Fashion-MNIST CNN Classification

Introduction:

The aim of the project is to build a convolutions Neural Network (CNN) for image classification using the Fashion-MNIST dataset. The assignment involved the training of a baseline CNN model, experimenting with at least one hyperparameter (here network depth), and training both models up to 20 epochs, and testing their performance on a reserved test set. Visualization of learned convolutional filters, feature maps, and confusion matrices and examples that are misclassified were also part of the project.

Fashion-MNIST consists of 70,000 grayscale images (28x28) with an equal representation of 10 types of clothing. The training images are divided into 60,000 and test images consist of 10,000. The training set was further broken down as per instructions into 90 percent training (54,000 images) and 10 percent validation (6,000 images).

Dataset and Preprocessing:

The Fashion-MNIST dataset includes ten classes:

['T-shirt/top','Trouser','Pullover','Dress','Coat','Sandal','Shirt','Sneaker','Bag','Ankle boot'].

Each image was normalized to [-1, 1] using:

No data augmentation was applied. The final split sizes:

- Training = 54,000
- Validation = 6,000
- Test = 10,000

Model Architectures

Two CNN architectures were implemented and evaluated:

Baseline Model (2 Convolutional Layers):

The model begins with a convoluted layer which accepts a single channel input and generates 32 feature maps with a 3 x3 kernel with padding of 1. This is then preceded by a ReLU activation and a max-pooling of 2x2. The next convolutional layer increases the features to 64 channels and once again, the kernel size is 3×3 and padded by 1, followed by another ReLU and 2 max-pooling layer. Once this is done the output is flattened into an output of size 64x7x7 (i.e. 3136). The resultant flattened vector is inputted into a fully-connected layer that has 128 units, a dropout of 0.5, and a ReLU activation. Lastly, the model produces 10 logits with the linear layer, which is the 10 Fashion-MNIST classes.

- **Total parameters:** 421,642

Deeper Model (3 Convolutional Layers):

This model starts with a convolutional layer which receives the single-channel input and creates 32 feature maps with the help of 3 x3 filter after which a ReLU activation and a max-pooling operation are applied. It then uses a second convolutional layer which doubles the number of channels to 64 with another 3×3 filter and followed by ReLU and max-pooling. There is a third convolutional layer, which increases the feature size even more, by a factor of 3 to 128 channels using a 3x3 kernel, which is then followed by a ReLU activation. The resultant output is flattened as a 128x7x7 (= 6272) vector. This vector is passed into a fully-connected layer with 256 units, combined with a ReLU activation and dropout. Lastly, the network produces 10 logits in the final layer which equals the amount of target classes.

- **Total parameters:** 1,701,130

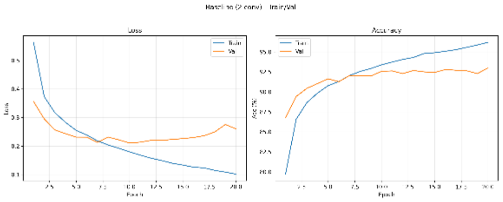
Training Procedure

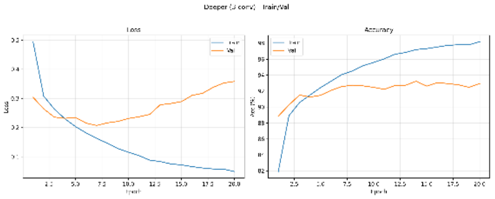
- **Optimizer**: Adam (lr=0.001)
- **Loss function**: CrossEntropyLoss
- **Batch size**: 64
- **Epochs**: 20

During training, both training and validation accuracy/loss were logged and plotted.

