Project Write-Up
Fashion-MNIST Classification Using Convolutional Neural Networks

Abstract This project aims to classify images from the Fashion-MNIST dataset using a Convolutional Neural Network (CNN). The CNN model was designed to extract relevant features from the images and classify them into ten distinct clothing categories. Data augmentation techniques were employed to improve the model's generalization capability. The results demonstrate the effectiveness of the CNN architecture in achieving high classification accuracy on this benchmark dataset.

Objective The primary objective of this project is to:

1. Develop a CNN model capable of accurately classifying images from the Fashion-MNIST dataset.
2. Investigate the impact of data augmentation techniques on model performance.
3. Analyze the influence of hyperparameter tuning on the model's accuracy.

Process

Data Preparation and Augmentation The Fashion-MNIST dataset was loaded and preprocessed. To enhance the model's generalization ability, data augmentation techniques were applied, including random horizontal flipping and rotations.

Model Architecture A CNN model was designed with the following architecture:

* **Convolutional Layers:** Multiple convolutional layers were used to extract features from the input images.
* **Batch Normalization:** Batch normalization was applied to stabilize the training process and improve convergence.

* **Pooling Layers:** Max pooling layers were used to reduce the spatial dimensions of the feature maps.
* **Dropout:** Dropout layers were incorporated to prevent overfitting.
* **Fully Connected Layers:** Fully connected layers were used to map the learned features to the output classes.

Training and Evaluation* The model was trained using the Adam optimizer and the Cross-Entropy loss function. The training process involved iterating over multiple epochs, adjusting the model's weights through backpropagation. The model's performance was evaluated on a validation set to monitor overfitting and adjust hyperparameters as needed.

Model Architecture:
A simple CNN architecture was employed to facilitate hyperparameter tuning:

- **Convolutional Layer:** A single convolutional layer with a kernel size of 5 and 8 filters.
- **Activation Function:** ReLU activation function.
- **Loss Function:** Cross-entropy loss.
- **Optimizer:** Adam optimizer.
- **Fully Connected Layers:** Multiple fully connected layers to map the learned features to the output classes.

Results The CNN model achieved a high classification accuracy on the Fashion-MNIST dataset.

Hyperparameter Tuning Experiment
To assess the impact of training epochs, we experimented with three different configurations:

- **Configuration 1: 3 Epochs**
  Results: Final Test Accuracy: 88%
- **Configuration 2: 5 Epochs**
  Results: Final Test Accuracy: 85%
- **Configuration 3: 7 Epochs**
  Results: Final Test Accuracy: 86%

Analysis and Results
Impact of Epochs:

- **3 Epochs:** The model achieved a high accuracy of 88% in a relatively short training time.
- **5 Epochs:** The model's performance slightly decreased to 85% compared to the 3-epoch model.
- **7 Epochs:** The model's performance improved slightly to 86% compared to the 5-epoch model.

View Appendix A at the end of the document

The following code snippet shows a simplified version of the CNN model architecture:

Python Code Snippet

`class CNNModel(nn.Module): def __init__(self): super(CNNModel, self).__init__() # ... (define convolutional layers, batch normalization, pooling, dropout, and fully connected layers) def forward(self, x): # ... (forward pass through the layers) return x`

Conclusion:

The CNN model demonstrated strong performance on the Fashion-MNIST dataset, showcasing the effectiveness of deep learning techniques for image classification tasks. Future work could explore more advanced architectures, such as residual networks or transformers, to further improve performance. Additionally, investigating the impact of different data augmentation techniques and hyperparameter tuning could lead to further optimizations. Based on the experimental results, training the model for 3 epochs appears to be the optimal choice. It offers a good balance between training time and model performance. While increasing the number of epochs can potentially improve accuracy, it might also lead to overfitting.

Future Directions:

- **Deeper Architectures:** Explore deeper CNN architectures with multiple convolutional layers and pooling layers.
- **Data Augmentation:** Implement more sophisticated data augmentation techniques like random cropping and color jittering.
- **Regularization:** Utilize regularization techniques like L1/L2 regularization and dropout to prevent overfitting.
- **Hyperparameter Optimization:** Employ automated hyperparameter tuning techniques like Grid Search or Randomized Search.

