dl-lab-exp-no-2

May 9, 2025

[1]:
```python
import numpy as np
import tensorflow as tf
from tensorflow.keras.datasets import imdb
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Embedding, LSTM, Dropout
from tensorflow.keras.optimizers import Adam
import matplotlib.pyplot as plt
```

[2]:
```python
(X_train, y_train), (X_test, y_test) = imdb.load_data(num_words=10000)
max_length = 200
X_train = pad_sequences(X_train, maxlen=max_length)
X_test = pad_sequences(X_test, maxlen=max_length)
print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
```

```
X_train shape: (25000, 200)
X_test shape: (25000, 200)
```

[3]:
```python
model = Sequential()
model.add(Embedding(input_dim=10000, output_dim=128))
model.add(LSTM(units=128, dropout=0.2, recurrent_dropout=0.2))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='adam', loss='binary_crossentropy',␣
 ↪metrics=['accuracy'])
model.summary()
```

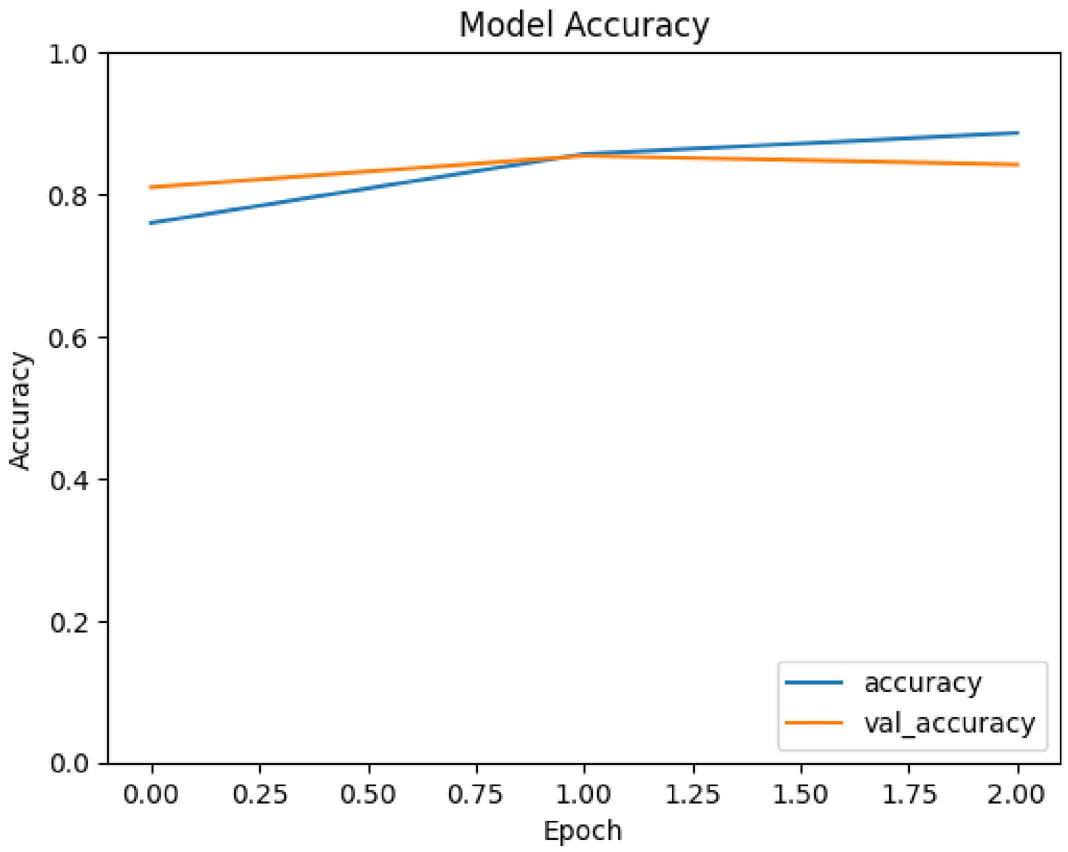
```
Model: "sequential"


 Layer (type)                        Output Shape                       ␣
↪Param #

 embedding (Embedding)               ?                               0␣
↪(unbuilt)
```

