Practical 02

```python
import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Bidirectional, Dense, Dropout
from tensorflow.keras.datasets import imdb
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
import matplotlib.pyplot as plt
import numpy as np

vocab_size = 10000
max_length = 250
(X_train, y_train), (X_test, y_test) = imdb.load_data(num_words=vocab_size)
X_train = pad_sequences(X_train, maxlen=max_length, padding='post')
X_test = pad_sequences(X_test, maxlen=max_length, padding='post')
model = Sequential([
Embedding(vocab_size, 256, input_length=max_length),
Bidirectional(LSTM(128, return_sequences=True)),
LSTM(64),
Dense(64, activation='relu'),
Dropout(0.5),
Dense(1, activation='sigmoid')
])
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.0005),
loss='binary_crossentropy',
metrics=['accuracy'])
history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=10,
batch_size=64, verbose=1)
y_pred = model.predict(X_test)
y_pred = (y_pred > 0.5).astype(int)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

print(f"\n🔹 Accuracy: {accuracy:.4f}")
print(f"🔹 Precision: {precision:.4f}")
print(f"🔹 Recall: {recall:.4f}")
print(f"🔹 F1 Score: {f1:.4f}")

plt.figure(figsize=(10,5))
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

```
391/391 ━━━━━━━━━━━━━━━━━━━━ 120s 306ms/step - accuracy: 0.7171 - loss: 0.5302 - val_accuracy: 0.8347 - val_loss: 0.3833
Epoch 6/10
391/391 ━━━━━━━━━━━━━━━━━━━━ 117s 298ms/step - accuracy: 0.8858 - loss: 0.2928 - val_accuracy: 0.8702 - val_loss: 0.3092
Epoch 7/10
391/391 ━━━━━━━━━━━━━━━━━━━━ 116s 298ms/step - accuracy: 0.9326 - loss: 0.1936 - val_accuracy: 0.8756 - val_loss: 0.3117
Epoch 8/10
391/391 ━━━━━━━━━━━━━━━━━━━━ 117s 300ms/step - accuracy: 0.9570 - loss: 0.1315 - val_accuracy: 0.8721 - val_loss: 0.3499
Epoch 9/10
391/391 ━━━━━━━━━━━━━━━━━━━━ 118s 303ms/step - accuracy: 0.9732 - loss: 0.0958 - val_accuracy: 0.8694 - val_loss: 0.4027
Epoch 10/10
391/391 ━━━━━━━━━━━━━━━━━━━━ 118s 303ms/step - accuracy: 0.9781 - loss: 0.0752 - val_accuracy: 0.8680 - val_loss: 0.4391
782/782 ━━━━━━━━━━━━━━━━━━━━ 57s 72ms/step

🔹 Accuracy: 0.8680
🔹 Precision: 0.8529
🔹 Recall: 0.8894
🔹 F1 Score: 0.8707
2025-04-22 11:08:33.968 Python[12480:8538170] +[IMKClient subclass]: chose IMKClient_Modern
2025-04-22 11:08:33.968 Python[12480:8538170] +[IMKInputSession subclass]: chose IMKInputSession_Modern
```