TASK 3

FASHION ITEM CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS (CNN)

ARNAV GUPTA – RA2211028010107

SPANDAN BASU CHAUDHURI – RA2211028010120

AIM

The aim of this project is to develop a deep learning model using Convolutional Neural Networks (CNN) to classify images of fashion items from the Fashion-MNIST dataset into their respective categories.

OBJECTIVES

1. To load and preprocess the Fashion-MNIST dataset.
2. To build a CNN-based classification model suitable for grayscale clothing images.
3. To train the model and validate its performance.
4. To visualize accuracy trends during training.
5. To test the model on unseen data and display prediction results.
6. To analyze the model's predictive performance.

INTRODUCTION

Image classification plays an important role in the areas of computer vision, retail automation, and e-commerce product recognition. Deep learning, specifically **Convolutional Neural Networks (CNNs)**, is widely used to classify images based on learned patterns and visual features.

In this project, the **Fashion-MNIST dataset** is used, which consists of 70,000 grayscale images of clothing and accessories across 10 categories. This dataset is a more challenging alternative to the classic MNIST handwritten digit dataset, making it suitable for CNN experimentation.

The CNN model built in this project learns to identify subtle differences in clothing items such as shirts, trousers, sneakers, etc., by extracting spatial patterns from images.

REQUIREMENTS

Hardware Requirements

- Computer with minimum 4 GB RAM
- GPU recommended for faster training (optional)

Software Requirements

Software/Library	**Version**	**Purpose**
Python	3.6+	Programming environment
TensorFlow / Keras	2.x	Model building and training
NumPy	Latest	Numerical operations
Matplotlib	Latest	Data visualization

Dataset

- Built-in **Fashion-MNIST** dataset (loaded via keras.datasets.fashion_mnist)

LIBRARIES USED

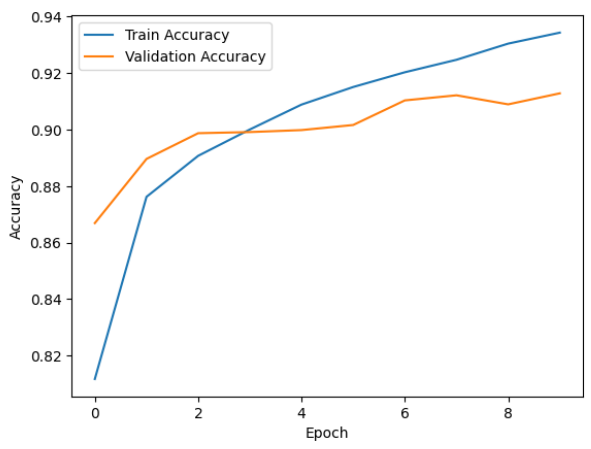
Library	**Purpose**
TensorFlow / Keras	Build and train the CNN model
NumPy	Array manipulation and preprocessing
Matplotlib	Display sample images and training graphs

ALGORITHM / METHODOLOGY

1. **Load Dataset**

- Fashion-MNIST dataset is loaded and split into training and test sets.

2. **Preprocessing**
    - Pixel values are normalized to range **0–1**.
    - A channel dimension is added to match CNN input format (28, 28, 1).
3. **Model Construction**
    - Two convolutional layers extract features.
    - MaxPooling layers reduce spatial dimensions.
    - Flattening converts feature maps to a vector.
    - Dense layers perform classification.
    - Dropout is used to reduce overfitting.
4. **Compilation**
    - Optimizer: **Adam**
    - Loss: **Sparse Categorical Crossentropy**
    - Metric: **Accuracy**
5. **Training**
    - Model is trained for **10 epochs** with validation data.
6. **Evaluation**
    - Test accuracy is computed.
    - Accuracy plot shows model performance trend.
7. **Prediction**
    - Model predicts class labels for test images.
    - True labels vs predicted labels visualized.



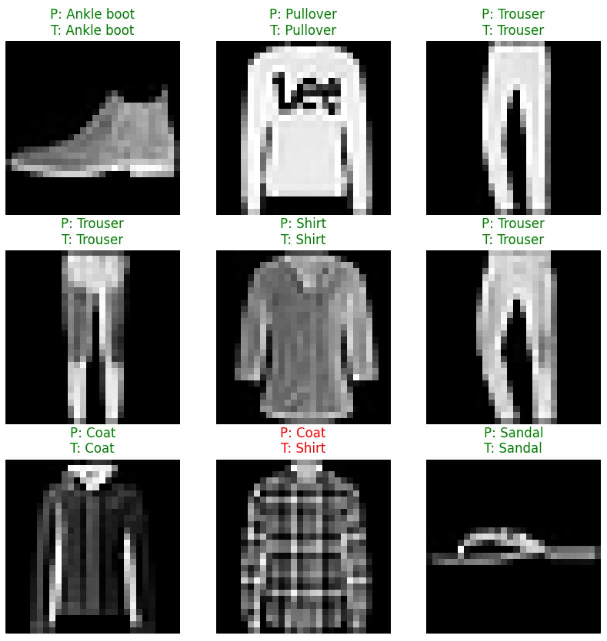
INPUT / OUTPUT

Input	Output
28×28 grayscale image	Predicted clothing category label

Example:

Input: Image of a shoe

Output: Sneaker (Confidence: High)



RESULTS

- The model successfully learned to classify fashion items.

- **Test Accuracy ≈ 0.88–0.92** (depending on system and training conditions).
- Training and validation accuracy curves show smooth learning progression.
- Predictions match actual labels for most test images.

The visualization demonstrates correct predictions in **green** and incorrect ones in **red**, indicating model reliability.

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 4s 12ms/step - accuracy: 0.9122 - loss: 0.2618
Test accuracy: 0.913
```

CONCLUSION

This project demonstrates the effectiveness of **Convolutional Neural Networks** for fashion image classification. The CNN model trained on the Fashion-MNIST dataset achieved strong accuracy and generalization performance. Applying normalization and using multiple convolutional layers significantly enhanced learning efficiency.

The model can be extended to:

- Real-world product recognition systems
- Automated retail checkout systems
- Visual-based search recommendation engines

Future improvement opportunities include:

- Adding Batch Normalization
- Increasing number of filters and layers
- Using Data Augmentation to improve robustness