Assignment 10

May 20, 2023

0.1 10.1.a

[2]:
```
import re

# function to tokenize text
def tokenize(sentence):
    text = re.sub('[^A-Za-z0-9 ]+', '', sentence)
    tokens = list(set(text.split(" ")))
    return tokens

test_str = "Research has determined 70% of #laughter is an accident. That's the␣
 ↪reason!"
print(tokenize(test_str))
```

```
['of', 'Research', 'determined', 'laughter', 'reason', 'has', 'accident',
'Thats', 'an', 'is', 'the', '70']
```

0.2 10.1.b

[3]:
```
tokens = tokenize(test_str)

# function to splits tokens into N-grams
def ngram(tokens, n):
    ngrams = []
    for i in range(len(tokens) - n + 1):
        ngrams.append(' '.join(tokens[i:i+n]))
    return ngrams

print(ngram(tokens,3))
```

```
['of Research determined', 'Research determined laughter', 'determined laughter
reason', 'laughter reason has', 'reason has accident', 'has accident Thats',
'accident Thats an', 'Thats an is', 'an is the', 'is the 70']
```

0.3 10.1.c

[4]:
```
tokens = tokenize(test_str)

# function to create a vector from a numerical vector from a list of tokens.
def one_hot_encode(tokens, num_words):
    token_index = {}
    results = []

    for i in range(len(tokens)):
        if tokens[i] not in token_index:
            token_index[tokens[i]] = i
        encoded = [0]*num_words
        encoded[token_index[tokens[i]]] = 1
        results.append(encoded)
    return results

one_hot_encode(tokens, 12)
```

[4]:
```
[[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]]
```

[ ]:
```

```

0.4 10.2 Word Embedding

[1]:
```
from pathlib import Path
import os
import pandas as pd


# Setting dataframe path
imdb_path = os.path.abspath('/Users/mithilpatel/Desktop/DSC_650/data/external/
 ↪imdb/aclImdb/')
```

[2]:
```
# Importing train dataset and converting to a pandas dataframe
pos_train_path = str(Path(imdb_path).joinpath('train/pos'))
```

```
neg_train_path = str(Path(imdb_path).joinpath('train/neg'))

# Function to read data from each files
def text_to_frame(folder_path):
    # Get a list of all subdirectories within the main folder
    file_content = []

    for file_name in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file_name)

        if os.path.isfile(file_path):
            with open(file_path, 'r') as f:
                content = f.read()
            file_content.append(content)

    return file_content

# Creating a dataframe with text and review
df_pos = pd.DataFrame({'text': text_to_frame(pos_train_path), 'review': [1] *␣
 ↪len(text_to_frame(pos_train_path))})
df_neg = pd.DataFrame({'text': text_to_frame(neg_train_path), 'review': [0] *␣
 ↪len(text_to_frame(neg_train_path))})

# merging both dataframe
df_train = pd.concat([df_pos, df_neg], ignore_index=True)
df_train = df_train.sample(frac=1, random_state=42).reset_index(drop=True)
df_train.head(10)
```

[2]:

	text	review
0	Great little thriller. I was expecting some ty…	1
1	Nothing could have saved this movie, not even …	0
2	This was a good movie. It wasn't your typical …	1
3	From the pen of Richard Condon (The Manchurian…	0
4	I suppose that today this film has relevance b…	0
5	Some guys think that sniper is not good becaus…	1
6	Che: Part One was a fascinating experiment, wh…	1
7	I found Horrorvision almost unwatchable. While…	0
8	I really enjoyed this movie. It succeeded in d…	1
9	Eric Rohmer's "The Lady and the Duke". could h…	1

[3]:
```
# Importing test dataset and converting to a pandas dataframe
pos_test_path = str(Path(imdb_path).joinpath('test/pos'))
neg_test_path = str(Path(imdb_path).joinpath('test/neg'))

# Creating dataframe using test dataset
df_pos_test = pd.DataFrame({'text': text_to_frame(pos_test_path), 'review': [1]␣
 ↪* len(text_to_frame(pos_test_path))})
```

```
df_neg_test = pd.DataFrame({'text': text_to_frame(neg_test_path), 'review': [0]␣
 ↪* len(text_to_frame(neg_test_path))})

# merging dataframes
df_test = pd.concat([df_pos_test, df_neg_test], ignore_index=True)
df_test = df_test.sample(frac=1, random_state=42).reset_index(drop=True)
df_test.head(10)
```

[3]:

	text	review
0	Yul Brynner was a symbol of villein in the tin…	1
1	This show has been performed live around the c…	0
2	To sum this story up in a few sentences: A tee…	1
3	This is absolutely beyond question the worst m…	0
4	A box with a button provides a couple with the…	0
5	First off let me say that this is probably in …	1
6	Albert Finney and Tom Courtenay are brilliant …	1
7	I don't know about the rest of the viewers of …	0
8	Sick of the current cinema output, particularl…	1
9	This movie is beautiful in all ways. It is vis…	1

[4]:
```
# Splitting test set into test and validation
test_split = int(len(df_test)*0.50)
df_testset = df_test.iloc[:test_split,:]
df_validation = df_test.iloc[test_split:,:]

# Splitting into training and testing variables
x_train = df_train['text']
y_train = df_train['review']

x_val = df_validation['text']
y_val = df_validation['review']

x_test = df_testset['text']
y_test = df_testset['review']
```

[5]:
```
from keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
import numpy as np

# Tokenizing text
tokenizer = Tokenizer()
tokenizer.fit_on_texts(x_train)

# Convert text to sequences
x_train = tokenizer.texts_to_sequences(x_train)  # Convert text to sequences of␣
 ↪integers
x_val = tokenizer.texts_to_sequences(x_val)
```

```
x_test = tokenizer.texts_to_sequences(x_test)

max_length = 100

# Apply padding to the tokenized sequences
x_train_padded = pad_sequences(x_train, maxlen=max_length, padding='post')
x_val_padded = pad_sequences(x_val, maxlen=max_length, padding='post')
x_test_padded = pad_sequences(x_test, maxlen=max_length, padding='post')
```

```
2023-05-14 13:14:18.227515: I tensorflow/core/platform/cpu_feature_guard.cc:182]
This TensorFlow binary is optimized to use available CPU instructions in
performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild
TensorFlow with the appropriate compiler flags.
```

[6]:
```
len(tokenizer.word_index)
```

[6]:
```
88582
```

[7]:
```
from keras.models import Sequential
from keras.layers import Embedding, Flatten, Dense

max_words = 88582
embedding_dim = 100

# Word embedding model architecture
model = Sequential()
model.add(Embedding(max_words, embedding_dim, input_length= max_length))
model.add(Flatten())
model.add(Dense(32, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.summary()
model.compile(optimizer='rmsprop',
              loss='binary_crossentropy',
              metrics=['acc'])

# training the model
history = model.fit(x_train_padded, y_train,
                    epochs=10,
                    batch_size=32,
                    validation_data=(x_val_padded, y_val))
```

```
Epoch 1/10
782/782 [==============================] - 36s 46ms/step - loss: 0.4373 - acc:
0.7849 - val_loss: 0.3516 - val_acc: 0.8426
Epoch 2/10
782/782 [==============================] - 36s 46ms/step - loss: 0.1537 - acc:
0.9439 - val_loss: 0.4660 - val_acc: 0.8122
```

