A Assessment Report on

"Fashion Item Classification"

submitted as partial fullfilment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25 in

CSE(AIML)

By

Shivam Rai (202401100400177)

Tarun Singh (202401100400198)

Vidhu Srivastava (202401100400209)

Vineet Singh (202401100400212)

Vishal Beniwal (202401100400213)

Under the supervision of

Abhishek Shukla

KIET Group of Institutions, Ghaziabad

May, 2025

Introduction

Fashion is a rapidly growing industry where accurate classification of clothing items is essential for improving search engines, recommendation systems, and inventory management. Traditionally, fashion item categorization is done manually or using basic tagging methods, which may not be scalable or consistent for large datasets.

This project aims to build a **machine learning-based Fashion Item Classification System** that automatically identifies the type of clothing item from an image. Using the **Fashion MNIST dataset**, which contains thousands of labeled grayscale images of clothing, the model will learn to categorize items into classes such as T-shirt, Trouser, Pullover, Dress, and more. The system uses image pixel data and deep learning to ensure high accuracy, and its performance will be evaluated using a **confusion matrix** to visualize correct and incorrect predictions.

Methodology

1. Data Collection

The dataset used in this project is Fashion MNIST, which contains 70,000 grayscale images (28x28 pixels) of fashion items. It includes:

- Image data representing various fashion products
- Target labels: Clothing categories such as *T-shirt/top, Trouser, Pullover, Dress, Coat, etc.*

2 Data Preprocessing

- Loaded the dataset using TensorFlow/Keras or from CSVs using Pandas.
- Normalized pixel values by scaling them between 0 and 1.
- Reshaped data to the format suitable for feeding into neural networks.
- Ensured data consistency and verified there were no missing values.

3 Model Selection

We used a Neural Network (Sequential model) built using TensorFlow/Keras, a popular deep learning framework. This model is suitable for image classification tasks due to its ability to learn complex patterns in pixel data.

4 Model Training

1. Split the dataset into training and test sets.
2. Trained the model using the training data.
3. Evaluated model performance using accuracy score, classification report, and a confusion matrix to visualize misclassifications.

5 Crop Recommendation Function

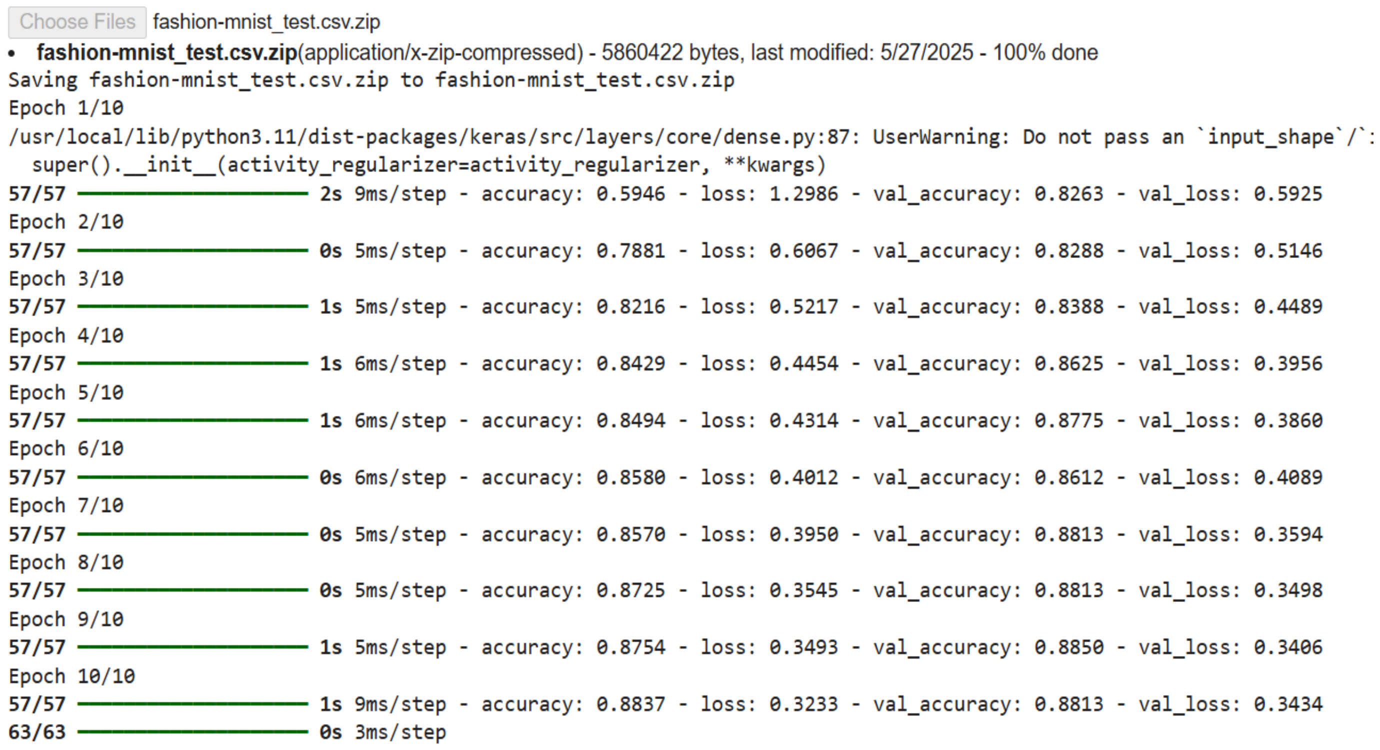
We created a prediction mechanism that takes an image as input, processes it, and returns the predicted fashion item class based on the trained model’s output.

