Assignment 4 Report - Classifying MNIST Images

By: Andy Lee & Monami Waki

Model Implementation

The convolutional neural network (CNN) model used for this project was implemented mainly using the Keras and Tensorflow Python libraries. The Sequential model mainly consisted of 5 Conv2D layers. Starting with 16 filters each with kernel size 3x3, we doubled the number of filters for each Conv2D layer. The 'relu' function was used as the activation function. Between each Conv2D layer, we added a MaxPooling2D later as well as a Dropout layer to perform dimensionality reduction and prevent overfitting. After the Conv2D layers, flattened the data and added 3 Dense layers of size 120, 84, and 10, in the respective order. The last Dense layer utilized a 'softmax' activation function. The optimizer used was 'adam' and the loss function used was 'categorical_crossentropy' which is the industry standard for image classification.

The CNN architecture was chosen after doing many test runs. This model resulted in the highest accuracy out of all the architectures we tried, and was also relatively fast, with each epoch taking around 7 minutes.

Results

Our best model performed with approximately a 0.87 categorization accuracy score, which was a step up from our second-best model with a 0.75 accuracy score. While a batch size of 128 was used in both models, our improved accuracy was attributed to our increase in the number of epochs. This was also a beneficial decision because we did not have to forgo risk of overfitting by training the model with 10 epochs rather than 5 because the "loss" and "val_loss" values continuously decreased with each subsequent epoch.

Challenges

There were many challenges that we encountered during implementation. The first of which was the Google Colab platform. The Google Colab session would frequently crash and restart due to the Out of RAM error, deeming it almost impossible at times to make any progress. This made training the model exceptionally difficult since more complicated models would quickly run out of RAM. Secondly, the most difficult part of the assignment was in preprocessing the data. At first, we tried simply using the images as they were given and the model accuracies stagnated around 0.25. To increase the accuracy, we realized that the backgrounds from the images had to be removed. Our original implementation was to scan through the entire nparray as modify non white pixels to completely black pixels. However, this process was very slow and used up a lot of RAM. We were unaware for a long time that this process could be done in one line of code using a threshold function. Lastly, the other difficulty arose from a misunderstanding of the image properties. When the show_image() function was called, the images were printed in color. So we believed that the images needed to be turned into grayscale first. However, it turned out that the show_image() function for some reason was outputting a colored image even though the original data was in greyscale, deeming out efforts to convert the image pointless.

Conclusion

We learned a lot about the power of using Python libraries to perform preprocessing tasks, which was much simpler than the manual approaches that we had originally attempted. We also learned about the influence of various CNN architectures and hyperparameters on the training accuracy score through many trials and adjustments of such inputs. Experimenting with the addition of new Conv2D layers with differing kernel sizes and dense layers was interesting and allowed us to implement concepts from the lectures in a more tangible way. Overall, while we learned a great deal about how to create, train, and test a CNN, our accuracy score was not as high as we would have hoped, so data augmentation would be a future avenue for improvement.

Individual Contributions

Andy Lee:

- ➢ Wrote entire code for Imports, Download Data, Unzip Data, Testing of CNN Model and Submission of Results sections
- ➢ Cooperated with Monami Waki to write code for the Data Preprocessing, Creation of CNN Model, Training of CNN Model sections.
- ➢ Wrote Model Implementation, Challenges, and personal section for Individual Contribution section of this report

Monami Waki:

- ➢ Cooperated with Andy to write code for the Data Preprocessing, Creation of CNN Model, Training of CNN Model sections.
- ➢ Wrote the Results, Conclusion, and Individual Contribution section of this report.