Machine Learning Homework 2
Autoencoder and Feedforward Neural Network

Ahmed Senih Yıldırım

June 12, 2025

Question 1: Autoencoder (AE)

1. Architecture and Hyperparameters

The autoencoder was implemented as a fully connected feedforward network. The encoder part compresses the 28x28 grayscale Fashion-MNIST images (784 dimensions) into a latent representation of size 64. The decoder reconstructs the original image from this latent code.

Encoder:

- Input: 784
- Linear(784, 256) → ReLU
- Linear(256, 64)

Decoder:

- Linear(64, 256) → ReLU
- Linear(256, 784) → Sigmoid
- Output reshaped to (1, 28, 28)

Other Hyperparameters:

- Loss: Mean Squared Error (MSE)
- Optimizer: Adam
- Learning Rate: 0.001
- Batch Size: 128
- Epochs: 20

2. Training Process and Reconstruction Loss

The model was trained for 20 epochs. After initial debugging, it was confirmed that the dataset was properly normalized to the [0, 1] range. The training loss decreased steadily over the epochs, and the final test reconstruction loss was:

```
Test Reconstruction Loss: 0.0066
```

```
(autoencoder) senih@Senih-MacBook-Air Homework2 % python autoencoder_fashionmnist.py
Example batch shape: torch.Size([128, 1, 28, 28])
dtype: torch.float32
min/max: 0.0 1.0
Epoch [1/20] - Train Loss: 0.0000
Epoch [2/20] - Train Loss: 0.0316
Epoch [3/20] - Train Loss: 0.0163
Epoch [4/20] - Train Loss: 0.0137
Epoch [5/20] - Train Loss: 0.0122
Epoch [6/20] - Train Loss: 0.0112
Epoch [7/20] - Train Loss: 0.0105
Epoch [8/20] - Train Loss: 0.0098
Epoch [9/20] - Train Loss: 0.0094
Epoch [10/20] - Train Loss: 0.0090
Epoch [11/20] - Train Loss: 0.0086
Epoch [12/20] - Train Loss: 0.0083
Epoch [13/20] - Train Loss: 0.0081
Epoch [14/20] - Train Loss: 0.0079
Epoch [15/20] - Train Loss: 0.0077
Epoch [16/20] - Train Loss: 0.0075
Epoch [17/20] - Train Loss: 0.0073
Epoch [18/20] - Train Loss: 0.0072
Epoch [19/20] - Train Loss: 0.0071
Epoch [20/20] - Train Loss: 0.0070

Test Reconstruction Loss: 0.0066
```

Figure 1: Autoencoder Training Loss per Epoch

3. Original vs Reconstructed Images

The model successfully reconstructed test images. Most shapes and object silhouettes were preserved, especially for simpler classes like T-shirt/top or Trouser.

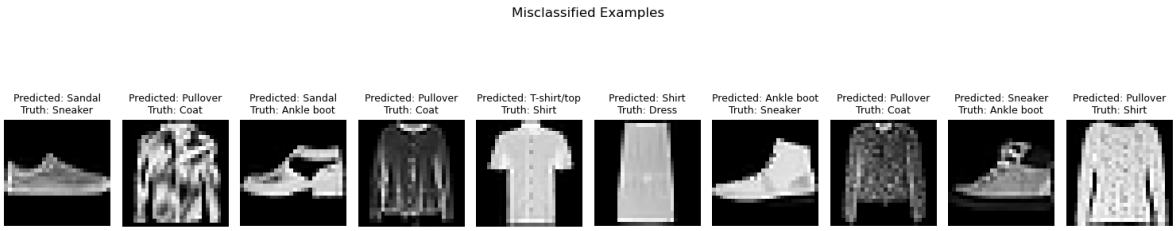

Figure 2: Examples of Misclassified or Poorly Reconstructed Outputs (for clarity)

4. Generated Samples and Observations

By sampling random vectors from the latent space, synthetic samples were generated. The quality of generated samples was lower than the reconstructed ones, but some categories like shoes and coats were distinguishable. This is expected due to the limited capacity of the small fully connected AE.

Question 2: Feedforward Neural Network (FFNN)

1. Architecture and Hyperparameters

The FFNN was built as a three-layer feedforward classifier for Fashion-MNIST.