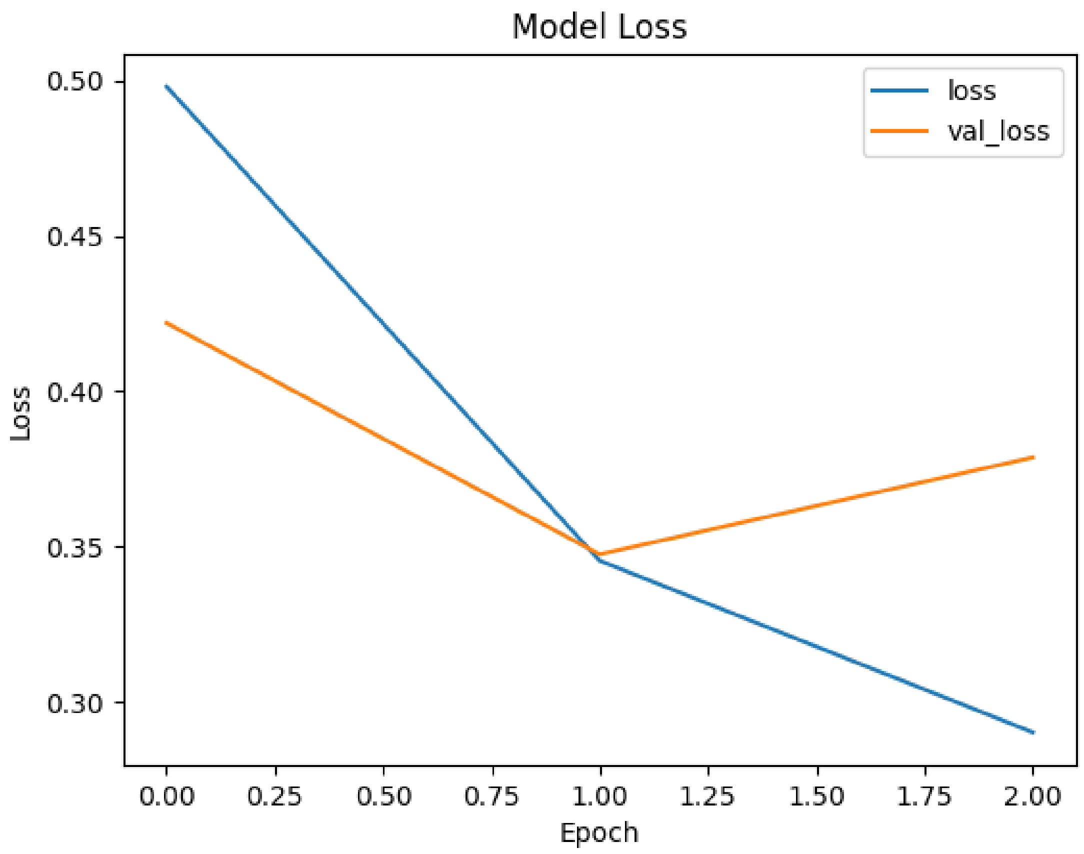
```
 lstm (LSTM)                         ?                                  0␣
↪(unbuilt)

 dropout (Dropout)                   ?                                     ␣
↪  0

 dense (Dense)                       ?                                  0␣
↪(unbuilt)



Total params: 0 (0.00 B)


Trainable params: 0 (0.00 B)


Non-trainable params: 0 (0.00 B)
```

[4]:
```
history = model.fit(X_train, y_train, epochs=3, batch_size=64,␣
 ↪validation_data=(X_test, y_test))
```

```
Epoch 1/3
391/391            247s 614ms/step -
accuracy: 0.6883 - loss: 0.5713 - val_accuracy: 0.8099 - val_loss: 0.4219
Epoch 2/3
391/391            207s 530ms/step -
accuracy: 0.8571 - loss: 0.3483 - val_accuracy: 0.8542 - val_loss: 0.3476
Epoch 3/3
391/391            191s 487ms/step -
accuracy: 0.8914 - loss: 0.2800 - val_accuracy: 0.8419 - val_loss: 0.3787
```

[5]:
```
plt.plot(history.history['accuracy'], label='accuracy')
plt.plot(history.history['val_accuracy'], label = 'val_accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.ylim([0, 1])
plt.legend(loc='lower right')
plt.title('Model Accuracy')
plt.show()

plt.plot(history.history['loss'], label='loss')
plt.plot(history.history['val_loss'], label = 'val_loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend(loc='upper right')
plt.title('Model Loss')
plt.show()
```


Model Accuracy
Accuracy
Epoch
1.0
0.8
0.6
0.4
0.2
0.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
accuracy
val_accuracy



[6]:
```
test_loss, test_acc = model.evaluate(X_test, y_test, verbose=2)
print(f"Test accuracy: {test_acc}")
```

```
782/782 - 40s - 51ms/step - accuracy: 0.8419 - loss: 0.3787
Test accuracy: 0.841920018196106
```

[7]:
```
from sklearn.metrics import classification_report, confusion_matrix
import seaborn as sns
```

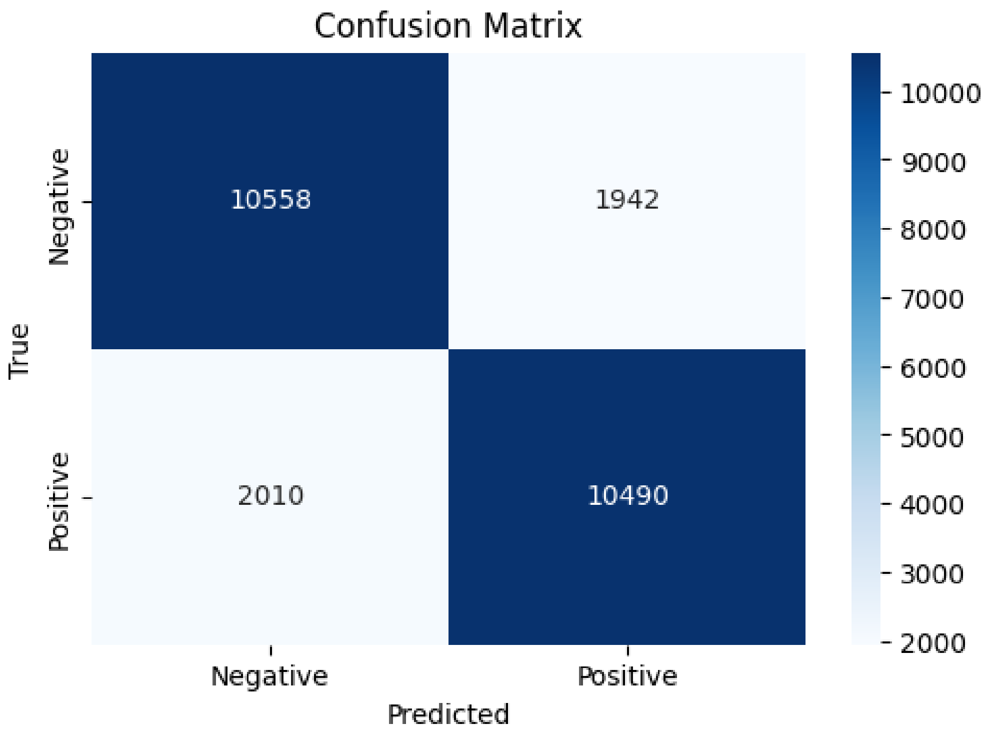
[8]:
```
# Get predictions
y_pred = (model.predict(X_test) >= 0.5).astype(int).flatten()
y_test = y_test.flatten()
```

```
782/782                 40s 50ms/step
```

[9]:
```
# Classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred, target_names=['Negative',␣
  ↪'Positive']))
```

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

    Negative       0.84      0.84      0.84     12500
    Positive       0.84      0.84      0.84     12500

    accuracy                           0.84     25000
   macro avg       0.84      0.84      0.84     25000
weighted avg       0.84      0.84      0.84     25000
```

[10]:
```
# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Negative',␣
 ↪'Positive'], yticklabels=['Negative', 'Positive'])
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()
```


Confusion Matrix
True
Negative
Positive
Negative
Positive
Predicted