In Homework 5, I have created a Convolutional Neural Network using TensorFlow designed to analyze the MNIST dataset and predict the numbers that appear in the 28x28 images.

The dataset was divided into 5 parts using 5-Fold cross validation provided by sklearn’s KFold library. This method was used to ensure that the model is not biased toward a particular part of the dataset.

For the CNN architecture, the first convolutional layer uses 32 filters with a kernel size of 3x3 to perform initial spatial feature extraction. The “relu” activation function is used to introduce non-linearity to help the model learn complex patterns.

Following the first layer, a pooling layer of a 2x2 pool size was used to reduce the spatial dimensions of the feature maps.

Then a second convolutional layer with 64 filters and a 3x3 kernel was used to allow the network to build a more abstract representation of the input images.

Another pooling layer of a 2x2 size was added to further reduce dimensionality.

I then implemented a flatten layer to flatten the pooled feature maps to a single long vector. This was necessary to transform the two-dimensional feature maps into a format ready for the fully connected layers that follow.

A Dense Layer of 128 units was used to learn non-linear combinations of the high-level features extracted by the convolutional layers. This also included the “relu” activation function.

A Dropout Layer with a 0.5 rate was applied to reduce overfitting by randomly setting a fraction of the input units to 0 at each update during training phase.

Lastly, an Output Dense Layer of 10 units and a “softmax” activation function was used to output the probability distribution across the 10 classes of digits.

As for the hyperparameters, I used the optimizer “Adam” because it is known for its efficiency with large datasets and its adaptive learning rate, which makes it better than a simple stochastic gradient descent. For the loss function, I used Spare Categorical Crossentropy. This was suitable for multi-class classification problems with labels as integers.

Accuracies

Accuracy for fold 1: 0.92500

Accuracy for fold 2: 0.93500

Accuracy for fold 3: 0.95500

Accuracy for fold 4: 0.93500

Accuracy for fold 5: 0.92965

Average Accuracy: 0.93593

A graph of a graph

Description automatically generated with medium confidence