Results:

- Training Curves

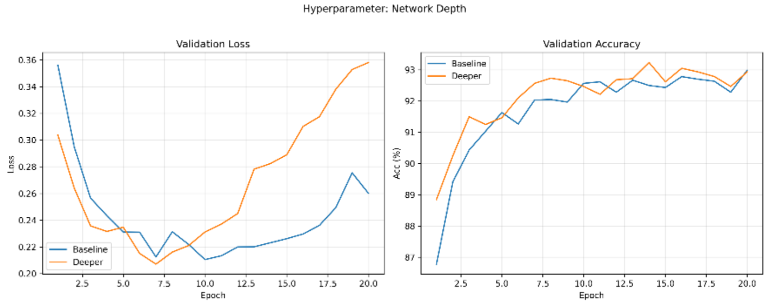




The baseline CNN converged linearly, and showed slightly overfitting after epoch 15. The deeper the network archives slightly high accuracy. But also aggressively overfitted.

Hyperparameter Experiment (Network Depth):

Comparison of validation curves



At epoch 15 the deeper model achieved the highest validation accuracy of 92.88%, slightly higher than the baseline 92.75% at epoch 12

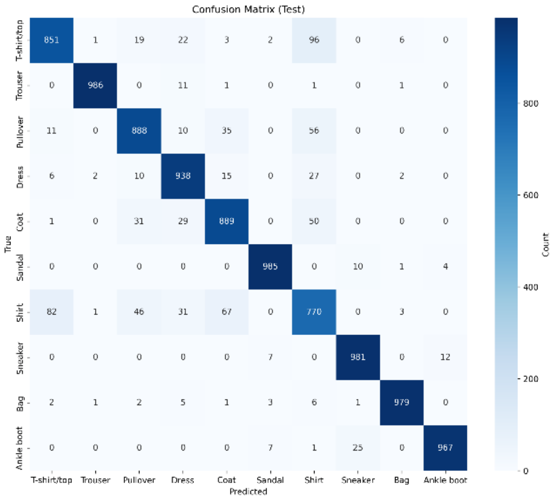
Summary Table of Results:

Model	**Best Val Accuracy**	**Final Val Acc**	**Final Train Acc**	**Test Accuracy**
Baseline (2-conv)	**92.88% (epoch 16)**	92.67%	95.85%	**92.23%**
Deeper (3-conv)	**92.95% (epoch 20)**	92.95%	98.00%	**92.23%**

Both models achieved **the same test accuracy (92.31%)**, but the deeper model required more parameters.

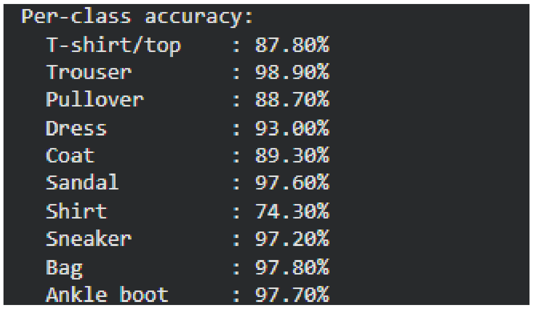
Confusion Matrix & Per-Class Accuracy:

confusion matrix heatmap:


Confusion Matrix (Test)

	T-shirt/top	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot
T-shirt/top	851	1	19	22	3	2	96	0	6	0
Trouser	0	986	0	11	1	0	1	0	1	0
Pullover	11	0	888	10	35	0	56	0	0	0
Dress	6	2	10	938	15	0	27	0	2	0
Coat	1	0	31	29	889	0	50	0	0	0
Sandal	0	0	0	0	0	985	0	10	1	4
Shirt	82	1	46	31	67	0	770	0	3	0
Sneaker	0	0	0	0	0	7	0	981	0	12
Bag	2	1	2	5	1	3	6	1	979	0
Ankle boot	0	0	0	0	0	7	1	25	0	967

Per-class accuracy:

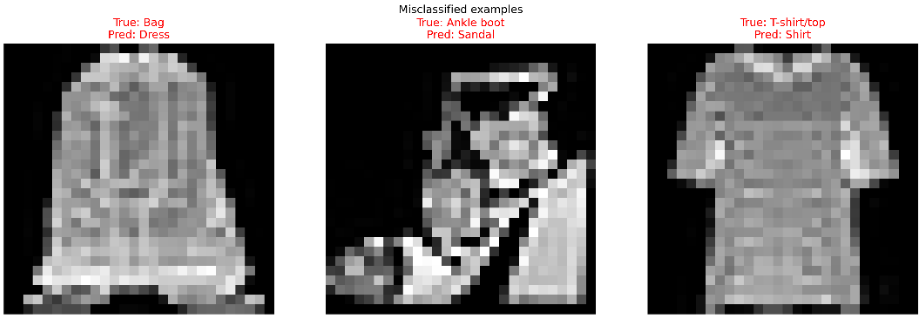


```
Per-class accuracy:
  T-shirt/top    : 87.80%
  Trouser        : 98.90%
  Pullover       : 88.70%
  Dress          : 93.00%
  Coat           : 89.30%
  Sandal         : 97.60%
  Shirt          : 74.30%
  Sneaker        : 97.20%
  Bag            : 97.80%
  Ankle boot     : 97.70%
```

Observations:

The model has the lowest accuracy over the Shirt class and only gets an accuracy of 76.2 percent over this model. On the other hand, all three footwear categories, including Sandal, Sneaker, and Ankle Boot are recognized with exceptionally high precision (more than 98).

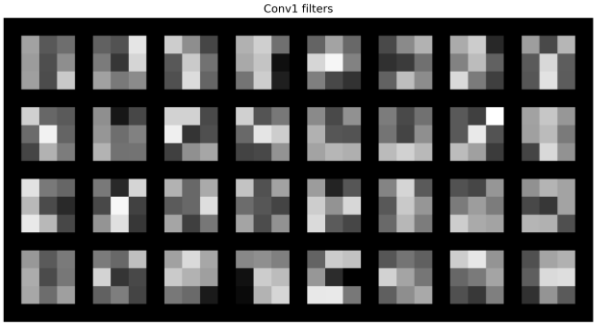
Misclassified Examples:



Observed patterns:

The model is also likely to mix up Shirt with T-shirt/Top, primarily due to the fact that the similarities between the two categories are high in terms of visual representation. It also has some problem with the differentiation between Coat and Pullover as the long sleeves are in both cases and the overall pattern is very similar. Similarly, Dress and Trouser are easily confused because when the image is of lower quality, the lines are less distinct to distinguish.

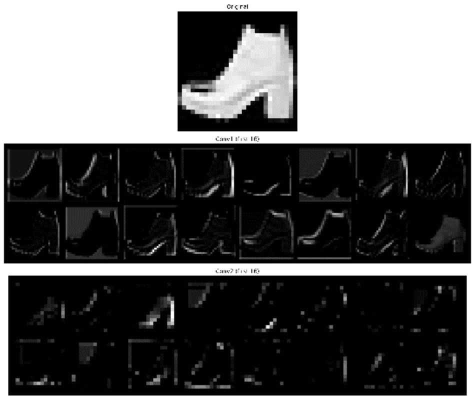
Convolutional Filters:



Filters learned:

- edge detectors
- vertical/horizontal gradients
- simple texture patterns

Feature Maps:



Feature maps show:

- early layers detect edges and outlines
- deeper layers highlight garment structure (e.g., sleeves, waist region)

Discussion:

The deeper network had a little higher accuracy of validation in the depth experiment and both models generalized in a very similar manner. Although the deeper model required almost four times as many parameters, the additional capacity did not result in a substantial gain in test performance. The depth and parameters were not easily converted to gains since Fashion-MNIST is a relatively easy dataset. Both networks also overfitted within approximately 15 epochs. This might be resolved with the introduction of data augmentation, stronger regularization, early stopping, or through scheduling the learning rate. It can be seen, based on the behavior of the model, that it is far simpler to classify footwear and bags as they have sharp, distinctive shapes, whereas different types of tops, including shirts, T-shirts, and pullovers, are far more challenging due to their visual proximity given their low-resolution grayscale images.