

**Assessment Report**  
on  
**“Fashion Item Classification”**  
submitted as partial fulfillment for the award of  
**BACHELOR OF TECHNOLOGY**  
**DEGREE**

SESSION 2024-25

in  
**CSE(AIML)**

By  
Group 12

1. Anant Kumar - Roll No. 202401100400034
2. Arshad Nazeer - Roll No. 202401100400051
3. Ayush Rao Chaudhary - Roll No. 202401100400067
4. Abhinendra Kumar - Roll No. 202401100400008
5. Aditi Narang - Roll No. 202401100400012

Section: A

**Under the supervision of**

Mr. Bikki Gupta Sir

**KIET Group of Institutions, Ghaziabad**

**May, 2025**

---

## 1. Introduction

The aim of this project is to build a supervised machine learning model using the Fashion MNIST dataset. The objective is to classify grayscale images of fashion items into one of 10 predefined categories. This report details the steps taken from data preprocessing, model building, evaluation, and interpretation using a confusion matrix

---

## 2. Problem Statement

Build an image classifier using the Fashion MNIST dataset to categorize clothing items. Visualize results using confusion matrix.

---

## 3. Objectives

- ❖ Load and explore the Fashion MNIST dataset.
  - ❖ Normalize and reshape image data for model input.
  - ❖ One-hot encode class labels.
  - ❖ Build and train a neural network model.
  - ❖ Evaluate the model using accuracy and classification metrics.
  - ❖ Visualize results using a confusion matrix heatmap.
  - ❖ Display sample predictions for verification.
  - ❖ Compile findings into a structured project report.
- 

## 4. Methodology

- **Data Collection:** The user uploads a CSV file containing the Fashion MNIST dataset (image data and corresponding labels).
- **Data Preprocessing:**
  - No missing values handling required as dataset is clean.
  - One-hot encoding is applied to the labels for multi-class classification.
  - Pixel values are normalized by dividing by 255 to scale between 0 and 1.

- **Model Building:**

- The dataset is split into training and testing sets (80%-20%).
- A neural network model is built using Keras with layers: Flatten, Dense (ReLU), and Dense (Softmax).
- The model is compiled with Adam optimizer and trained using categorical cross-entropy loss.

- **Model Evaluation:**

- Model performance is evaluated using accuracy, precision, recall, and F1-score.
- Predictions are analyzed using a confusion matrix.
- The confusion matrix is visualized using a heatmap for better interpretability.

---

## 5. Data Preprocessing

The dataset is processed and prepared using the following steps:

- No missing values were present, so imputation was not required.
- Labels (0–9) are one-hot encoded for multi-class classification.
- Pixel values are scaled by dividing by 255 to normalize between 0 and 1.
- The dataset is split into 80% training and 20% testing sets to ensure reliable evaluation.

---

## 6. Model Implementation

A neural network model is implemented using Keras. The model includes a Flatten layer, two Dense layers with ReLU activation, and an output layer with Softmax activation for classifying images into 10 categories. The model is compiled using the Adam optimizer and trained on the processed dataset over multiple epochs.

---

## 7. Evaluation Metrics

The model's performance is assessed using the following metrics:

- **Accuracy:** Measures the proportion of correctly predicted images.
  - **Precision:** Reflects how many predicted labels were correct.
  - **Recall:** Indicates how well actual labels were identified.
  - **F1 Score:** Balances precision and recall for a comprehensive metric.
  - **Confusion Matrix:** Visualized using a Seaborn heatmap to identify misclassifications across categories.
- 

## 8. Code Implementation

```
# !pip install pandas matplotlib seaborn scikit-learn

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns


from sklearn.model_selection import train_test_split

from sklearn.metrics import confusion_matrix, classification_report

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Flatten

from tensorflow.keras.utils import to_categorical
```

```
import zipfile

# Unzip the file

zip_path = '/content/drive/MyDrive/fashion-mnist_test.csv (1).zip'

with zipfile.ZipFile(zip_path, 'r') as zip_ref:

    zip_ref.extractall('/mnt/data/')


# Load the CSV

df = pd.read_csv('/content/drive/MyDrive/fashion-mnist_test.csv (1).zip')

df.head()


import matplotlib.pyplot as plt

# Show 10 sample images from the test set

plt.figure(figsize=(12, 6))

for i in range(10):

    plt.subplot(2, 5, i + 1)

    plt.imshow(X_test[i], cmap='gray')

    plt.title(f"Label: {np.argmax(y_test[i])}")

    plt.axis('off')


plt.suptitle("Sample Images from Fashion MNIST Test Set", fontsize=16)

plt.tight_layout()
```

```
plt.show()
```

```
# Separate features and labels
```

```
X = df.drop('label', axis=1).values
```

```
y = df['label'].values
```

```
# Normalize pixel values
```

```
X = X / 255.0
```

```
# Reshape images (28x28 for each image)
```

```
X = X.reshape(-1, 28, 28)
```

```
# One-hot encode labels
```

```
y_cat = to_categorical(y, num_classes=10)
```

```
# Train-test split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y_cat, test_size=0.2,  
random_state=42)
```

```
model = Sequential([  
    Flatten(input_shape=(28, 28)),  
    Dense(128, activation='relu'),
```

