Assignment 10.4

Using listing 6.46 in Deep Learning with Python as a guide, fit the same data with a simple 1D convnet. Produce the model performance metrics and training and validation accuracy curves within the Jupyter notebook.

In [1]:
```
import os
import numpy as np
import matplotlib.pyplot as plt
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, Conv1D, MaxPooling1D, GlobalMaxPooling1

# Set the paths to the train and test datasets
train_dir = 'C:/Users/chris/DSC650-T301/dsc/dsc650/data/external/imdb/aclImdb/train'
test_dir = 'C:/Users/chris/DSC650-T301/dsc/dsc650/data/external/imdb/aclImdb/test'

# Load the train dataset
train_texts = []
train_labels = []
for label_type in ['neg', 'pos']:
    dir_name = os.path.join(train_dir, label_type)
    for fname in os.listdir(dir_name):
        if fname.endswith('.txt'):
            with open(os.path.join(dir_name, fname), encoding='utf-8') as f:
                train_texts.append(f.read())
            train_labels.append(0 if label_type == 'neg' else 1)

# Load the test dataset
test_texts = []
test_labels = []
for label_type in ['neg', 'pos']:
    dir_name = os.path.join(test_dir, label_type)
    for fname in os.listdir(dir_name):
        if fname.endswith('.txt'):
            with open(os.path.join(dir_name, fname), encoding='utf-8') as f:
                test_texts.append(f.read())
            test_labels.append(0 if label_type == 'neg' else 1)

# Tokenize the text data
max_words = 10000
tokenizer = Tokenizer(num_words=max_words)
tokenizer.fit_on_texts(train_texts)
train_sequences = tokenizer.texts_to_sequences(train_texts)
test_sequences = tokenizer.texts_to_sequences(test_texts)

# Pad the sequences
maxlen = 200
x_train = pad_sequences(train_sequences, maxlen=maxlen)
x_test = pad_sequences(test_sequences, maxlen=maxlen)
y_train = np.array(train_labels)
y_test = np.array(test_labels)
```

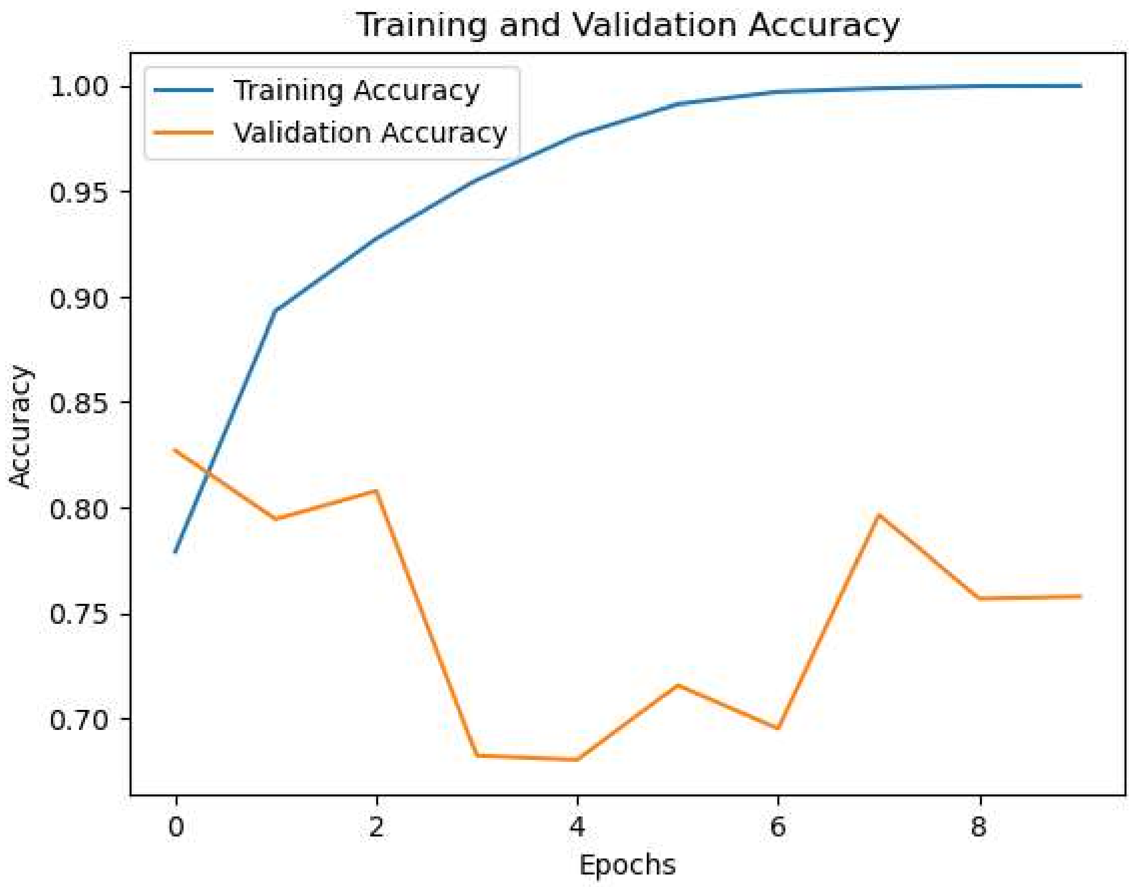
```
# Build the model
embedding_dim = 100
model = Sequential()
model.add(Embedding(max_words, embedding_dim, input_length=maxlen))
model.add(Conv1D(32, 7, activation='relu'))
model.add(MaxPooling1D(5))
model.add(Conv1D(32, 7, activation='relu'))
model.add(GlobalMaxPooling1D())
model.add(Dense(1, activation='sigmoid'))

# Compile and train the model
model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=['accuracy'])
history = model.fit(x_train, y_train, epochs=10, batch_size=32, validation_split=0.2)

# Evaluate the model
loss, accuracy = model.evaluate(x_test, y_test)
print(f'Test loss: {loss:.4f}')
print(f'Test accuracy: {accuracy:.4f}')

# Plot the training and validation accuracy curves
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