Architecture:

- Flatten Input (28x28 = 784)
- Linear(784, 512) → BatchNorm → ReLU → Dropout(0.2)
- Linear(512, 256) → BatchNorm → ReLU → Dropout(0.2)
- Linear(256, 128) → ReLU
- Linear(128, 10) → Softmax (via CrossEntropyLoss)

Training Hyperparameters:

- Loss: CrossEntropyLoss
- Optimizer: Adam
- Learning Rate: 0.001
- Batch Size: 128
- Epochs: 50

2. Training Process

The model was trained for 50 epochs. The accuracy and loss steadily improved, starting from 82.6% and reaching over 97% training accuracy. A learning rate scheduler was not used in this run.

```
(autoencoder) senih@Senih-MacBook-Air Homework2 % python feedforward_classifier.py
Using device: cpu
Epoch 1/50 - Loss: 0.4855 - Accuracy: 0.8269
Epoch 2/50 - Loss: 0.3606 - Accuracy: 0.8691
Epoch 3/50 - Loss: 0.3213 - Accuracy: 0.8807
Epoch 4/50 - Loss: 0.3004 - Accuracy: 0.8882
Epoch 5/50 - Loss: 0.2824 - Accuracy: 0.8937
Epoch 6/50 - Loss: 0.2692 - Accuracy: 0.8993
Epoch 7/50 - Loss: 0.2549 - Accuracy: 0.9033
Epoch 8/50 - Loss: 0.2460 - Accuracy: 0.9082
Epoch 9/50 - Loss: 0.2360 - Accuracy: 0.9124
Epoch 10/50 - Loss: 0.2263 - Accuracy: 0.9153
Epoch 11/50 - Loss: 0.2175 - Accuracy: 0.9171
Epoch 12/50 - Loss: 0.2097 - Accuracy: 0.9206
Epoch 13/50 - Loss: 0.2021 - Accuracy: 0.9226
Epoch 14/50 - Loss: 0.1929 - Accuracy: 0.9267
Epoch 15/50 - Loss: 0.1885 - Accuracy: 0.9292
Epoch 16/50 - Loss: 0.1811 - Accuracy: 0.9308
Epoch 17/50 - Loss: 0.1748 - Accuracy: 0.9338
Epoch 18/50 - Loss: 0.1715 - Accuracy: 0.9346
Epoch 19/50 - Loss: 0.1635 - Accuracy: 0.9383
Epoch 20/50 - Loss: 0.1581 - Accuracy: 0.9402
Epoch 21/50 - Loss: 0.1547 - Accuracy: 0.9414
Epoch 22/50 - Loss: 0.1505 - Accuracy: 0.9417
Epoch 23/50 - Loss: 0.1449 - Accuracy: 0.9446
Epoch 24/50 - Loss: 0.1416 - Accuracy: 0.9459
Epoch 25/50 - Loss: 0.1339 - Accuracy: 0.9496
Epoch 26/50 - Loss: 0.1333 - Accuracy: 0.9494
Epoch 27/50 - Loss: 0.1285 - Accuracy: 0.9504
Epoch 28/50 - Loss: 0.1280 - Accuracy: 0.9511
Epoch 29/50 - Loss: 0.1248 - Accuracy: 0.9527
Epoch 30/50 - Loss: 0.1183 - Accuracy: 0.9540
Epoch 31/50 - Loss: 0.1161 - Accuracy: 0.9554
Epoch 32/50 - Loss: 0.1135 - Accuracy: 0.9569
Epoch 33/50 - Loss: 0.1106 - Accuracy: 0.9581
Epoch 34/50 - Loss: 0.1091 - Accuracy: 0.9576
Epoch 35/50 - Loss: 0.1073 - Accuracy: 0.9591
Epoch 36/50 - Loss: 0.1023 - Accuracy: 0.9613
Epoch 37/50 - Loss: 0.1004 - Accuracy: 0.9621
Epoch 38/50 - Loss: 0.1007 - Accuracy: 0.9617
Epoch 39/50 - Loss: 0.0961 - Accuracy: 0.9633
Epoch 40/50 - Loss: 0.0943 - Accuracy: 0.9640
Epoch 41/50 - Loss: 0.0926 - Accuracy: 0.9654
Epoch 42/50 - Loss: 0.0897 - Accuracy: 0.9660
Epoch 43/50 - Loss: 0.0912 - Accuracy: 0.9658
Epoch 44/50 - Loss: 0.0869 - Accuracy: 0.9674
Epoch 45/50 - Loss: 0.0876 - Accuracy: 0.9665
Epoch 46/50 - Loss: 0.0832 - Accuracy: 0.9677
Epoch 47/50 - Loss: 0.0819 - Accuracy: 0.9694
Epoch 48/50 - Loss: 0.0851 - Accuracy: 0.9688
Epoch 49/50 - Loss: 0.0780 - Accuracy: 0.9710
Epoch 50/50 - Loss: 0.0804 - Accuracy: 0.9688

Test Accuracy: 0.9010
```