CODE

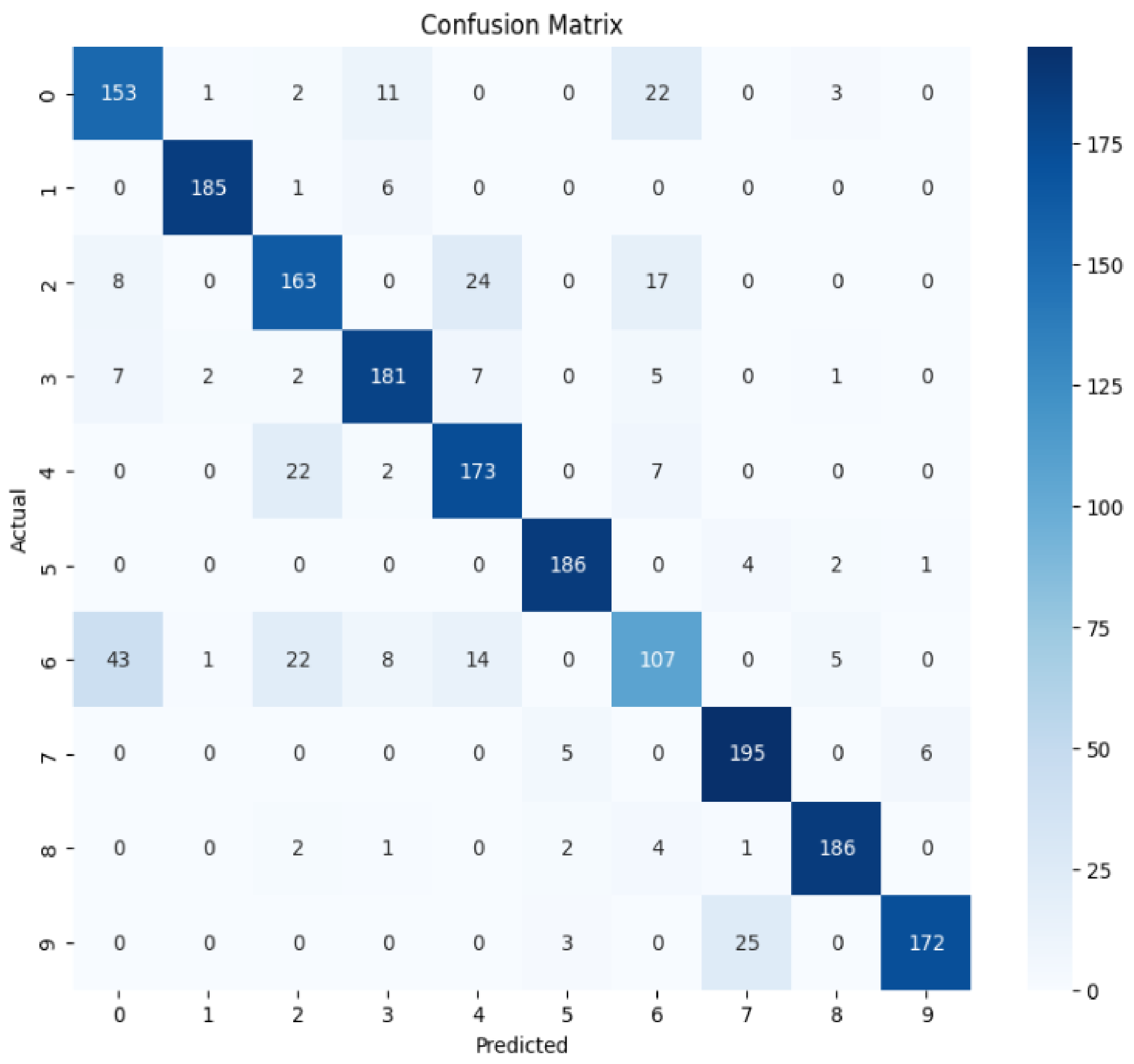
```
import zipfile
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, classification_report
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.utils import to_categorical
from sklearn.model_selection import train_test_split
from google.colab import files
import os

# Step 1: Upload the ZIP file
uploaded = files.upload()
zip_filename = list(uploaded.keys())[0]

# Step 2: Unzip the file
with zipfile.ZipFile(zip_filename, 'r') as zip_ref:
    zip_ref.extractall("unzipped_data")

# Step 3: Read the extracted CSV file
csv_files = os.listdir("unzipped_data")
csv_path = os.path.join("unzipped_data", csv_files[0])
data = pd.read_csv(csv_path)

# Step 4: Prepare data
X = data.iloc[:, 1:].values / 255.0
y = data.iloc[:, 0].values
y_cat = to_categorical(y)
```