```
Epoch 1/10
625/625 [==============================] - 21s 32ms/step - loss: 0.4470 - accuracy:
0.7789 - val_loss: 0.3946 - val_accuracy: 0.8270
Epoch 2/10
625/625 [==============================] - 20s 32ms/step - loss: 0.2618 - accuracy:
0.8932 - val_loss: 0.4784 - val_accuracy: 0.7944
Epoch 3/10
625/625 [==============================] - 21s 33ms/step - loss: 0.1910 - accuracy:
0.9275 - val_loss: 0.4627 - val_accuracy: 0.8078
Epoch 4/10
625/625 [==============================] - 19s 30ms/step - loss: 0.1311 - accuracy:
0.9554 - val_loss: 0.8683 - val_accuracy: 0.6822
Epoch 5/10
625/625 [==============================] - 20s 31ms/step - loss: 0.0754 - accuracy:
0.9766 - val_loss: 1.0635 - val_accuracy: 0.6802
Epoch 6/10
625/625 [==============================] - 20s 32ms/step - loss: 0.0351 - accuracy:
0.9913 - val_loss: 1.0536 - val_accuracy: 0.7156
Epoch 7/10
625/625 [==============================] - 18s 29ms/step - loss: 0.0129 - accuracy:
0.9970 - val_loss: 1.4811 - val_accuracy: 0.6950
Epoch 8/10
625/625 [==============================] - 18s 28ms/step - loss: 0.0042 - accuracy:
0.9987 - val_loss: 1.0305 - val_accuracy: 0.7964
Epoch 9/10
625/625 [==============================] - 19s 31ms/step - loss: 9.5748e-04 - accurac
y: 0.9998 - val_loss: 1.5573 - val_accuracy: 0.7568
Epoch 10/10
625/625 [==============================] - 20s 31ms/step - loss: 6.0088e-04 - accurac
y: 0.9998 - val_loss: 1.6825 - val_accuracy: 0.7578
782/782 [==============================] - 5s 6ms/step - loss: 0.9650 - accuracy: 0.8
412
Test loss: 0.9650
Test accuracy: 0.8412
```


Training and Validation Accuracy
Training Accuracy
Validation Accuracy
Accuracy
Epochs
1.00
0.95
0.90
0.85
0.80
0.75
0.70
0
2
4
6
8

In [ ]: