Report of Classification of Handwritten Digits by a Convolutional Neural Network

Xuan ZHOU

My report is a classification of handwritten digits using two models of convolutional neural networks (CNN). This classification is performed in the MNIST handwritten database, a dataset of handwritten digits that is divided into two parts, which are a training set and a test set of data. The database contains a training set of 60,000 examples and a test set of 10,000 examples is composed of digits. The size of each image is 28x28 pixels and the grayscale of each image is between 0 and 255.

I. The First Model of CNN

First, we need to import libraries to read and split the data. Then we define 4 variables, 'x_train', 'y_train', 'x_test' and 'y_ test'. Reshape "x_train" and "x_test" according to the input image format. In reshape, we have converted the RGB of the image to grayscale easy computation by dividing with a 255-pixel value to get the value between 0-1. Figure 1 shows the results of the visualization of 200 images from the MNIST dataset.

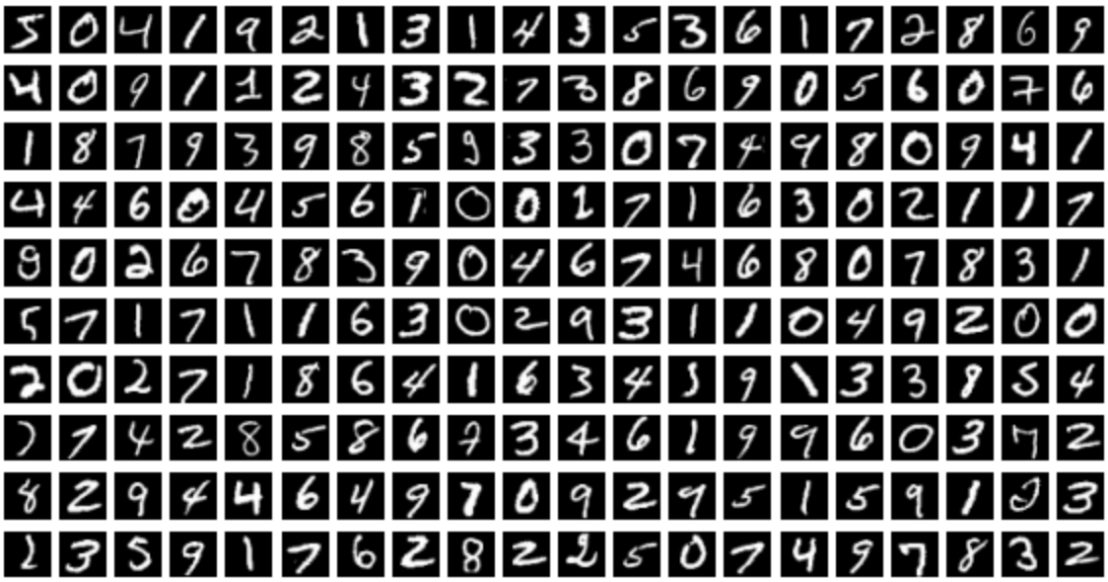

Figure 1. Data Visualization

After completing the data visualization, the next step is to create a CNN model. CNN consists of 4 hidden layers which help in the extraction of the features from the images and is able to predict the result. The layers of CNN are the Convolutional Layer, ReLu Layer, Pooling Layer and Fully Connected Layer. The convolution layer acts as a filter for the activation function. It takes a feature from the input image, then filters different features from that image and makes a feature map, and finally moves the filter over the whole image and calculates the value of each pixel. The role of ReLu is to remove all negative pixel values from the image and replace them with zeros. The pooling layer serves to reduce the image size, thus increasing the computational speed and reducing

the computational cost. The pooling layer is moved from pixel to pixel in 2 x 2 steps and the window is moved to the whole image. In each window, the largest value is selected and this process is performed for each part of the image. The fully connected layer is the last layer of the CNN, which is the part where the actual classification takes place. All matrices from the pooling layer are stacked in this layer and placed in a list. The higher values are the prediction points for a given image.

