PROJECT WRITE-UP

MINIPROJECT 3: MULTI-LABEL CLASSIFICATION OF IMAGE DATA

December 14, 2020

Jeongwoo Moon, Mairead Maloney, Cong Zhu
McGill University
COMP 551, Fall 2020

Introduction

CNN, the convolutional neural network, is a class of deep learning neural networks. In this project, we explored how it performs when dealing with multi-label image classification problems, given that we had up to 5 digits to classify in a given data entry. We classified images by assigning a digit label to the corresponding image of that digit. To do so, we built a CNN model. We then improved that performance by exploring how different hyper-parameters and model setting affect the model. We then selected number of training epochs that best-performed for our model. This was done to achieve an overall multi-label classification accuracy of 99.488%. Finally, we discussed some of our model observations and possible future improvements.

Training Objective

The training objective of this project was to train our model to accurately classify each digit drawing in the image to the corresponding digit label using a modified version of the MNIST dataset [**mnist**]. First however, we pre-processed the data to prepare it for training.

The first step was to determine the number of digits in each image. This was done two ways. The first was to use the openCV findContours method [**opencv**] to identify each digit in the image, a contour, in the image. The second way was to create a tensor, where each entry was 5. Then, for each entry, we subtracted the number of non-digits. The resulting tensor contained the number of digits in that data entry. We then removed any training data entry where the number of digits could not be determined, which we deemed noise.

Then, we retrieved the images of the individual digits. First, we removed the top, bottom and side sections of the images, keeping only the central pixels of the image, from $0+((64/n)/2)$ to $64-((64/n)/2)$, where n denotes the number of digits in the image. These pixels were then split into 12x12 pixel images containing a single digit. Appendix A provides a demonstration of a digit retrieval.

Results

Model Analysis

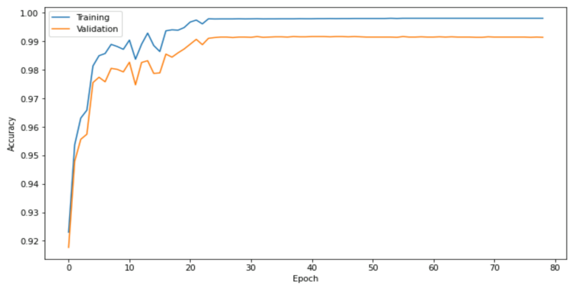
We then determined the best model for our data by tuning the number of CNN layers, learning rate, and batch size. For each, we measured the accuracy while using the best performing values for the other parameters. These accuracy measures are recorded in Appendix B.

Training Epochs

The optimal number of epochs varied based on hyper-parameter values. In our best model, we used 80 epochs, at the cost of training time and processing power used. However, when plotting the accuracy for each hyper-parameter value versus the number of epochs, we observed that the validation accuracy converged as of 30 epochs, regardless of the model configuration. From this we concluded that setting the number of epochs to 30 or more was best for our model.

Best Performing Model

We determined our final model with best performance by selecting the best performing hyper-parameter values and 80 epochs. This model had the following hyper-parameter values and associated accuracy graph.



(a) Accuracy

Hyper-Parameter	Value
CNN Layers	2
Batch Size	4
Learning Rate	0.001
Max Number of Epochs	80
Accuracy (%)	99.488

(b) Hyper-Parameters

Figure 1: Best Performance Hyper-Parameters and Accuracy

Discussion and Conclusion

Model and Parameters

During the project, we made several observations to better understand the CNN concepts. First, when we added more training layers to the model, the accuracy did not significantly decrease after reaching a maximum. This differs from other regression models, which often over-fit in this case. Another observation was that when we set the learning rate to be higher than approximately 0.01, the model performance dropped to approximately 0. This is probably caused by the model weights being overly rectified.

Possible Improvement

A major component of our CNN classification model was the findContour() method in openCV which counts how many digits are in an image. Although we classified the correct number of digits inside each training instance with zero error, this may not generalize to test data. This is because image segmentation with openCV has a lot of fine-tuning required to find the appropriate contours, and is sensitive to noise in the image. To further optimize the performance, a possible improvement would be to improve our own digit retrieval helper method.

