Fashion-MNIST Image Classification using CNN

1. Objective

The goal of this project is to build a Convolutional Neural Network (CNN) to classify images of clothing items from the Fashion-MNIST dataset.
The project involves model development, data preprocessing, model evaluation, deployment through a Streamlit app, and capturing user feedback for future model improvements.

2. Dataset Description

- Dataset Name: Fashion-MNIST
- Source: Official Fashion-MNIST Dataset
- Data Format: CSV files with 785 columns (1 label + 784 pixel values)
- Number of Classes: 10 clothing categories:
    - T-shirt/top
    - Trouser
    - Pullover
    - Dress
    - Coat
    - Sandal
    - Shirt
    - Sneaker
    - Bag
    - Ankle boot

3. Data Preprocessing

- Loaded and visualized sample images.
- Normalized pixel values (0–255) to (0–1).
- Reshaped data for CNN input (28x28x1).
- One-hot encoded the labels.
- Split the data into Training and Validation sets (80/20 split).

---

4. CNN Model Architecture

- Model Type: Sequential
- Layers:
    - 2 Convolutional layers with ReLU activation and MaxPooling.
    - Dropout layers after each pooling to prevent overfitting.
    - Flattening layer.
    - Dense layer with 128 units.
    - Output layer with 10 units and Softmax activation.
- Optimizer: Adam
- Loss Function: Categorical Crossentropy
- Metrics: Accuracy

---

5. Model Training

- Training without Augmentation:
    - ✓ Achieved Validation Accuracy: ~91%
- Training with Data Augmentation:
    - ✓ Applied random rotation, zoom, width/height shifts.
    - ✓ Generalization improved significantly.

---

6. Model Evaluation

- Test Accuracy: ~91%
- Metrics:
    - Accuracy
    - Precision
    - Recall
    - F1-Score
- Confusion Matrix: Displayed clear separation across classes with minor misclassification.

(Screenshots of confusion matrix and accuracy graph included.)

---

7. Streamlit Interface Development

- Built a simple and user-friendly Streamlit application.
- User can upload an image and get real-time prediction and confidence score.
- Added feedback mechanism:
    - If prediction is wrong, user can select the correct class.
    - Corrected image and label are saved for future model improvements.

---

8. Correction Feedback and Future Retraining

- Corrected images are stored inside /corrections/ folder.
- Corrections are logged into corrections.csv.
- Separate script will be used to convert corrected images into Fashion-MNIST style CSV (flattened 784 pixel rows).
- Allows future fine-tuning of the model on real-world data.

---

9. Learnings

- Handling real-world images with different backgrounds and angles required preprocessing enhancements.
- Data Augmentation significantly improved model robustness.
- Building correction feedback loop added real-world ML project complexity.

---

10. Future Scope

- Retrain the model periodically using collected corrections for continual improvement.
- Upgrade to Transfer Learning using models like MobileNet for better handling of colored real-world images.
- Deploy on cloud platforms like AWS or Azure for public access.

---

11. Screenshots

- Streamlit app home page.
- Sample predictions.
- Correction submission.
- Confusion matrix and evaluation graphs.
- Modal training, accuracy and visualizations

---

12. Conclusion

A complete end-to-end deep learning project is built — covering model development, evaluation, deployment, user interaction, and future improvement.