In this model, a 32-layer convolutional layer of 3x3 is used and Couv2D is performed twice. The activation function in this study is ReLu, which is a widely used activation function nowadays. The size of the pooling layer is set to 2x2 and Dropout is set after Maxpooling. A Flatten is set before the dense of 128 neurons. The total parameters of CNN are 1,199,882. The trainable params are 1,199,882, and the non-trainable params are 0. Figure 2 in the following display.

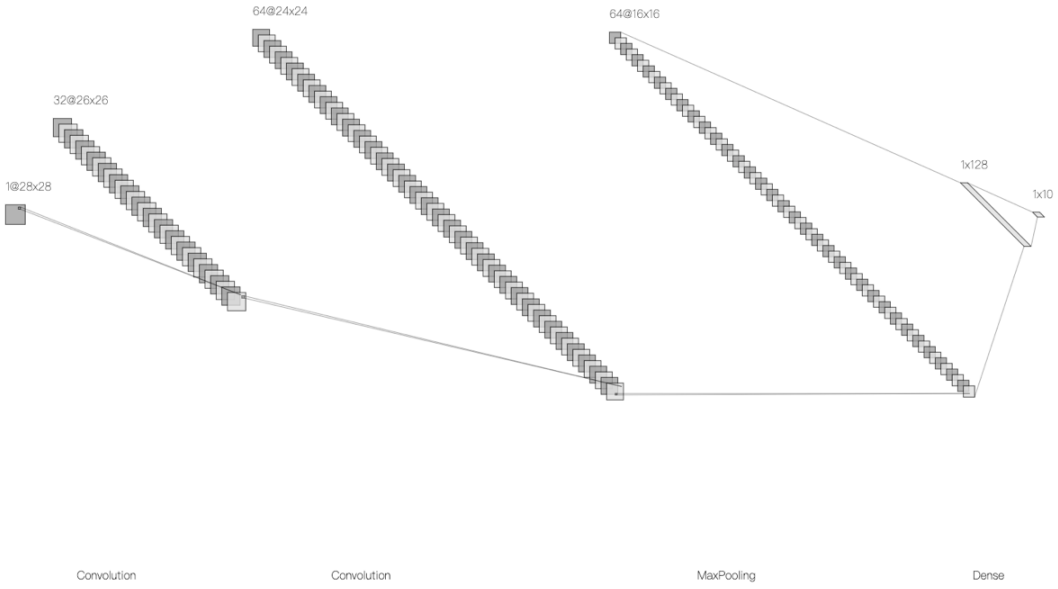

Figure 2. The Architecture of CNN

Before evaluating the model, we can first evaluate the training model on the dataset. In training the model, I performed 10 iterations. As the process started, validation accuracy and validation loss resulted. With each period, the accuracy of the validation is improving. We can see the results of our model on the test dataset, which are the test loss that is 1.30 and epochs of 10. The test accuracy is 0.78.

Predicting the model data is the final step in our project, where we will check if our model accurately predicts the numbers and the percentage of correct predictions. To analyze the results, I have drawn a confusion matrix as shown below.

