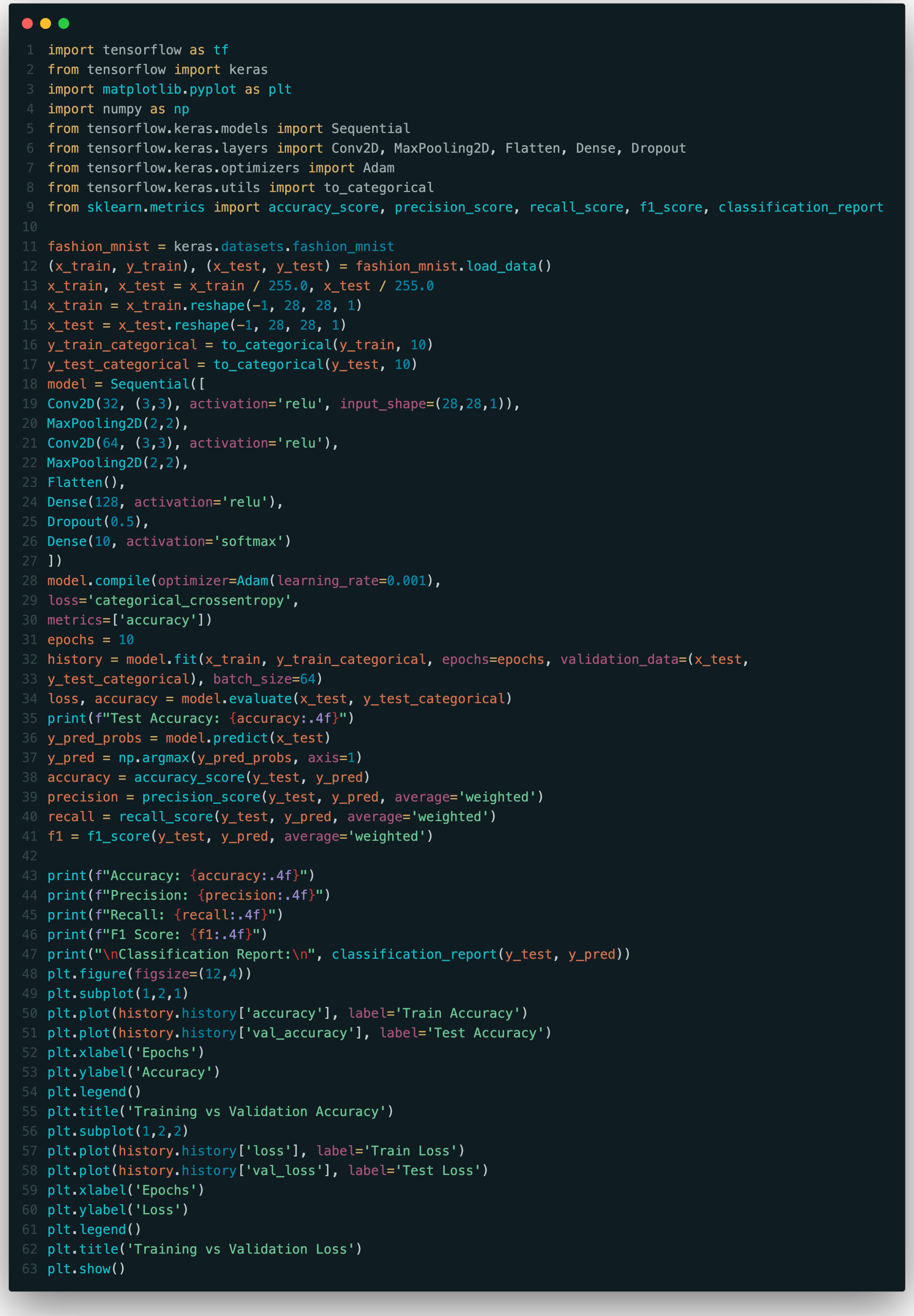
Practical 03

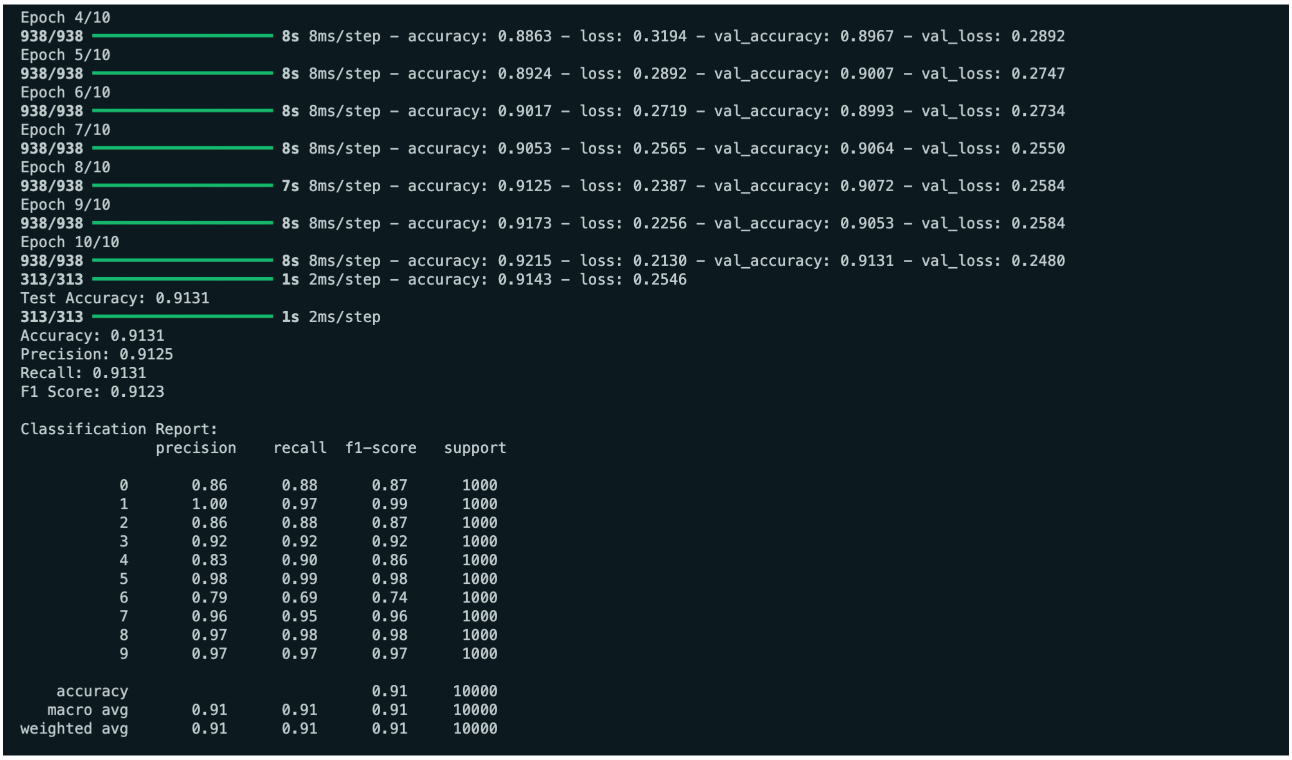


```python
import tensorflow as tf
from tensorflow import keras
import matplotlib.pyplot as plt
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.utils import to_categorical
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report

fashion_mnist = keras.datasets.fashion_mnist
(x_train, y_train), (x_test, y_test) = fashion_mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0
x_train = x_train.reshape(-1, 28, 28, 1)
x_test = x_test.reshape(-1, 28, 28, 1)
y_train_categorical = to_categorical(y_train, 10)
y_test_categorical = to_categorical(y_test, 10)
model = Sequential([
Conv2D(32, (3,3), activation='relu', input_shape=(28,28,1)),
MaxPooling2D(2,2),
Conv2D(64, (3,3), activation='relu'),
MaxPooling2D(2,2),
Flatten(),
Dense(128, activation='relu'),
Dropout(0.5),
Dense(10, activation='softmax')
])
model.compile(optimizer=Adam(learning_rate=0.001),
loss='categorical_crossentropy',
metrics=['accuracy'])
epochs = 10
history = model.fit(x_train, y_train_categorical, epochs=epochs, validation_data=(x_test,
y_test_categorical), batch_size=64)
loss, accuracy = model.evaluate(x_test, y_test_categorical)
print(f"Test Accuracy: {accuracy:.4f}")
y_pred_probs = model.predict(x_test)
y_pred = np.argmax(y_pred_probs, axis=1)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')

print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1 Score: {f1:.4f}")
print("\nClassification Report:\n", classification_report(y_test, y_pred))
plt.figure(figsize=(12,4))
plt.subplot(1,2,1)
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Test Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.title('Training vs Validation Accuracy')
plt.subplot(1,2,2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Test Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.title('Training vs Validation Loss')
plt.show()
```

Output

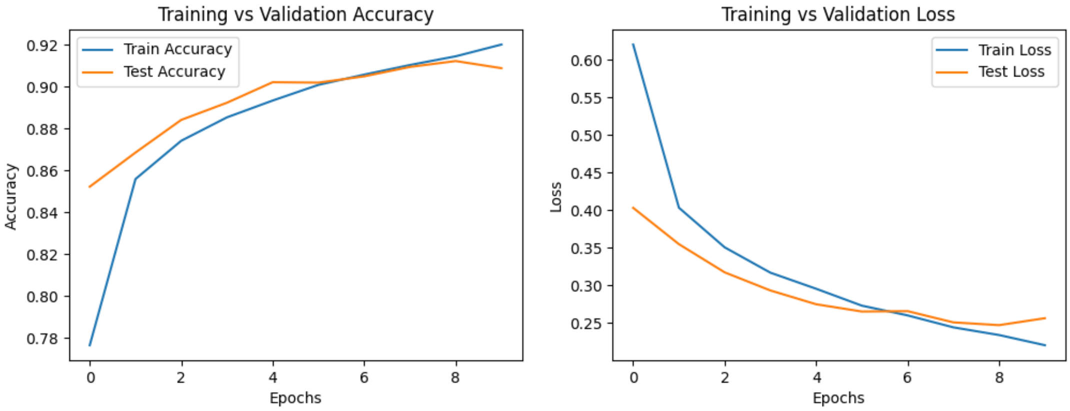


```
Epoch 4/10
938/938 ━━━━━━━━━━━━━━━━━━━━ 8s 8ms/step - accuracy: 0.8863 - loss: 0.3194 - val_accuracy: 0.8967 - val_loss: 0.2892
Epoch 5/10
938/938 ━━━━━━━━━━━━━━━━━━━━ 8s 8ms/step - accuracy: 0.8924 - loss: 0.2892 - val_accuracy: 0.9007 - val_loss: 0.2747
Epoch 6/10
938/938 ━━━━━━━━━━━━━━━━━━━━ 8s 8ms/step - accuracy: 0.9017 - loss: 0.2719 - val_accuracy: 0.8993 - val_loss: 0.2734
Epoch 7/10
938/938 ━━━━━━━━━━━━━━━━━━━━ 8s 8ms/step - accuracy: 0.9053 - loss: 0.2565 - val_accuracy: 0.9064 - val_loss: 0.2550
Epoch 8/10
938/938 ━━━━━━━━━━━━━━━━━━━━ 7s 8ms/step - accuracy: 0.9125 - loss: 0.2387 - val_accuracy: 0.9072 - val_loss: 0.2584
Epoch 9/10
