

```

import numpy as np
from keras.datasets import imdb
from keras.preprocessing import sequence
from keras.models import Sequential
from keras.layers import Embedding, LSTM, Dense, Bidirectional

# --- 1. Load and Prepare Data (from Lab 6) ---
vocabulary_size = 5000
max_words = 500

print(f"Loading data with vocabulary size = {vocabulary_size}...")
(X_train, y_train), (X_test, y_test) =
imdb.load_data(num_words=vocabulary_size)

print(f"Padding sequences to max length = {max_words}...")
X_train = sequence.pad_sequences(X_train, maxlen=max_words)
X_test = sequence.pad_sequences(X_test, maxlen=max_words)

# --- 2. Build and Train Model 1: Standard LSTM ---
print("\n--- Building Model 1: Standard LSTM ---")
embedding_size = 32
model_lstm = Sequential()
model_lstm.add(Embedding(vocabulary_size, embedding_size,
input_length=max_words))
model_lstm.add(LSTM(100))
model_lstm.add(Dense(1, activation='sigmoid'))
print(model_lstm.summary())

# Compile the model
model_lstm.compile(loss='binary_crossentropy',
optimizer='adam',
metrics=['accuracy'])

# Train the model
batch_size = 64
num_epochs = 3
print("\n--- Training Model 1: Standard LSTM ---")
model_lstm.fit(X_train, y_train,
validation_data=(X_test, y_test),
batch_size=batch_size,
epochs=num_epochs)

# --- 3. Build and Train Model 2: Bidirectional LSTM (BiLSTM) ---
print("\n--- Building Model 2: Bidirectional LSTM (BiLSTM) ---")
model_bilstm = Sequential()
model_bilstm.add(Embedding(vocabulary_size, embedding_size,
input_length=max_words))
model_bilstm.add(Bidirectional(LSTM(100))) # The only change is
wrapping LSTM in Bidirectional
model_bilstm.add(Dense(1, activation='sigmoid'))

```

```

print(model_bilstm.summary())

# Compile the model
model_bilstm.compile(loss='binary_crossentropy',
                      optimizer='adam',
                      metrics=['accuracy'])

# Train the model
print("\n--- Training Model 2: BiLSTM ---")
model_bilstm.fit(X_train, y_train,
                 validation_data=(X_test, y_test),
                 batch_size=batch_size,
                 epochs=num_epochs)

# --- 4. Evaluate and Compare Results ---
print("\n--- Final Results Comparison ---")

scores_lstm = model_lstm.evaluate(X_test, y_test, verbose=0)
print(f"Standard LSTM Test Accuracy: {scores_lstm[1]*100:.2f}%")

scores_bilstm = model_bilstm.evaluate(X_test, y_test, verbose=0)
print(f"Bidirectional LSTM (BiLSTM) Test Accuracy: {scores_bilstm[1]*100:.2f}%")

Loading data with vocabulary size = 5000...
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/imdb.npz
17464789/17464789 _____ 6s 0us/step
Padding sequences to max length = 500...

--- Building Model 1: Standard LSTM ---

/opt/anaconda3/lib/python3.12/site-packages/keras/src/layers/core/embedding.py:97: UserWarning: Argument `input_length` is deprecated. Just remove it.
  warnings.warn(

Model: "sequential"

```

Layer (type) Param #	Output Shape	
embedding (Embedding) (unbuilt)	?	0
lstm (LSTM) (unbuilt)	?	0

Layer (type)	Output Shape	Param #
dense (Dense) (unbuilt)	?	0

Total params: 0 (0.00 B)

Trainable params: 0 (0.00 B)

Non-trainable params: 0 (0.00 B)

None

--- Training Model 1: Standard LSTM ---

Epoch 1/3

391/391 111s 283ms/step - accuracy: 0.6730 -

loss: 0.5924 - val_accuracy: 0.7279 - val_loss: 0.5331

Epoch 2/3

391/391 115s 293ms/step - accuracy: 0.8349 -

loss: 0.3790 - val_accuracy: 0.8727 - val_loss: 0.3055

Epoch 3/3

391/391 163s 417ms/step - accuracy: 0.9025 -

loss: 0.2489 - val_accuracy: 0.8785 - val_loss: 0.3114

--- Building Model 2: Bidirectional LSTM (BiLSTM) ---

Model: "sequential_1"

Layer (type) Param #	Output Shape	
embedding_1 (Embedding) (unbuilt)	?	0
bidirectional (Bidirectional) (unbuilt)	?	0
dense_1 (Dense) (unbuilt)	?	0

Total params: 0 (0.00 B)

Trainable params: 0 (0.00 B)

Non-trainable params: 0 (0.00 B)

None

--- Training Model 2: BiLSTM ---

Epoch 1/3

391/391 _____ 268s 683ms/step - accuracy: 0.6943 -
loss: 0.5577 - val_accuracy: 0.8204 - val_loss: 0.4068

Epoch 2/3

391/391 _____ 199s 507ms/step - accuracy: 0.8645 -
loss: 0.3243 - val_accuracy: 0.8482 - val_loss: 0.3468

Epoch 3/3

391/391 _____ 182s 465ms/step - accuracy: 0.8974 -
loss: 0.2653 - val_accuracy: 0.8714 - val_loss: 0.3195

--- Final Results Comparison ---

Standard LSTM Test Accuracy: 87.85%