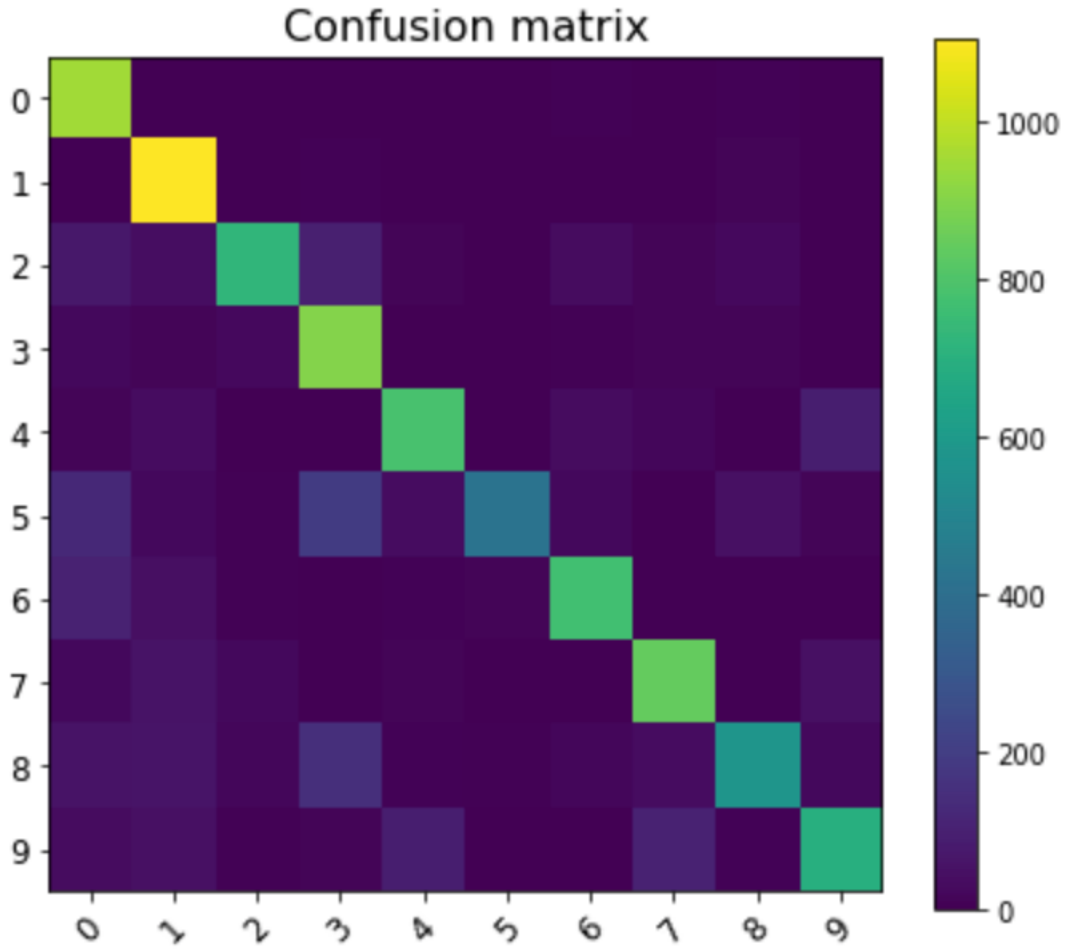

Figure 3. Confusion Matrix of CNN

We can find that the training data of the diagonal of the confusion matrix are more accurate. In addition, I also made a graph to clearly observe the model loss and accuracy, as shown in Figure 4.

