Theory and Practice of Deep Learning

Chang Jun Qing 1002088

February 19, 2019

Coding Assignment
The best average accuracy obtained was 88% (8821/10000) with an average loss of 0.3269

Per class accuracy we have
Class 0: Accuracy: 841/1001 (84%)
Class 1: Accuracy: 972/987 (98%)
Class 2: Accuracy: 810/1051 (77%)
Class 3: Accuracy: 894/1030 (87%)
Class 4: Accuracy: 823/1036 (79%)
Class 5: Accuracy: 963/999 (96%)
Class 6: Accuracy: 653/877 (74%)
Class 7: Accuracy: 932/984 (95%)
Class 8: Accuracy: 969/1010 (96%)
Class 9: Accuracy: 964/1025 (94%)

Where the hardest class to predict as class 6 with accuracy of only 74%
The folling are the images of loss over epochs (50 epochs with early stopping), as well as the output from the best model parameters. Attached in the folder: $fashionmnist\_fc.pt$ is the saved model parameters.
This was done with cuda available (also works without cuda). Best model parameters will be saved in the current working directory

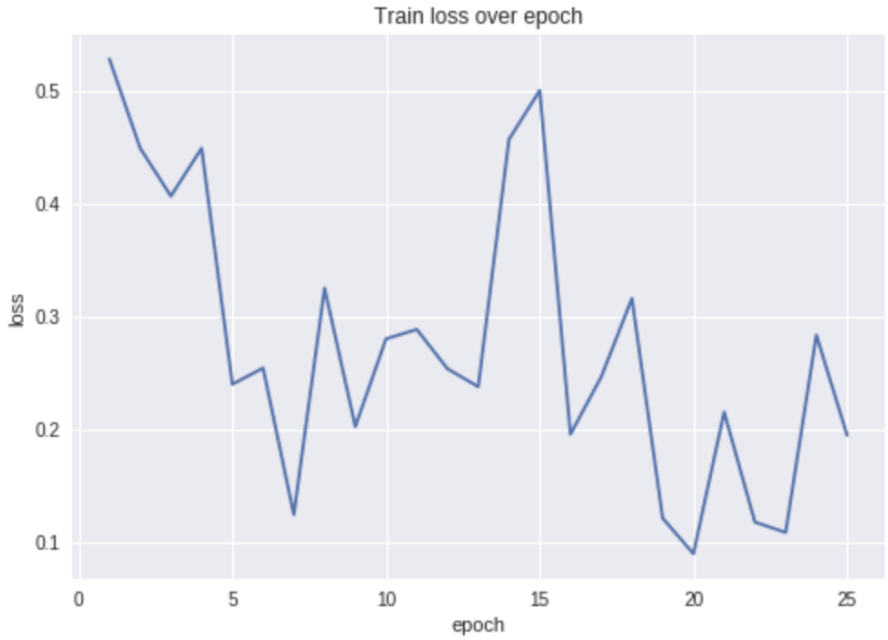

Figure 1: Train loss over epochs

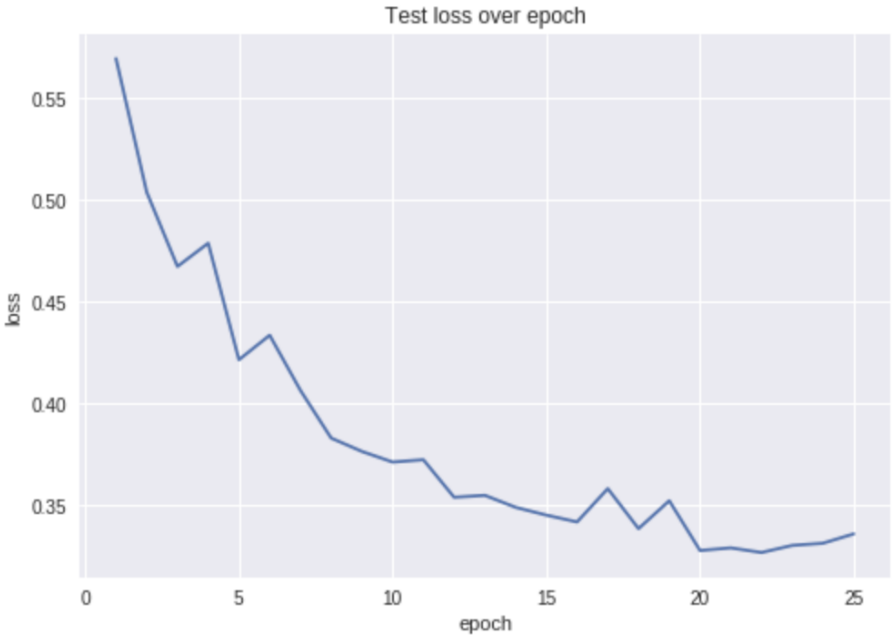

Figure 2: Test loss over epochs

```
Test set: Average loss: 0.3269, Accuracy: 8821/10000 (88%)

Class 4: Accuracy: 823/1036 (79%)
Class 1: Accuracy: 972/987 (98%)
Class 9: Accuracy: 964/1025 (94%)
Class 0: Accuracy: 841/1001 (84%)
Class 8: Accuracy: 969/1010 (96%)
Class 2: Accuracy: 810/1051 (77%)
Class 5: Accuracy: 963/999 (96%)
Class 7: Accuracy: 932/984 (95%)
Class 3: Accuracy: 894/1030 (87%)
Class 6: Accuracy: 653/877 (74%)
```

Figure 3: Output of best parameters

```python
use_cuda = torch.cuda.is_available()
device = torch.device("cuda" if use_cuda else "cpu")
log_interval = 300
learning_rate = 0.01
momentum = 0.01
epochs = 50
```

Figure 4: Hyperparameters

Editing Hyperparameters
Hyperparameters can be edited at the top of the python file