Figure 3: Training Accuracy and Loss for Feedforward Neural Network

3. Final Performance and Evaluation

The test accuracy achieved was:

```
Test Accuracy: 0.9010
```

This result is visualized in the confusion matrix below:

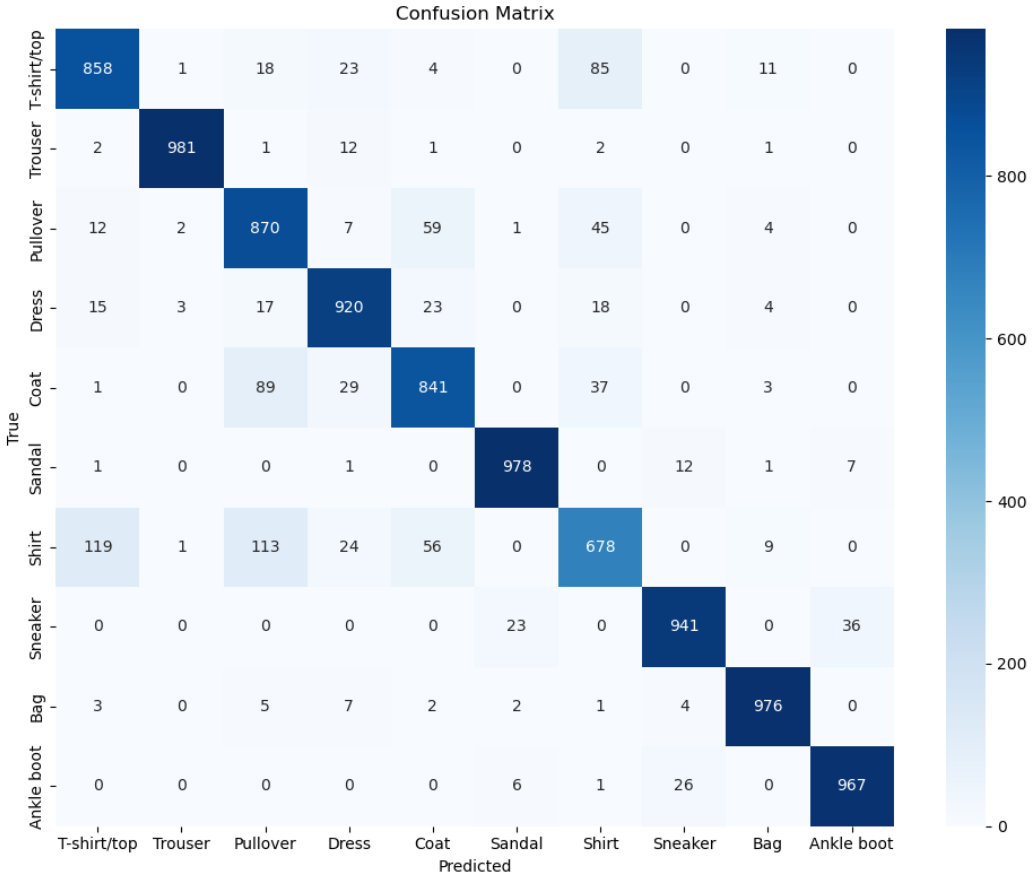

Figure 4: Confusion Matrix on Test Set

Observations:

- Most confusion occurred between similar classes like Shirt vs T-shirt or Coat vs Pullover.
- Misclassified examples are shown in Figure 5.

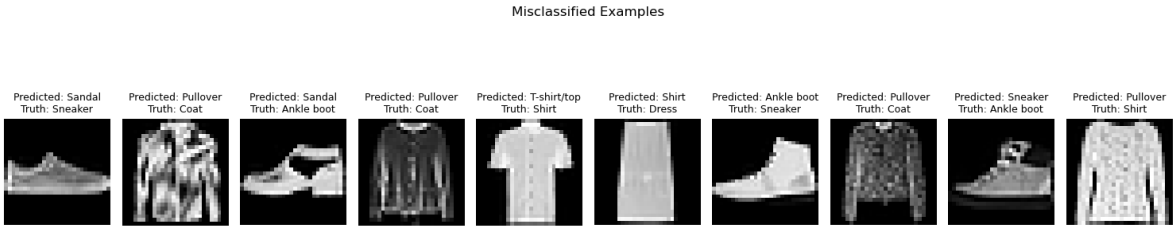

Figure 5: Misclassified Examples with Predicted and True Class Names