Statement of Contributions

- Jeongwoo Moon: Model and Helper Method Implementation
- Mairead Maloney: Model Testing, Report
- Cong Zhu: Hyper-Parameter Tuning, Report

Appendices

A. Split Image Demonstration

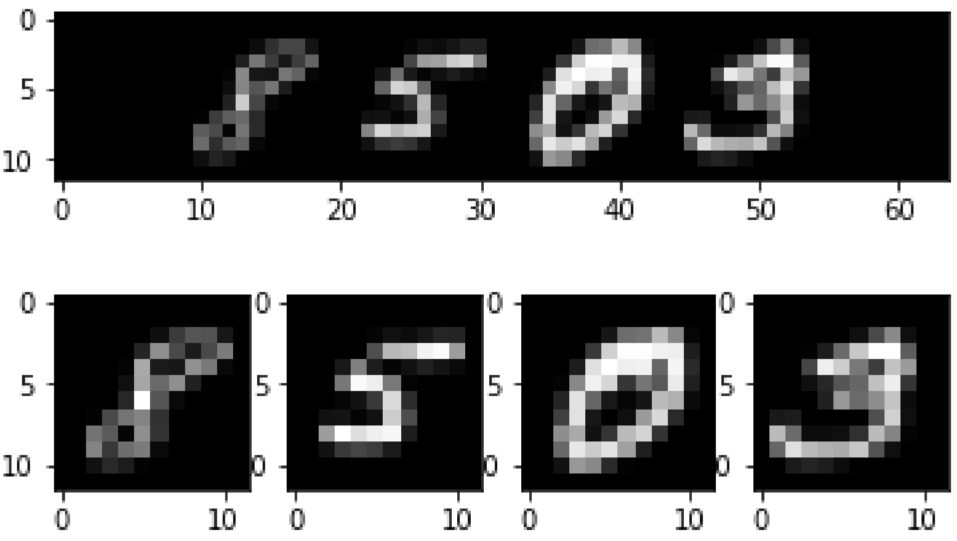


Figure 2: Individual Digits Split Image Demonstration

B. Hyper-Parameter Tuning

B.1. CNN Layer Tuning

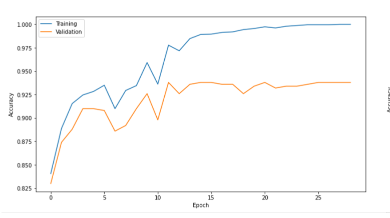
Number of CNN Layers	1	2	3
Accuracy	0.938	0.945	0.934

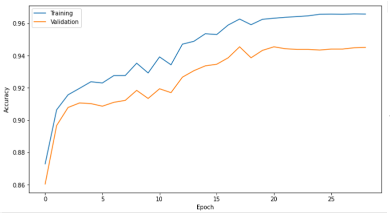
Figure 3: Number of CNN Layers Based Accuracy Measures

B.2. Batch Size Tuning

Batch Size	1	5	10	20
Accuracy	0.92	0.945	0.944	0.938

Figure 5: Batch Size Based Accuracy Measures


(a) 1 Layer


(b) 2 Layers

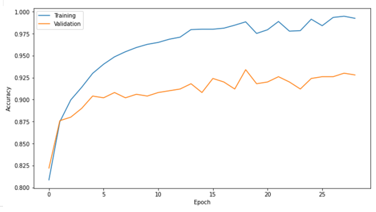
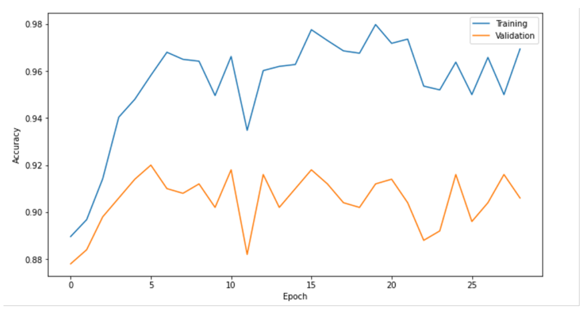
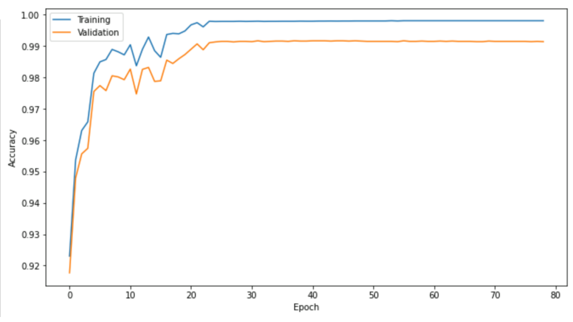
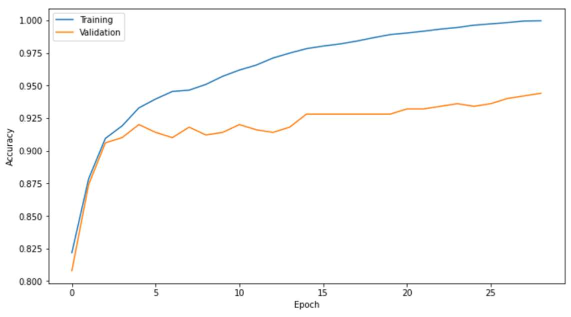

