Comp 475

Reinforcement and deep learning

IN13/00028/20

Brian Kiptoo

Methodology

In developing the convolutional neural network that classifies based on the MINST dataset, the first step was to import the necessary packages such as TensorFlow, Keras, NumPy, and Matplotlib.

Then loading of the MNIST dataset which contains grayscale images of fashion items. After which normalization of the pixel values of the images to be between 0 and 1is done, followed by reshaping of the data to fit the CNN model's input requirements and conversion of class labels to one-hot encoded vectors.

After carrying out preprocessing created a CNN model with 8 layers and then compiled the model with an optimizer, loss function and metrics after which I trained the model on training data over multiple epochs and trained the model on the test data to assess the performance.

Model architecture

The model architecture of the model has the following layers

a) Input Layer
   Convolutional layer with 32 filters, a 3x3 kernel size, ReLU activation function.
   MaxPooling layer with a 2x2 pool size.
b) Hidden Layer
   Convolutional layer with 64 filters, a 3x3 kernel size, ReLU activation function.
   MaxPooling layer with a 2x2 pool size.
c) Hidden Layer
   Convolutional layer with 64 filters, a 3x3 kernel size, ReLU activation function.
   Flatten layer to convert the 2D output to a 1D vector.
d) Fully Connected (Dense) Layers

   Dense layer with 64 neurons and ReLU activation function.
   Output layer with 10 neurons (equal to the number of classes) and softmax activation function.

Training and results

With the first part of the, I increased the filters with the power of two from 32 to 64 and maintained the next one at 64, with 10 epochs the model achieved an accuracy of 90.63% and loss of 0.294.

For the second set of tests, I increased the filters with the powers without ending at 64 that is 32, 64,128 with 10 epochs the accuracy achieved was 90.56% less than the start tests by 0.1% and achieved a loss of 0.3204

For the third test I decided to increase the number of epochs to 20 to see the effects of epochs on the model. The model achieved accuracy of 90.99% which is 0.12% less than the highest achieved at some point in the process, the results produced a wave like results if the were to be drawn with higher accuracies being achieved at epoch 8 and epoch 20. From this I deduced that the optimal points for epochs range between 7 and 10 after which the circle will repeat itself no matter the epochs

Challenges

Accuracy, the accuracy of the model achieved is very low with an initial percentage of 90% but through tuning the number of epochs and filters it increased by 1%

Parameters, its time consuming trying on several parameters e.g. epochs

Performance improvement insights

To improve the model Architecture, experimentation with increasing or decreasing the number of layers. A deeper model may capture more complex features. Adjusting the number of filters in convolutional layers. Experimentation with different kernel sizes in convolutional layers with the aim of making them small.
To improve on the training process, adjustment on the learning rate of the optimizer, experimentation with different batch sizes during training with the aim of making the batches mall. Other steps would include, ensuring that input features are scaled appropriately. For images, normalizing pixel values to a range of 0, 1. Using pre-trained models and fine-tuning them for your specific task and using automated tools or hyperparameter tuning libraries to efficiently search for the best hyperparameters.