ECS69P Coursework

Problem

This project aims to build a neural network classifier with a specific architecture. It will use the Fashion-MNIST dataset where every image will need to be classified into 1 of 10 classes. The model will be built and trained using the training set and then the model will be evaluated using the test set.

Model

The final model consists of a stem and four blocks (four blocks performed better than one, two or three blocks in experiments). In the stem the images are split into 4 patches before passing through the rest of the model. Between each linear layer throughout the model there is an activation stage, consisting of either ReLU function or a transpose function. The final block gives 10 outputs for each feature.

The features of each image are consolidated by taking the mean of each of the 10 output values across all features and the images are classified using Softmax regression.

Training

To train the model the Adam optimizer is used. This was selected as it performed better than SGD during experiments. A scheduler was also included to reduce the learning rate of the optimizer on plateau of results to optimize further.

The Final Model

To reach the final hyperparameters seen below, many experiments were run. The model started with one block, an SGD optimizer over 10 epochs with 16 patches (Accuracy: 22.89%). This significantly improved by changing to the Adam optimizer (Accuracy: 72.71%). The model improved updating the learning rate to 0.001 and increasing the number of epochs to 50 (Accuracy: 82.4). At this stage a second block was added (Accuracy: 87.63). Other experiments such as adding a third block, adding data augmentation (horizontal flip, random erasing), changing batch size, adding dropout, adjusting learning rate during training using a scheduler. The only experiment that increased the accuracy was to increase number of blocks to 4 and to add a ColorJitter transformation. This enabled the model to reach an accuracy of 89.44%. The hyperparameters as well as the curves of the training accuracy, test accuracy and training loss can be seen below.

The hyperparameters that were used to train the final model are as follows:

Batch Size: 256
Number of inputs: 196

Number of patches: 4
Hidden Layers: 256, 128, 64, 32, 16

Dropout: p=0.3

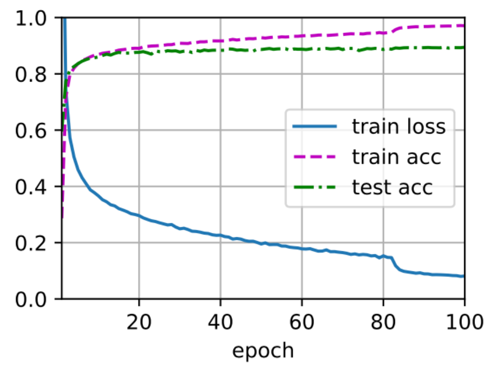
Adam learning rate: 0.001

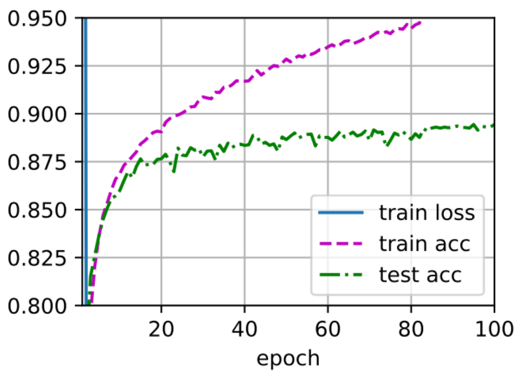
Number of epochs: 100

Scheduler patience = 2

Scheduler threshold = 0.001

The below figures show the curves of the training accuracy, test accuracy and loss across the 50 epochs.





The final results were:

```
Training Loss = 0.08143206913471222
Training Accuracy = 0.9707166666666667
Test Accuracy = 0.8944
```

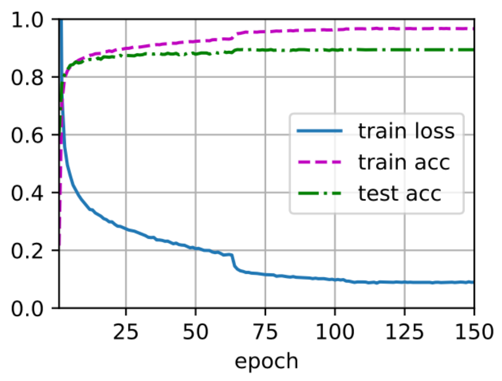
Further Experiments

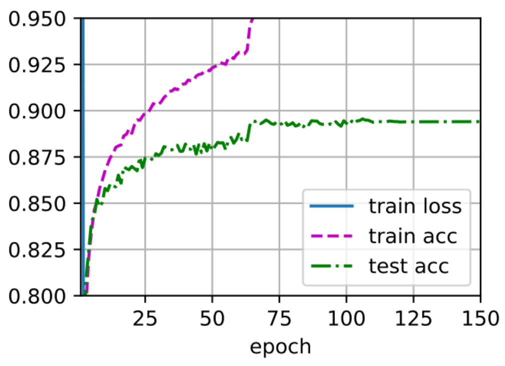
From the results it’s clear that the model is overfitting. Further experiments aimed to reduce overfitting by adding weight decay to the optimizer, increasing the dropout rate and instances of dropout throughout the model. While these models reduced overfitting and test accuracy’s came close to the one recorded above, none exceeded it. Two examples of the results of such experiments can be seen below.

For this experiment only the number of epochs was adjusted and one more instance of dropout in the model.

Hyperparameters:

Number of epochs: 150





The results for this experiment were:

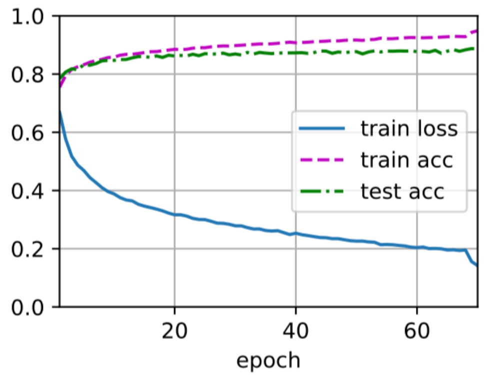
```
Training Loss = 0.0896784720102946
Training Accuracy = 0.9668
Test Accuracy = 0.8941
```

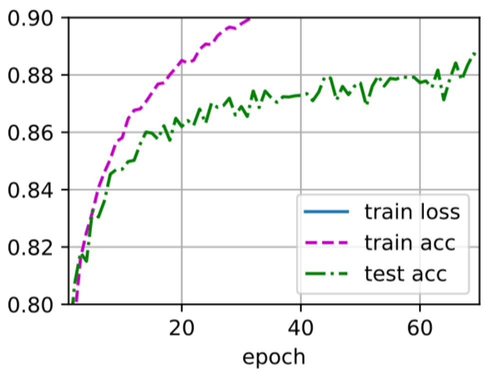
For this experiment the number of epochs was reduced (due to time constraints) and optimizer weight decay was added.

Hyperparameters:

Number of epochs: 70   Dropout: p=0.35

Optimizer weight decay = 0.00002





The final results were:

```
Training Loss = 0.1296784720102946
Training Accuracy = 0.9458
Test Accuracy = 0.8872
```

Future experiments would include running this experiment for more epochs and fine-tuning hyperparameters such as weight decay, dropout.