By carefully considering these factors, we can further enhance the performance of the CNN model on the Fashion-MNIST dataset.

Appendix A

Epoch 1/3

Test set: Average loss: 0.4532, Accuracy: 8378/10000 (84%)

Epoch 2/3

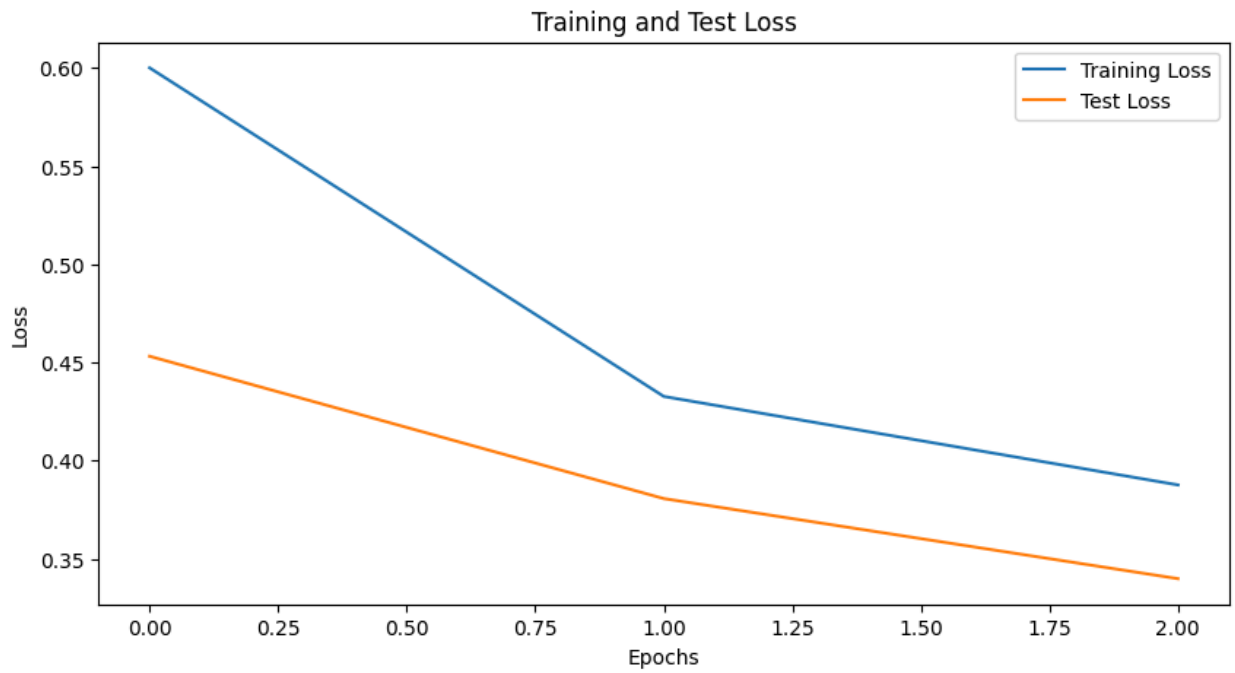
Test set: Average loss: 0.3807, Accuracy: 8594/10000 (86%)

Epoch 3/3

Test set: Average loss: 0.3399, Accuracy: 8760/10000 (88%)

Classification Report:

	precision	recall	f1-score	support
T-shirt/top	0.78	0.87	0.82	1000
Trouser	1.00	0.97	0.98	1000
Pullover	0.76	0.87	0.81	1000
Dress	0.87	0.91	0.89	1000
Coat	0.80	0.79	0.80	1000
Sandal	0.97	0.96	0.97	1000
Shirt	0.73	0.52	0.61	1000
Sneaker	0.92	0.96	0.94	1000
Bag	0.96	0.98	0.97	1000
Ankle boot	0.96	0.94	0.95	1000
accuracy			0.88	10000
macro avg	0.87	0.88	0.87	10000
weighted avg	0.87	0.88	0.87	10000



Model saved as 'fashion_mnist_cnn_full.pth'.