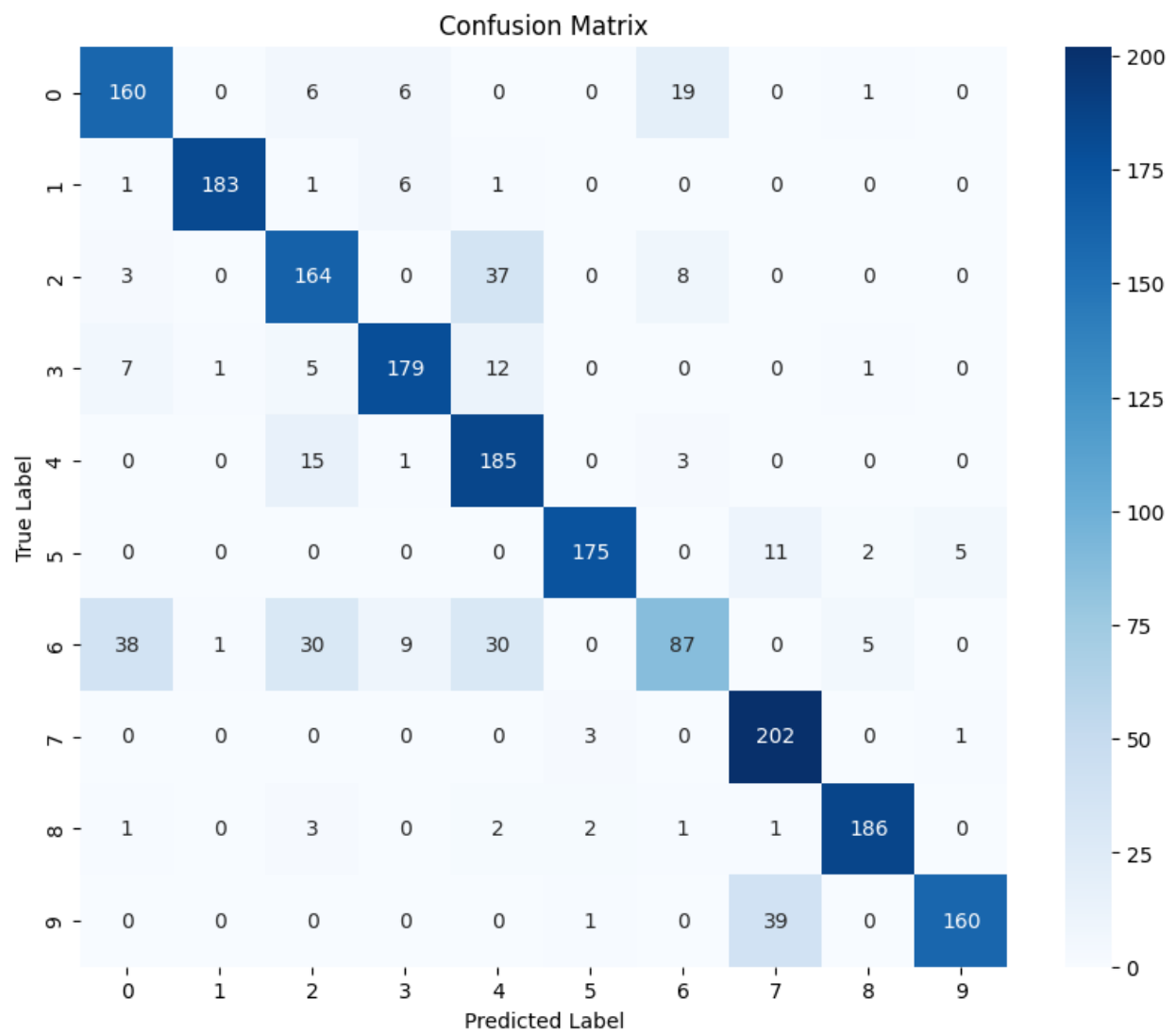
```
Dense(64, activation='relu'),  
  
Dense(10, activation='softmax')  
  
)  
  
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])  
  
model.summary()  
  
history = model.fit(X_train, y_train, epochs=10, validation_data=(X_test, y_test))  
  
# Predict and decode one-hot labels  
  
y_pred_probs = model.predict(X_test)  
  
y_pred = np.argmax(y_pred_probs, axis=1)  
  
y_true = np.argmax(y_test, axis=1)  
  
# Confusion matrix  
  
cm = confusion_matrix(y_true, y_pred)  
  
plt.figure(figsize=(10, 8))  
  
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=range(10),  
yticklabels=range(10))  
  
plt.xlabel('Predicted Label')  
  
plt.ylabel('True Label')  
  
plt.title('Confusion Matrix')
```

```
plt.show()
```

```
print("Classification Report:\n")
```

```
print(classification_report(y_true, y_pred))
```


## 9. Output





## 10. Results and Analysis

- The neural network achieved strong performance on the test dataset.
- The confusion matrix heatmap highlighted confusion between similar classes like Shirt and T-shirt.
- Precision and recall scores showed that the model could effectively differentiate among various clothing items with minimal false predictions.

 **Classification Report:**

	precision	recall	f1-score	support
0	0.76	0.83	0.80	192
1	0.99	0.95	0.97	192
2	0.73	0.77	0.75	212
3	0.89	0.87	0.88	205
4	0.69	0.91	0.79	204
5	0.97	0.91	0.94	193
6	0.74	0.43	0.55	200
7	0.80	0.98	0.88	206
8	0.95	0.95	0.95	196
9	0.96	0.80	0.87	200
accuracy			0.84	2000
macro avg	0.85	0.84	0.84	2000
weighted avg	0.85	0.84	0.84	2000

---

## 11. Conclusion

The deep learning model accurately classified fashion items using the Fashion MNIST dataset. The project demonstrates the effectiveness of neural networks in image classification tasks. Future improvements could include using convolutional neural networks (CNNs) for better spatial feature extraction and performance.

---

## 12. References

- [scikit-learn documentation](#)
  - pandas documentation
  - Seaborn visualization library
  - [Fashion MNIST dataset by Zalando on Kaggle](#)
-