HW4 Report Deep Learning

This report explores the use of transfer learning with a pre-trained ResNet18 model on the Fashion-MNIST dataset. The goal was to leverage the power of a pre-trained model for image classification and compare its performance with a custom Convolutional Neural Network (CNN).

Methodology

Dataset: Fashion-MNIST

Preprocessing:

- images resized to (224x224)
- Grayscale converted to RGB using transforms. Grayscale (num_output_channels=3)
- Normalized using ImageNet mean and standard deviation

Data Augmentation:

- Random horizontal flips, random rotations, color jitter

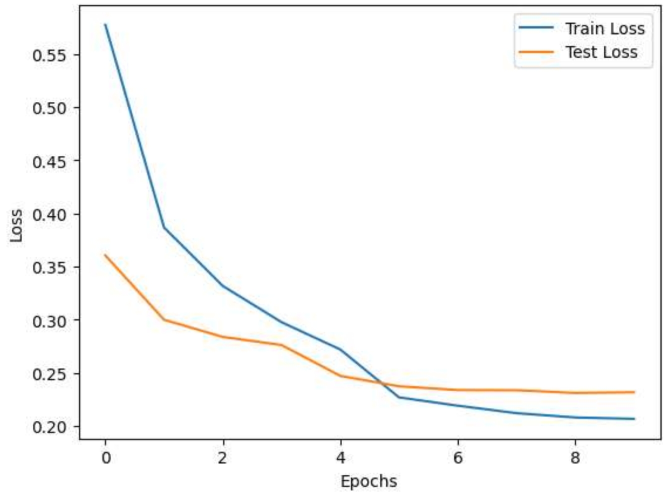
Comparing between CNN and ResNet18

Model	Accuracy	Precision	Recall	F1- Score
CNN	0.9184	0.919	0.916	0.919
ResNet18	0.9352	0.9369	0.9352	0.9348

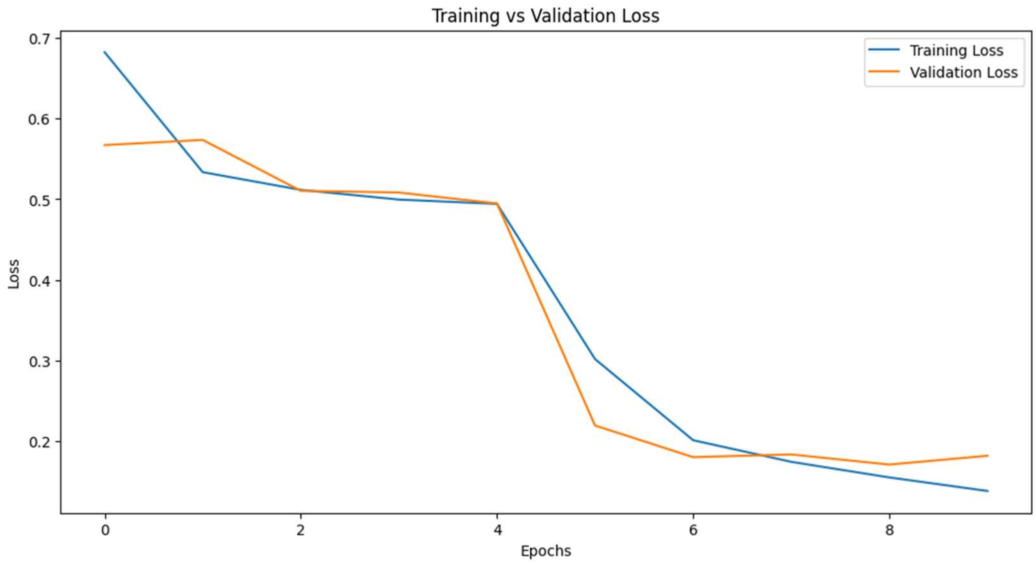
The comparison between the **custom CNN** and the **ResNet18** model on the Fashion-MNIST dataset demonstrates the advantages of transfer learning. The **ResNet18** achieved a higher accuracy of **93.52%** compared to **91.84%** for the CNN, along with superior precision (**93.69% vs. 91.9%**), recall (**93.52% vs. 91.6%**), and F1-score (**93.48% vs. 91.9%**). This improvement can be attributed to the pre-trained weights of ResNet18, which provided a better starting point for feature extraction, leading to faster convergence and enhanced generalization. Meanwhile, the custom CNN, trained from scratch, performed well but lacked the pre-trained advantages seen in the ResNet18 model.