Epoch 1/5

Test set: Average loss: 0.5421, Accuracy: 8116/10000 (81%)

Epoch 2/5

Test set: Average loss: 0.4704, Accuracy: 8319/10000 (83%)

Epoch 3/5

Test set: Average loss: 0.4342, Accuracy: 8404/10000 (84%)

Epoch 4/5

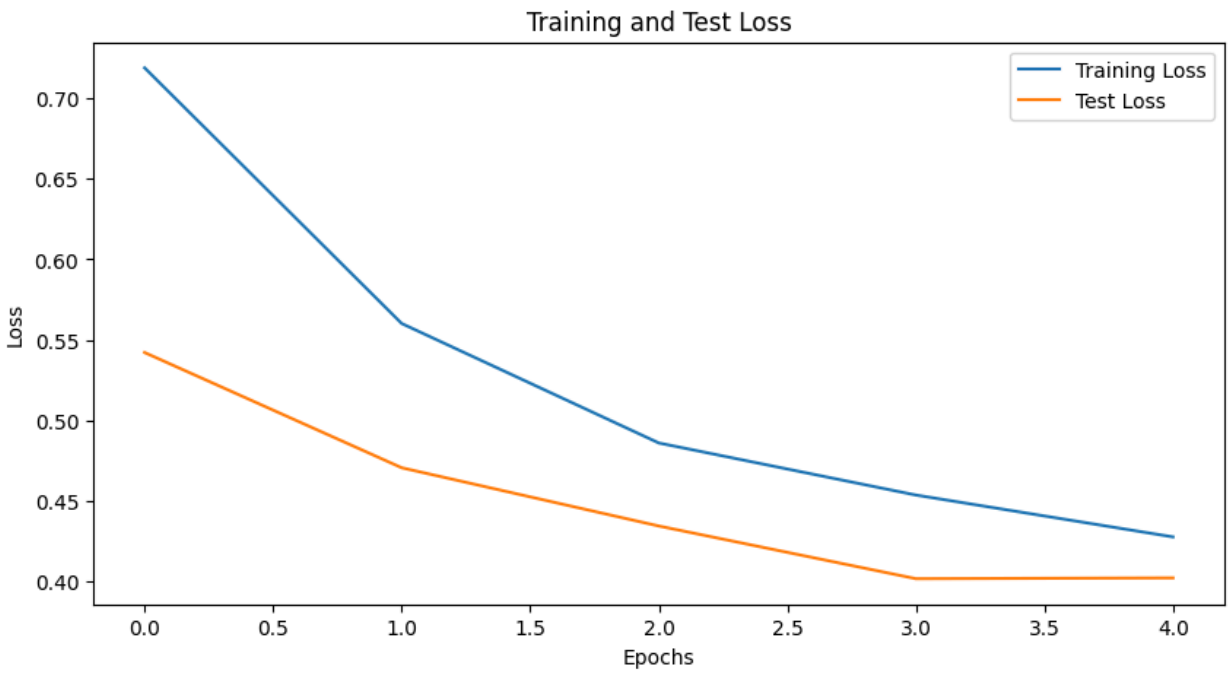
Test set: Average loss: 0.4015, Accuracy: 8504/10000 (85%)

Epoch 5/5

Test set: Average loss: 0.4019, Accuracy: 8487/10000 (85%)

Classification Report:

	precision	recall	f1-score	support
T-shirt/top	0.72	0.88	0.79	1000
Trouser	0.97	0.98	0.98	1000
Pullover	0.64	0.90	0.75	1000
Dress	0.91	0.86	0.88	1000
Coat	0.79	0.68	0.73	1000
Sandal	0.98	0.92	0.95	1000
Shirt	0.70	0.38	0.49	1000
Sneaker	0.90	0.98	0.94	1000
Bag	0.96	0.97	0.96	1000
Ankle boot	0.97	0.94	0.95	1000
accuracy			0.85	10000
macro avg	0.85	0.85	0.84	10000
weighted avg	0.85	0.85	0.84	10000



Model saved as 'fashion_mnist_cnn_full.pth'.