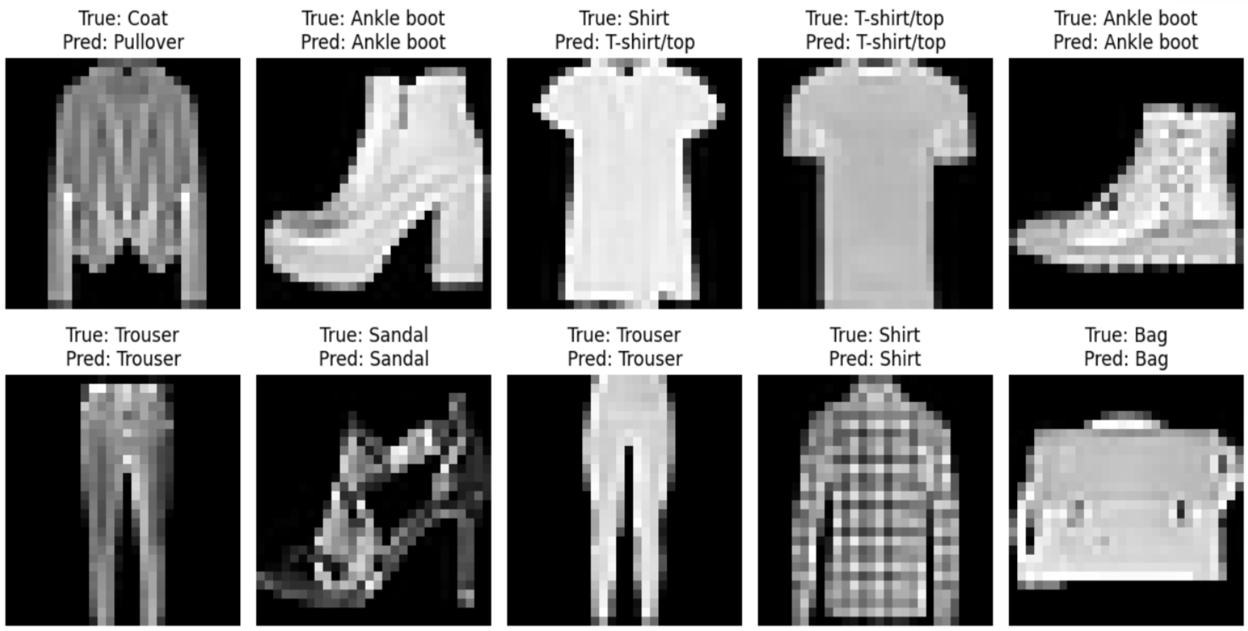
```
X_train, X_test, y_train, y_test = train_test_split(X, y_cat, test_size=0.2, random_state=42)
X_test_images = X_test.reshape(-1, 28, 28)

# Step 5: Build and train model
model = Sequential([
    Dense(128, activation='relu', input_shape=(784,)),
    Dense(64, activation='relu'),
    Dense(10, activation='softmax')
])
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(X_train, y_train, epochs=10, batch_size=128, validation_split=0.1)

# Step 6: Predictions
y_pred_probs = model.predict(X_test)
y_pred = np.argmax(y_pred_probs, axis=1)
y_true = np.argmax(y_test, axis=1)

# Step 7: Confusion Matrix
cm = confusion_matrix(y_true, y_pred)
plt.figure(figsize=(10, 8))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()

# Step 8: Visualize 10 predictions
label_map = {
    0: "T-shirt/top", 1: "Trouser", 2: "Pullover", 3: "Dress", 4: "Coat",
    5: "Sandal", 6: "Shirt", 7: "Sneaker", 8: "Bag", 9: "Ankle boot"
}
plt.figure(figsize=(12, 6))
for i in range(10):
    plt.subplot(2, 5, i+1)
    plt.imshow(X_test_images[i], cmap='gray')
    plt.title(f"True: {label_map[y_true[i]]}\nPred: {label_map[y_pred[i]]}")
    plt.axis('off')
plt.tight_layout()
plt.show()

# Step 9: Classification report
print("Classification Report:")
print(classification_report(y_true, y_pred, target_names=list(label_map.values())))
```

