MNIST Data Classification with Deep Neural Network

09.17.2023

—

By

Parmar Tarun

Introduction & Proposal

As a human, we have a very good visual system that helps us to recognize and process whatever we see around in this world. Humans have the brain to process the visuals sent by eyes and classify them based on what they have learnt in the past. Past experience and knowledge helps them to label whatever they see around them. For machines, to perform such a task is not less than challenging. For a machine to be able to recognize or label an object of the real world requires a robust, well trained classification model. In this report I have tried to implement such a task. I have used MNIST dataset to train a deep neural network model which is responsible for classifying the handwritten digits into one of the 10 digit classes.

Methodology

1. Network Structure

The network built for the project is a fully connected network. To build this network I have used keras's sequential model. The first layer, which is a dense layer, consists of 256 neurons considering the image vector of size 256 followed by an activation layer of type relu. Next the network is appended with a dropout layer of 0.45, a dense layer of 256 neurons and a relu activation layer. Further for the final output dropout layer of 0.45 followed by a dense layer of 10 neurons for the 10 digits with activation function of softmax. Softmax function is used as the output layer so that we have an output in the form of a probability. These probabilities will be used to decide the classification of the input image.

Keeping the above network as a baseline, I have modified this baseline network into 3 other networks:

- Network with more dense layers
- Network with more dense layers and more epochs
- Network with more dense layers, more epochs and tanh as activation function for hidden layers.

All the above models are compiled using the categorical cross entropy as loss function, adam as optimizer and accuracy as metric for evaluating the training and testing results.

2. Training & Validation process

The MNIST Dataset is loaded directly using the keras library. The dataset includes a total of 70000 data points with 60000 splitted for training and 10000 for testing purposes. The labels in the dataset are numerical values which are not suitable for a classification task. Hence, the labels were converted to categorical type where the numerical value is modified to a vector of size 10, each element in the vector being 0 except for the one element to be 1, corresponding to that digit. This is done using the to_categorical function offered by keras.

Next, each data point which is of size 28x28 is converted to a vector of size 256. Such a conversion is done because we are using a fully connected neural network which accepts the input of 256 size vectors. Also the values in the vector are normalized by dividing each of them by 255 (highest possible value)

The training process for first two models was done with 10 epochs and next two with 50 epochs in an experiment of improving the training and testing accuracies. The batch size for all the 4 models was a constant of 128.

Evaluation & Results

I. Training & Validation Results

After completing the training process in just a few seconds, the base model resulted in the accuracy of 97.5% in training and 97.9% in testing. The rest 3 models took a bit longer to train but resulted with better testing and training accuracies.

II. Performance Comparison

The following table depicts the comparative results of all the 4 models used in this project.

MODEL	ACTIVATION Fn.	EPOCHS	TRAIN ACCURACY	TEST ACCURACY
1	Relu	10	97.5 %	97.9 %
2	Relu	10	96.8 %	97.6 %
3	Relu	50	98.5 %	98.4 %
4	Tanh	50	97.6 %	97.8 %

III. Hyperparameter

The following hyperparameters were used / tweaked while training process of the 4 models:

- ***Number of layers***: Hidden layers were increased from 1 in base model to 3 in the other 3 models
- ***Dropout***: A constant dropout of 0.45 was used in the 4 models. The 0.45 dropout indicates that neurons have 0.45 probability of dropping out during the training process.
- ***Activation function***: The activation function for hidden layers for first 3 models was relu and for the 4th model I used tanh. The activation function for the output layer was softmax in all the 4 models to ensure that output corresponding to each digit ranges between 0 and 1.
- ***Number of epochs***: Different epochs' values were used in half of the models in an experiment to get better results than the previous one.

Conclusion

The classification model with deep neural network resulted in some good accuracies for testing along with training. Neural networks for classification of handwritten images proved to be a good choice in developing a better model.

With just 1 hidden layer in the base model outputted a good accuracy of 95+. The increase in the hidden layers in the next model did not improve the accuracy further, but a side by side increase in the epochs in the 3rd model did.