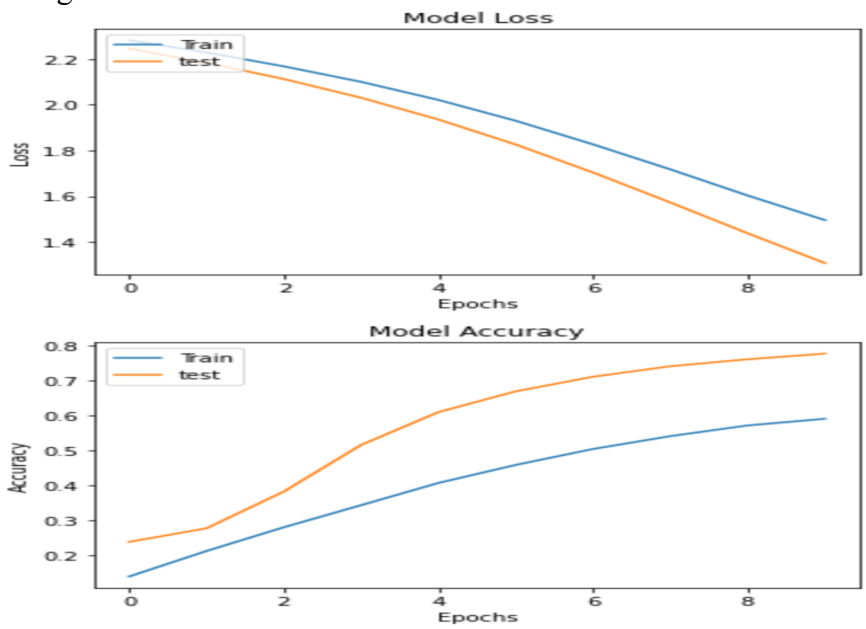

Figure 4. The Model Loss and Accuracy

II. A New Model

In this section, I created a new model to test on the MNIST dataset. In this model, I used 5x5 30-layer convolutional layers, and the first Couv2D was still performed. the second Couv2D was 3x3 15-layer convolutional layers. The activation function in this study is still ReLu. the size of the pooling layer is set to 2x2 and Dropout is set after Maxpooling. A Flatten was set before the dense 128 neurons, and then two more neurons with dense of 50 and 10 were added with activation functions of Relu and softmax. The total parameters of the CNN were 59,933. 59,933 for trainable parameters and 0 for untrained parameters, as shown in Figure 5 below.

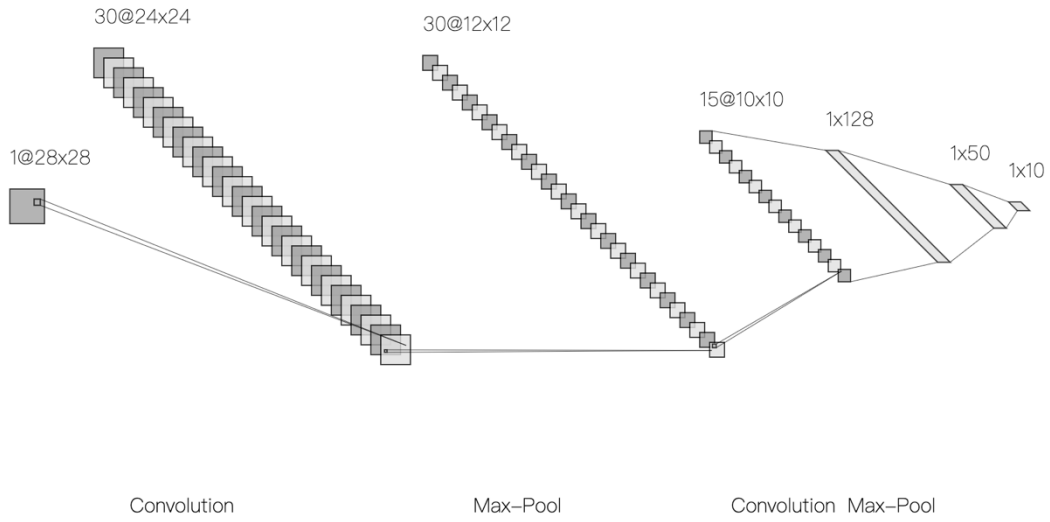

Figure 5. The Architecture of CNN

In training the model, I performed 10 iterations. As the process began, the accuracy of validation and the loss of validation arose. With each period, the speed of validation is improving, but the accuracy is very low. We can see the results of our model on the test dataset that the new model error is 79.92%. To analyze the results, I drew a confusion matrix as shown in the following figure.

