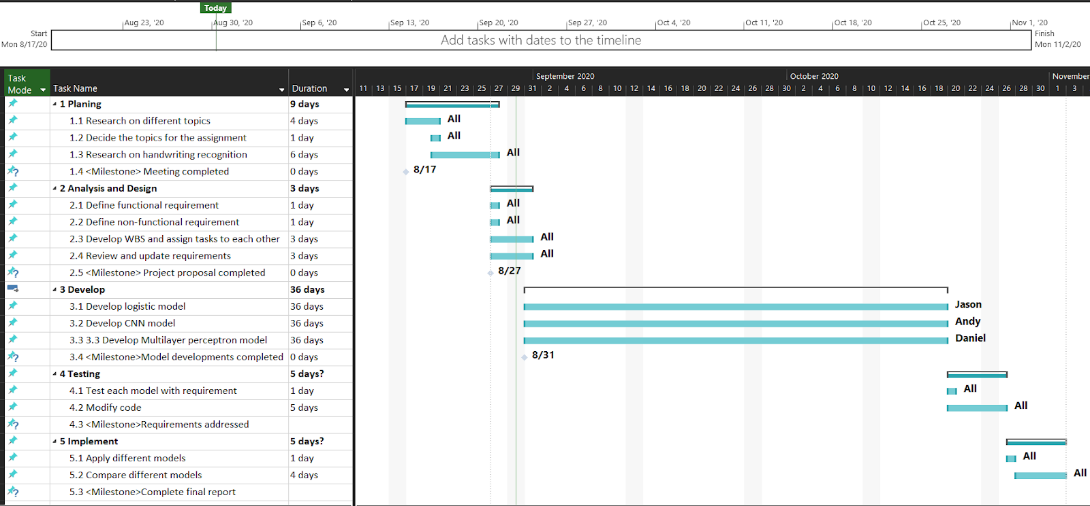


Fig 5: Confusion matrix of the best model

4. Contribution:

As a part of the team, Andy (Xieyuan) participated every group meeting and discussion including the progress report and final report. As our Gantt chart shown below, Andy focused on developing convolutional neural network individually in Develop stage, while he also contributed his effort on Planning stage, Analysis and design stage, Testing stage and Implement stage.



References

Bertrand, C., 2020. MNIST: CNN, Grid Search, Data Augmentation. [online] Kaggle.com. Available at: <https://www.kaggle.com/cedricb/mnist-cnn-grid-search-data-augmentation> [Accessed 24 October 2020].

Ghouzam, Y., 2020. Introduction To CNN Keras - 0.997 (Top 6%). [online] Kaggle.com. Available at: <https://www.kaggle.com/yassineghouzam/introduction-to-cnn-keras-0-997-top-6> [Accessed 20 October 2020].

GitHub. 2020. Tpvt99/Character-Recognition-Cnn. [online] Available at: <https://github.com/tpvt99/character-recognition-cnn/blob/master/cnn.py> [Accessed 20 October 2020].

Gupta, J., 2020. Going Beyond 99% — MNIST Handwritten Digits Recognition. [online] Medium. Available at: <https://towardsdatascience.com/going-beyond-99-mnist-handwritten-digits-recognition-cfff96337392> [Accessed 20 October 2020].

LeCun, Y., Cortes, C. and Burges, C., 2020. MNIST Handwritten Digit Database, Yann Lecun, Corinna Cortes And Chris Burges. [online] Yann.lecun.com. Available at: <http://yann.lecun.com/exdb/mnist/> [Accessed 24 October 2020].

Lee, W., Park, S. and Sim, K., 2020. Optimal hyperparameter tuning of convolutional neural networks based on the parameter-setting-free harmony search algorithm. Optimal hyperparameter tuning of convolutional neural networks based on the parameter-setting-free harmony search algorithm, 17.

Stöttner, T., 2020. Why Data Should Be Normalized Before Training A Neural Network. [online] Medium. Available at: <https://towardsdatascience.com/why-data-should-be-normalized-before-training-a-neural-network-c626b7f66c7d#:~:text=Among%20the%20best%20practices%20for,and%20leads%20to%20faster%20convergence.> [Accessed 24 October 2020].