NORTHWESTERN UNIVERSITY | SCHOOL OF PROFESSIONAL STUDIES

Assignment 1: Getting Started with CNNs

MSDSP 462-DL – Computer Vision

Sachin Sharma

January 11, 2025

Introduction

Convolutional Neural Networks (CNNs) are specialized deep learning models designed for analysing visual data. They are inspired by the visual cortex of animals and are capable of learning spatial hierarchies of features from images. CNNs are widely applied in tasks such as image classification, object detection, and segmentation. The key components of CNNs include convolutional layers (for feature extraction), pooling layers (for dimensionality reduction), and fully connected layers (for classification). In this project, we utilized CNNs to classify images from the Fashion MNIST dataset into one of ten categories. This dataset is a benchmark for image classification tasks and presents a practical challenge due to the subtle visual similarities between certain classes.

Dataset Overview

The Fashion MNIST dataset is a collection of 70,000 grayscale images of clothing items. Each image has a resolution of 28×28 pixels and belongs to one of the following categories

1. T-shirt/top
2. Trouser
3. Pullover
4. Dress
5. Coat
6. Sandal
7. Shirt
8. Sneaker
9. Bag
10. Ankle boot

Dataset Details:

- **Training Set**: 60,000 images.
- **Test Set**: 10,000 images.
- **Image Format**: Single-channel grayscale images (pixel values range from 00 to 255).
- **Labels**: Integer values (0-9) representing the class categories.

Data Preprocessing:

- Pixel values were normalized to the range [0.0, 1.0] to improve convergence during training.
- The images were reshaped to 28×28×1 to match the input format required by CNNs.
- The labels were one-hot encoded for compatibility with the softmax activation function in the output layer.

Model Architecture

The CNN architecture was specifically designed for this task to balance computational efficiency with classification accuracy. Below are the details of the model:

1. **Input Layer**:
    - Accepts images of shape 28×28×1 (height, width, channels).
2. **Convolutional Layers**:
    - First Convolutional Layer:
        - 56 filters, 3×3 kernel, stride (2,2), ReLU activation, padding = "same".
    - Second Convolutional Layer:
        - 96 filters, 3×3 kernel, stride (1,1), ReLU activation, padding = "same".
    - Third and Fourth Convolutional Layers:
        - 128 filters, 2×2 kernel, stride (1,1), ReLU activation, padding= "same".

3. **Pooling Layers**:
    - MaxPooling layers were applied after the first and second convolutional layers to reduce spatial dimensions.
    - An AveragePooling layer was used after the fourth convolutional layer for final feature aggregation.
4. **Flatten Layer**:
    - Converts the multi-dimensional output of the convolutional layers into a 1D vector for input to the dense layer.
5. **Fully Connected Layer**:
    - A dense layer with 10 units (one for each class) and softmax activation for probability output.
6. **Training Details:**
    - **Optimizer**: RMSprop
    - **Loss Function**: Sparse Categorical Crossentropy
    - **Evaluation Metric**: Accuracy
    - **Epochs:** 20
    - **Batch Size:** 64

Training Results:

The CNN model was trained for **20 epochs**, and the following metrics summarize its performance:

- **Training Accuracy (Epoch 20)**: 98.85%
- **Training Loss (Epoch 20)**: 0.0318
- **Validation Accuracy (Epoch 20)**: 91.59%
- **Validation Loss (Epoch 20)**: 0.5669

Observations:

- The model achieved a high training accuracy of 98.85%, indicating that it effectively learned from the training data.
- The validation accuracy of 91.59% demonstrates strong generalization to unseen data, although there is a noticeable gap between the training and validation accuracy. This suggests that while the model performs well overall, slight overfitting may be present.
- The relatively higher validation loss (0.5669) compared to training loss (0.0318) reinforces the observation of overfitting.

Visualization of Predictions

Test images were displayed with their true and predicted labels. The model's predictions were visually validated, and some errors were observed, particularly for visually ambiguous classes.