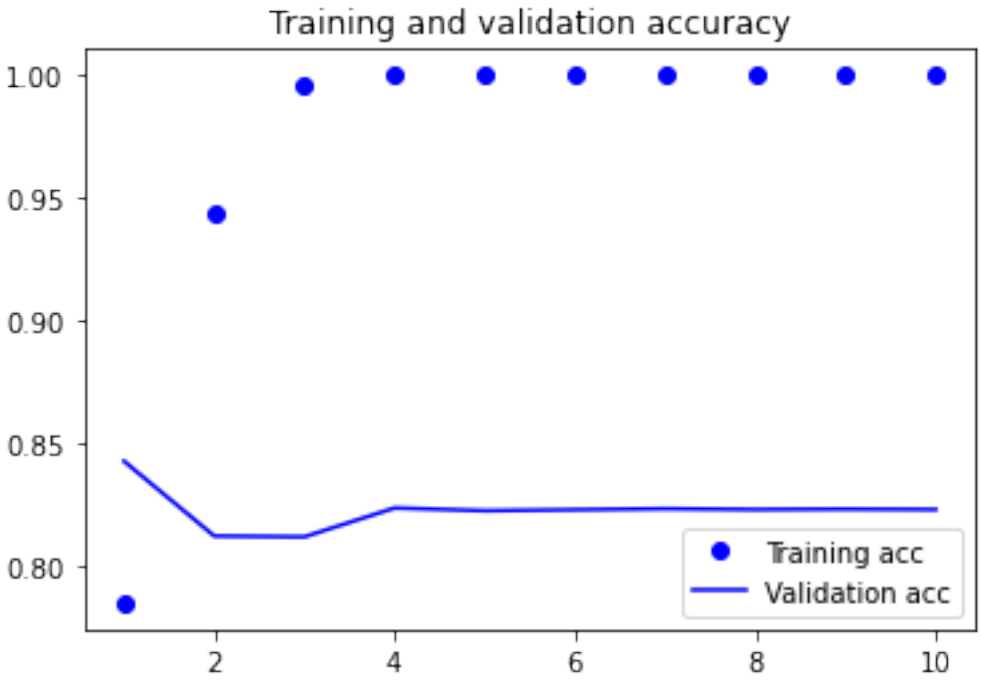
```
Epoch 3/10
782/782 [==============================] - 38s 49ms/step - loss: 0.0175 - acc:
0.9960 - val_loss: 0.6596 - val_acc: 0.8118
Epoch 4/10
782/782 [==============================] - 39s 50ms/step - loss: 0.0015 - acc:
0.9998 - val_loss: 0.7687 - val_acc: 0.8236
Epoch 5/10
782/782 [==============================] - 40s 51ms/step - loss: 1.4540e-04 -
acc: 1.0000 - val_loss: 0.8228 - val_acc: 0.8225
Epoch 6/10
782/782 [==============================] - 41s 53ms/step - loss: 3.2849e-05 -
acc: 1.0000 - val_loss: 0.8643 - val_acc: 0.8229
Epoch 7/10
782/782 [==============================] - 42s 54ms/step - loss: 1.8217e-05 -
acc: 1.0000 - val_loss: 0.8665 - val_acc: 0.8233
Epoch 8/10
782/782 [==============================] - 42s 54ms/step - loss: 1.3150e-05 -
acc: 1.0000 - val_loss: 0.8797 - val_acc: 0.8230
Epoch 9/10
782/782 [==============================] - 40s 51ms/step - loss: 1.0162e-05 -
acc: 1.0000 - val_loss: 0.8922 - val_acc: 0.8231
Epoch 10/10
782/782 [==============================] - 38s 49ms/step - loss: 8.6209e-06 -
acc: 1.0000 - val_loss: 0.9018 - val_acc: 0.8230
```

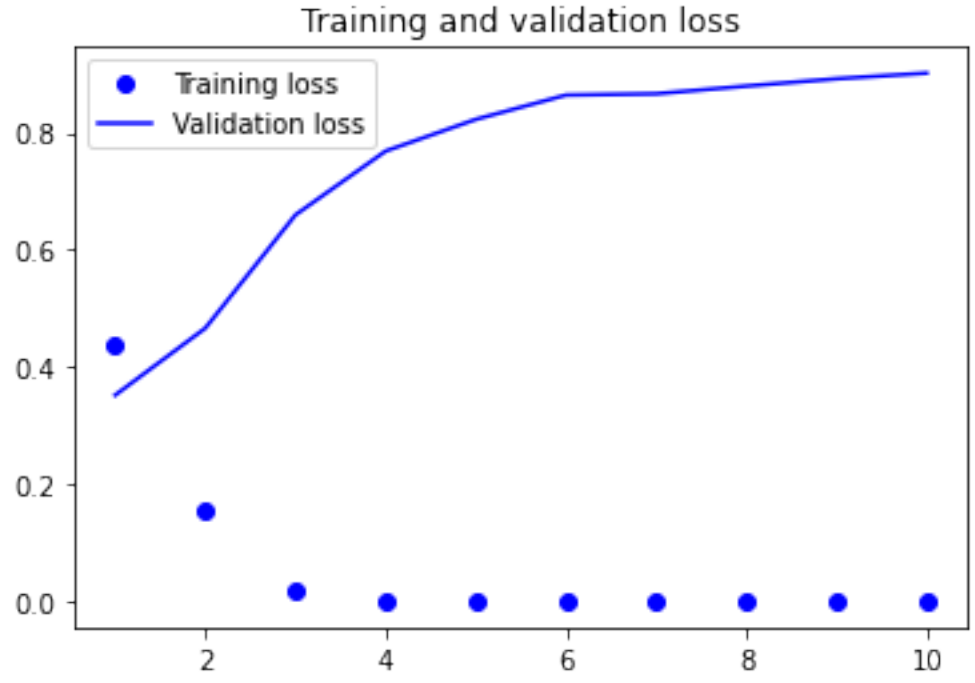
[8]:
```python
# Calculating accuracy
test_loss, test_acc = model.evaluate(x_test_padded, y_test)
print("Model Accuracy:", test_acc)
```

```
391/391 [==============================] - 1s 3ms/step - loss: 0.8909 - acc:
0.8250
Model Accuracy: 0.8250399827957153
```