938/938 ━━━━━━━━━━━━━━━━━━━━ 8s 8ms/step - accuracy: 0.9173 - loss: 0.2256 - val_accuracy: 0.9053 - val_loss: 0.2584
Epoch 10/10
938/938 ━━━━━━━━━━━━━━━━━━━━ 8s 8ms/step - accuracy: 0.9215 - loss: 0.2130 - val_accuracy: 0.9131 - val_loss: 0.2480
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.9143 - loss: 0.2546
Test Accuracy: 0.9131
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
Accuracy: 0.9131
Precision: 0.9125
Recall: 0.9131
F1 Score: 0.9123

Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.88      0.87      1000
           1       1.00      0.97      0.99      1000
           2       0.86      0.88      0.87      1000
           3       0.92      0.92      0.92      1000
           4       0.83      0.90      0.86      1000
           5       0.98      0.99      0.98      1000
           6       0.79      0.69      0.74      1000
           7       0.96      0.95      0.96      1000
           8       0.97      0.98      0.98      1000
           9       0.97      0.97      0.97      1000

    accuracy                           0.91     10000
   macro avg       0.91      0.91      0.91     10000
weighted avg       0.91      0.91      0.91     10000
```


Training vs Validation Accuracy
Train Accuracy
Test Accuracy
Accuracy
Epochs
Training vs Validation Loss
Train Loss
Test Loss
Loss
Epochs