Epoch 1/7

Test set: Average loss: 0.5426, Accuracy: 8065/10000 (81%)

Epoch 2/7

Test set: Average loss: 0.5100, Accuracy: 8144/10000 (81%)

Epoch 3/7

Test set: Average loss: 0.4346, Accuracy: 8403/10000 (84%)

Epoch 4/7

Test set: Average loss: 0.4262, Accuracy: 8425/10000 (84%)

Epoch 5/7

Test set: Average loss: 0.4010, Accuracy: 8543/10000 (85%)

Epoch 6/7

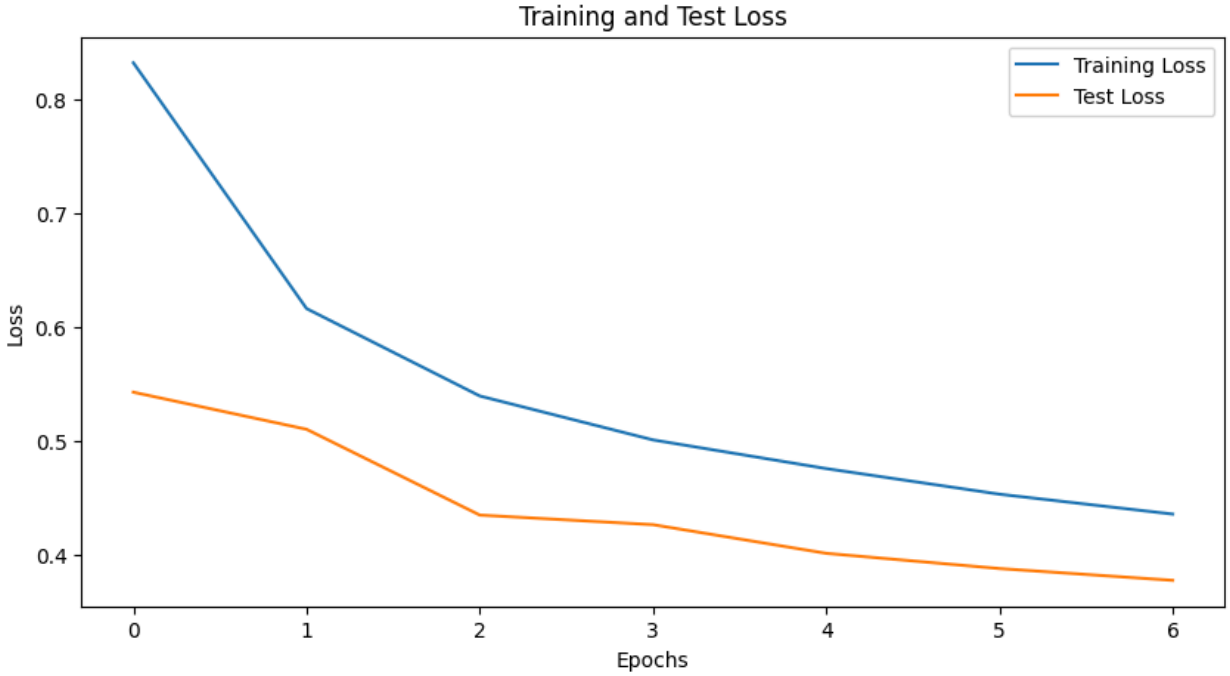
Test set: Average loss: 0.3877, Accuracy: 8581/10000 (86%)

Epoch 7/7

Test set: Average loss: 0.3773, Accuracy: 8603/10000 (86%)

Classification Report:

	precision	recall	f1-score	support
T-shirt/top	0.78	0.81	0.79	1000
Trouser	0.99	0.97	0.98	1000
Pullover	0.77	0.77	0.77	1000
Dress	0.85	0.90	0.88	1000
Coat	0.74	0.82	0.78	1000
Sandal	0.97	0.92	0.94	1000
Shirt	0.68	0.54	0.60	1000
Sneaker	0.90	0.96	0.93	1000
Bag	0.95	0.97	0.96	1000
Ankle boot	0.95	0.95	0.95	1000
accuracy			0.86	10000
macro avg	0.86	0.86	0.86	10000
weighted avg	0.86	0.86	0.86	10000



Model saved as 'fashion_mnist_cnn_full.pth'.