(c) 3 Layers

Figure 4: Number of CNN Layers Accuracy Visualization


(a) Batch Size = 1


(b) Batch Size = 4


(c) Batch Size = 10

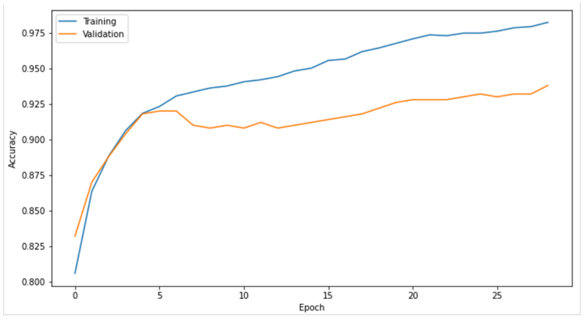
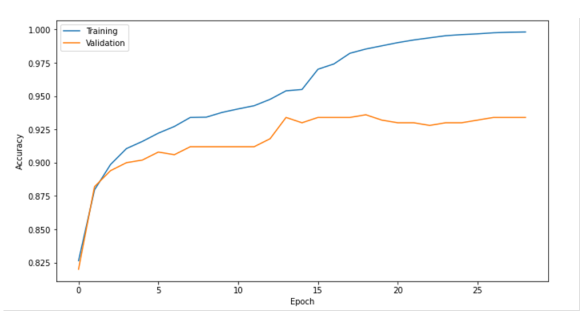

(d) Batch Size = 20

Figure 6: Batch Size Accuracy Visualization

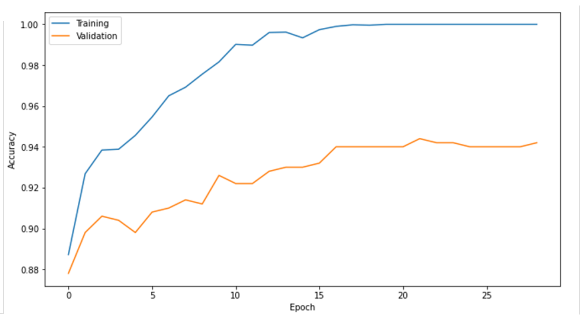
B.3. Learning Rate Tuning

Learning Rate	0.0005	0.001	0.003	0.01
Accuracy	0.936	0.945	0.94	0.91

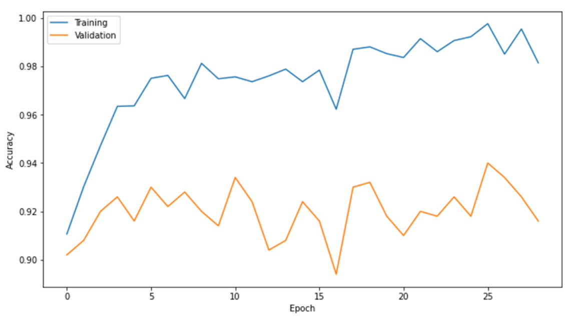
Figure 7: Learning Rate Based Accuracy Measures



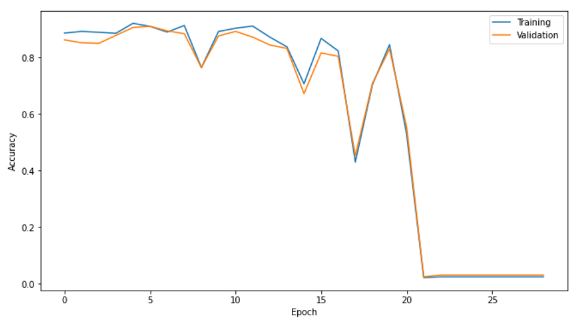
(a) LR = 0.0005



(b) LR = 0.001



(c) LR = 0.003



(d) LR = 0.01

Figure 8: Number of CNN Layers Accuracy Visualization