Output



```
Choose Files  fashion-mnist_test.csv.zip
• fashion-mnist_test.csv.zip(application/x-zip-compressed) - 5860422 bytes, last modified: 5/27/2025 - 100% done
Saving fashion-mnist_test.csv.zip to fashion-mnist_test.csv.zip
Epoch 1/10
/usr/local/lib/python3.11/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`:
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
57/57 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - accuracy: 0.5946 - loss: 1.2986 - val_accuracy: 0.8263 - val_loss: 0.5925
Epoch 2/10
57/57 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.7881 - loss: 0.6067 - val_accuracy: 0.8288 - val_loss: 0.5146
Epoch 3/10
57/57 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.8216 - loss: 0.5217 - val_accuracy: 0.8388 - val_loss: 0.4489
Epoch 4/10
57/57 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.8429 - loss: 0.4454 - val_accuracy: 0.8625 - val_loss: 0.3956
Epoch 5/10
57/57 ━━━━━━━━━━━━━━━━━━━━ 1s 6ms/step - accuracy: 0.8494 - loss: 0.4314 - val_accuracy: 0.8775 - val_loss: 0.3860
Epoch 6/10
57/57 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - accuracy: 0.8580 - loss: 0.4012 - val_accuracy: 0.8612 - val_loss: 0.4089
Epoch 7/10
57/57 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.8570 - loss: 0.3950 - val_accuracy: 0.8813 - val_loss: 0.3594
Epoch 8/10
57/57 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.8725 - loss: 0.3545 - val_accuracy: 0.8813 - val_loss: 0.3498
Epoch 9/10
57/57 ━━━━━━━━━━━━━━━━━━━━ 1s 5ms/step - accuracy: 0.8754 - loss: 0.3493 - val_accuracy: 0.8850 - val_loss: 0.3406
Epoch 10/10
57/57 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - accuracy: 0.8837 - loss: 0.3233 - val_accuracy: 0.8813 - val_loss: 0.3434
63/63 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step
```


Confusion Matrix
Actual
Predicted
0 1 2 3 4 5 6 7 8 9
0: 153 1 2 11 0 0 22 0 3 0
1: 0 185 1 6 0 0 0 0 0 0
2: 8 0 163 0 24 0 17 0 0 0
3: 7 2 2 181 7 0 5 0 1 0
4: 0 0 22 2 173 0 7 0 0 0
5: 0 0 0 0 0 186 0 4 2 1
6: 43 1 22 8 14 0 107 0 5 0
7: 0 0 0 0 0 5 0 195 0 6
8: 0 0 2 1 0 2 4 1 186 0
9: 0 0 0 0 0 3 0 25 0 172
175
150
125
100
75
50
25
0



```
Classification Report:
              precision    recall  f1-score   support

 T-shirt/top       0.73      0.80      0.76       192
     Trouser       0.98      0.96      0.97       192
    Pullover       0.76      0.77      0.77       212
       Dress       0.87      0.88      0.87       205
        Coat       0.79      0.85      0.82       204
      Sandal       0.95      0.96      0.96       193
       Shirt       0.66      0.54      0.59       200
     Sneaker       0.87      0.95      0.90       206
         Bag       0.94      0.95      0.95       196
  Ankle boot       0.96      0.86      0.91       200

    accuracy                           0.85      2000
   macro avg       0.85      0.85      0.85      2000
weighted avg       0.85      0.85      0.85      2000
```

■ References

- Dataset Link: https://www.kaggle.com/datasets/zalando-research/fashionmnist
- Scikit-learn Documentation: https://scikit-learn.org/
- Pandas Documentation: https://pandas.pydata.org/
- Streamlit Documentation (for future deployment): https://docs.streamlit.io/

Credits

This Crop Recommendation System project was inspired and made possible by various opensource tools, communities, and resources. Special thanks to:

- **Kaggle** for providing the Crop Recommendation Dataset.
- **Scikit-learn** for its powerful machine learning tools and documentation.
- **Google Colab** for providing a free and accessible cloud-based environment for running Python code.
- **Pandas**, **NumPy**, **Seaborn**, and **Matplotlib** for data manipulation and visualization.
- Online contributors and educators who share tutorials and projects that help learners build practical skills in data science and machine learning.