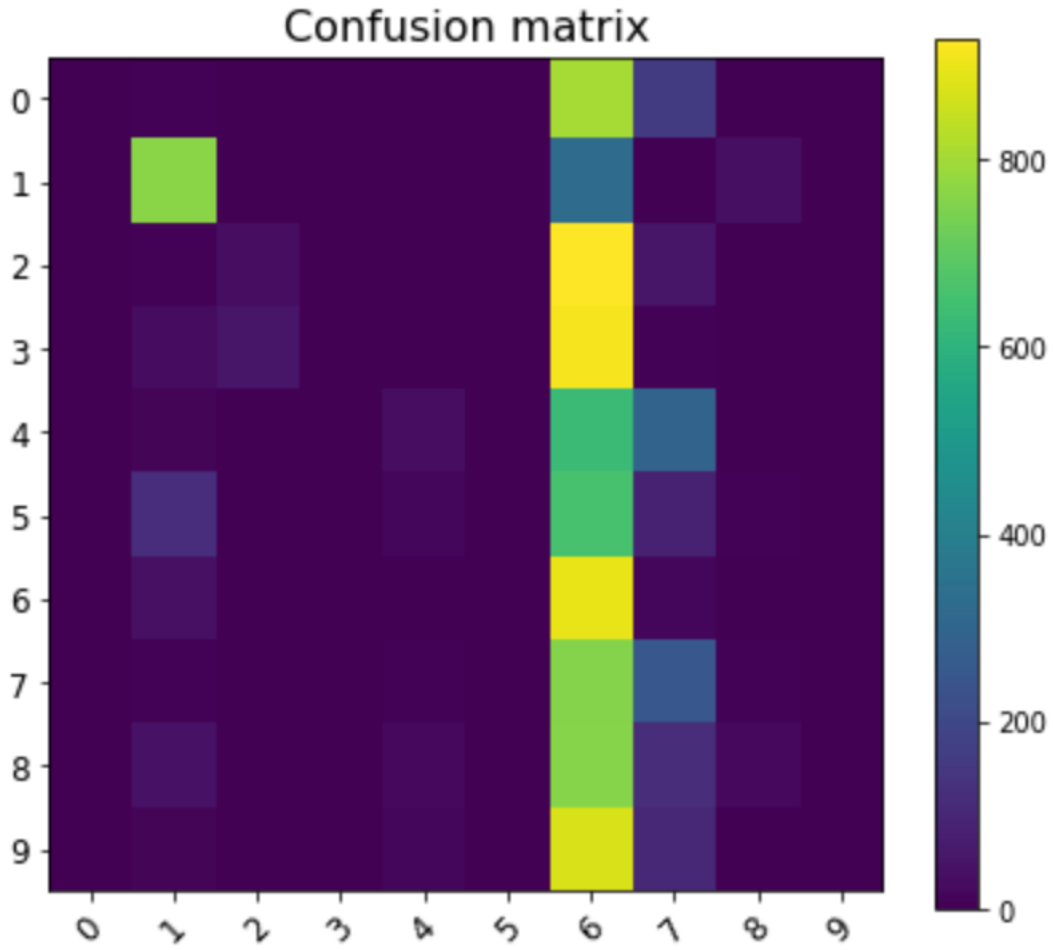

Figure 6. Confusion Matrix of CNN

In addition, I also made a graph to clearly observe the model loss and accuracy, as shown in Figure 7.

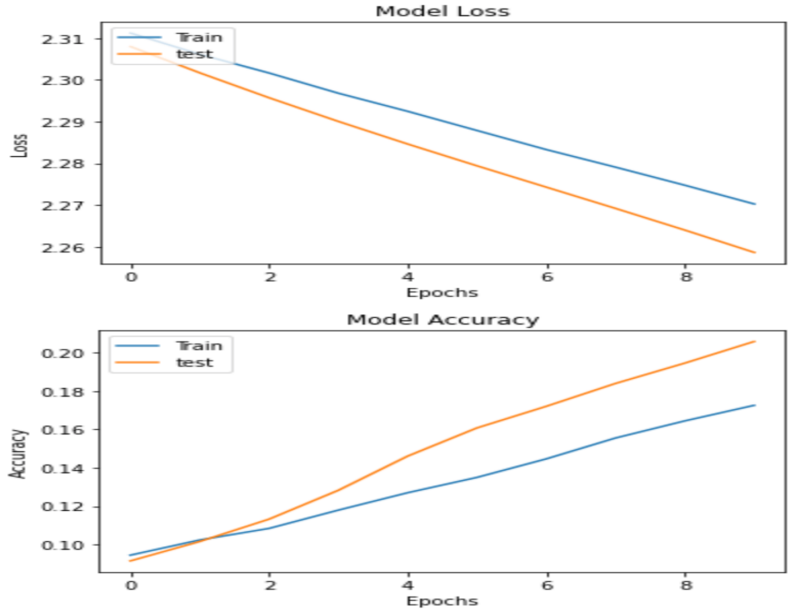

Figure 7. The Model Loss and Accuracy

III. Conclusion

In this work, we can find that through an extensive evaluation of the MNIST dataset, this new model runs faster than the first model, but it is not as accurate and is less accurate than the first model. The first model will run slower but with higher accuracy. The impact of expanding the convolutional layers in the design of CNNs on digit recognition was clarified through testing. In the future, the impact of more hyperparameters on the model is discussed. And discuss more comparisons between classical machine learning algorithms and CNN’s