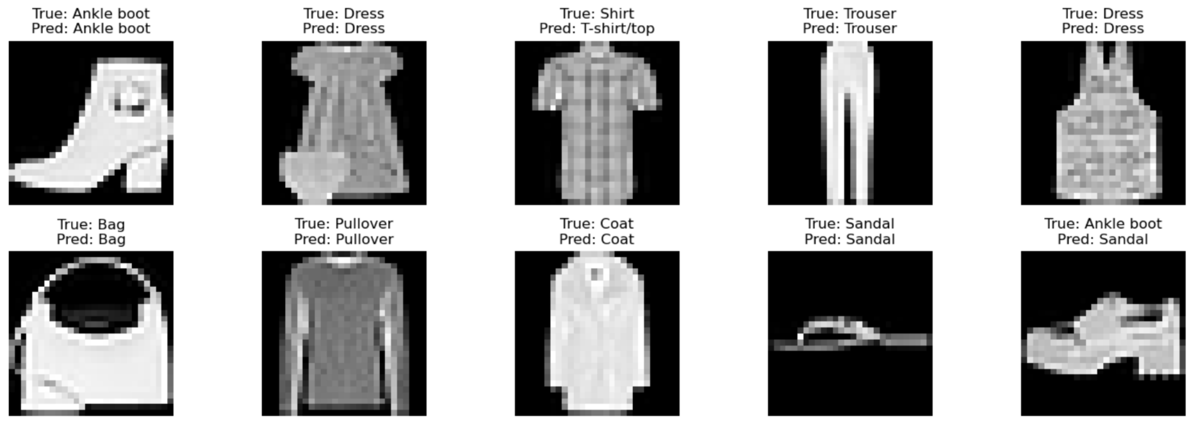
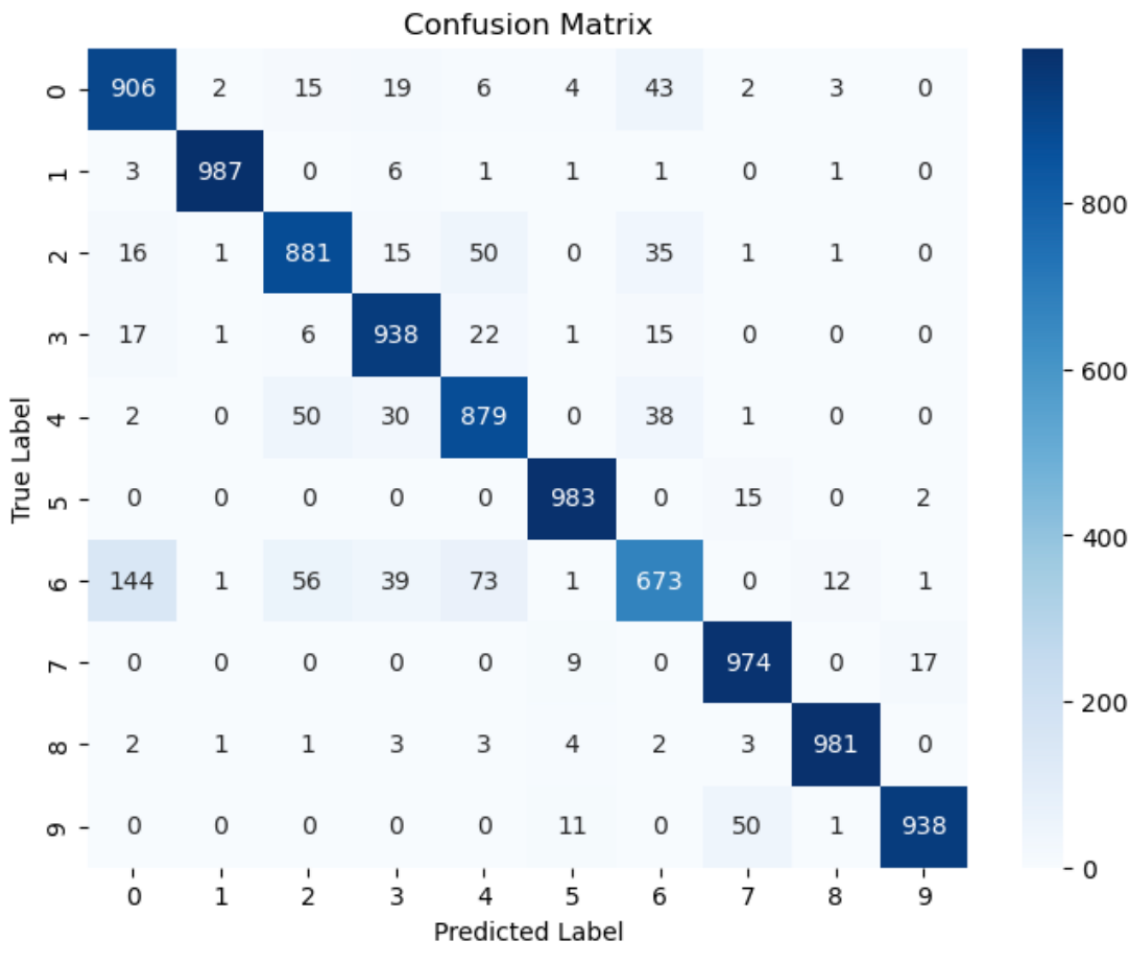


Fig 1: Test Images with their original and Predicted Labels.

Confusion Matrix: A confusion matrix was generated to analyze class-wise performance. Below are the key observations:

- The model performed exceptionally well for distinct classes such as **Shirt** and **Pullover**.
- Some misclassifications were observed between similar classes, such as **Sandal** and **Ankle boot**.



True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	906	2	15	19	6	4	43	2	3	0
1	3	987	0	6	1	1	1	0	1	0
2	16	1	881	15	50	0	35	1	1	0
3	17	1	6	938	22	1	15	0	0	0
4	2	0	50	30	879	0	38	1	0	0
5	0	0	0	0	0	983	0	15	0	2
6	144	1	56	39	73	1	673	0	12	1
7	0	0	0	0	0	9	0	974	0	17
8	2	1	1	3	3	4	2	3	981	0
9	0	0	0	0	0	11	0	50	1	938

Fig 2: Confusion Matrix of trained CNN model

Model Performance

The CNN achieved a robust test accuracy of **91.4%,** which is competitive for the Fashion MNIST dataset. However, the gap between training and validation accuracy suggests some degree of overfitting, which could be mitigated with further improvements.

Ways to Improve Model Accuracy:

1. **Data Augmentation**:
   - Augment the training data with transformations like rotations, zooming, and shifts to improve generalization.
2. **Regularization**:
   - Use Dropout layers to prevent overfitting.
   - Apply L2 regularization to convolutional layers.
3. **Hyperparameter Tuning**:
   - Experiment with different learning rates, optimizers, and batch sizes.
4. **Advanced Architectures**:
   - Incorporate deeper or pre-trained architectures such as ResNet or MobileNet for enhanced feature extraction.
5. **Early Stopping**:
   - Use EarlyStopping to halt training once the validation performance stops improving.

Conclusion

This project successfully demonstrated the use of CNNs for image classification on the Fashion MNIST dataset. The model achieved high accuracy and was able to generalize effectively. By incorporating additional techniques like data augmentation and advanced architectures, the performance can be further optimized.

This study highlights the power of CNNs in solving real-world image classification tasks and provides a foundation for exploring more complex datasets and problems.