Comparison between losses graphs

CNN Model



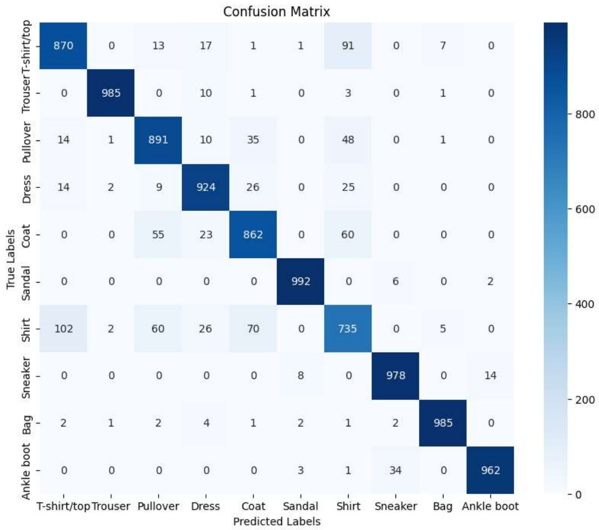
ResNet18 Model



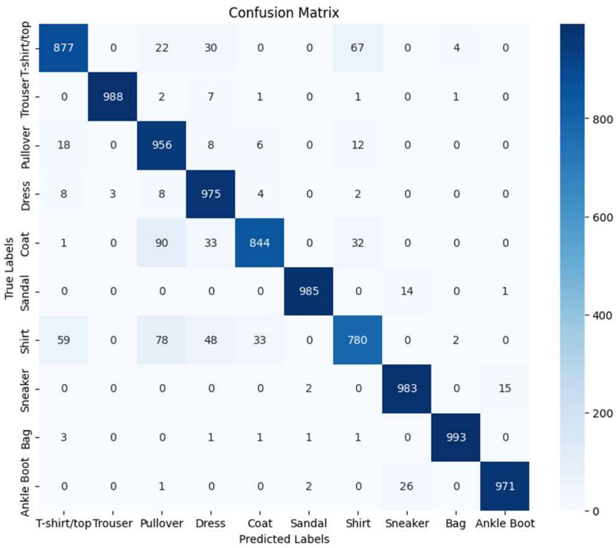
The CNN model shows a rapid decrease in both training and test loss, indicating quick convergence. In contrast, the ResNet18 model, with a layer freeze until the 5th epoch, experiences a slower initial decline in loss. After unfreezing, its training loss decreases more gradually, with less fluctuation in validation loss, suggesting better generalization. Although both models performed similarly by the end, the ResNet18 took longer to reach optimal performance due to the initial layer freeze.

Comparison between confusion matrices (HW3 / HW4)

Confusion matrix of the CNN model (HW3)



Confusion matrix of the ResNet18 model (HW4)



The confusion matrices for both the CNN and ResNet18 models show strong performance on the Fashion-MNIST dataset, but with one notable exception. The ResNet18 model outperformed the CNN in 9 out of 10 classes, demonstrating better overall classification accuracy and fewer misclassifications across most categories. The only exception was the "Coat" class, where the CNN model achieved higher accuracy. Despite this, the ResNet18 model displayed more consistent performance across the remaining classes, indicating its better generalization ability overall.

Conclusion on Transfer Learning Effectiveness

Transfer learning with ResNet18 proved to be highly effective for the Fashion-MNIST dataset. The pre-trained model allowed for faster convergence and achieved higher accuracy compared to a CNN trained from scratch. By leveraging pre-trained weights, the model learned useful features faster, improving generalization on the new dataset.

Scenarios Where Transfer Learning Excels vs. Training from Scratch

- **Small Datasets:** Transfer learning excels when the available training data is limited, as the pre-trained weights already capture general patterns.
- **Time-Constrained Projects:** When faster results are required, transfer learning provides a head start due to pre-trained feature extraction.
- **Complex Image Tasks:** Datasets with complex image patterns benefit from the rich feature representation learned from large datasets like ImageNet.
- **Limited Computational Resources:** Transfer learning reduces the need for extensive computation since fewer layers need to be trained from scratch.

Summary of Pre-trained ResNet Impact:

The use of a pre-trained ResNet model positively impacted the results by:

- Reducing the need for extensive feature learning.
- Improving overall classification metrics such as accuracy, precision, recall, and F1-score.
- Allowing faster convergence while maintaining high performance.

Challenges

- During the model training process, we initially did not store the training and validation losses in lists, which prevented us from generating performance graphs. As a result, we had to modify the code to initialize these lists and rerun the entire training process, which extended the training time by an additional 30 minutes.