[9]:
```python
import matplotlib.pyplot as plt

# Creating training and validation loss and accuracy curves
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(acc) + 1)
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()
plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
```

```
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()
plt.show()
```





[ ]:

0.5 10.3 LSTM layer

[11]:
```
import warnings
warnings.filterwarnings("ignore")

from keras.layers import LSTM

# model architecture
model = Sequential()
model.add(Embedding(max_words, embedding_dim, input_length= max_length))
model.add(LSTM(32)) # Adding LSTM layer
model.add(Dense(1, activation='sigmoid'))
model.compile(optimizer='rmsprop',
              loss='binary_crossentropy',
              metrics=['acc'])

# training the model
history = model.fit(x_train_padded, y_train,
                    epochs=10,
                    batch_size=32,
                    validation_data=(x_val_padded, y_val))

test_loss, test_acc = model.evaluate(x_test_padded, y_test)
print("Model Accuracy:", test_acc)
```

```
Epoch 1/10
```

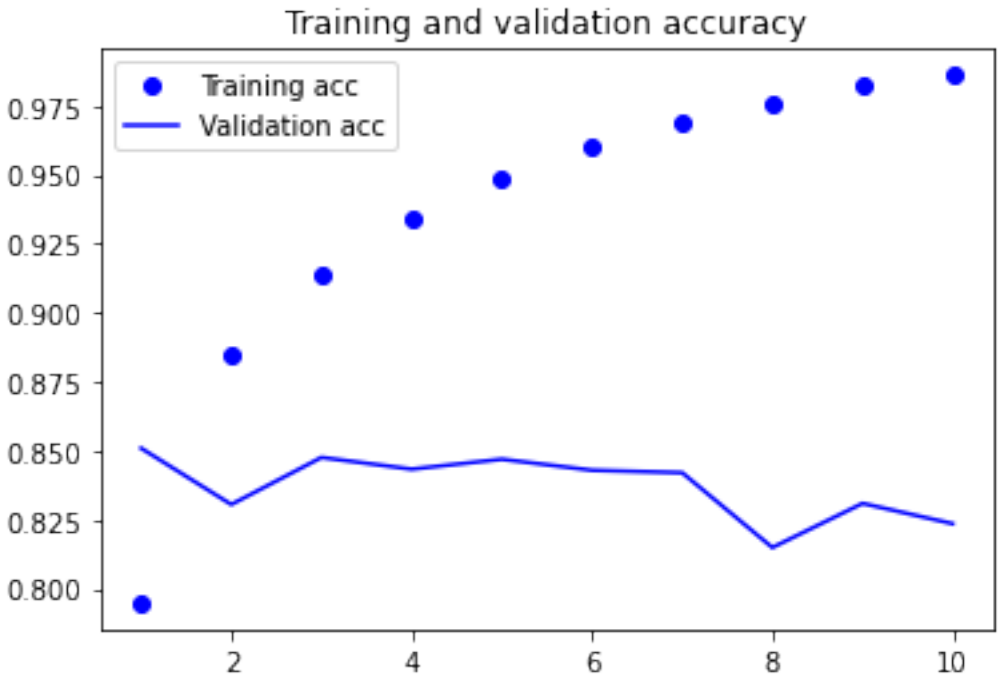
```
2023-05-14 13:38:45.413704: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_2_grad/concat/split_2/split_dim' with
dtype int32
         [[{{node gradients/split_2_grad/concat/split_2/split_dim}}]]
2023-05-14 13:38:45.416641: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_grad/concat/split/split_dim' with dtype
int32
         [[{{node gradients/split_grad/concat/split/split_dim}}]]
2023-05-14 13:38:45.419194: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_1_grad/concat/split_1/split_dim' with
```

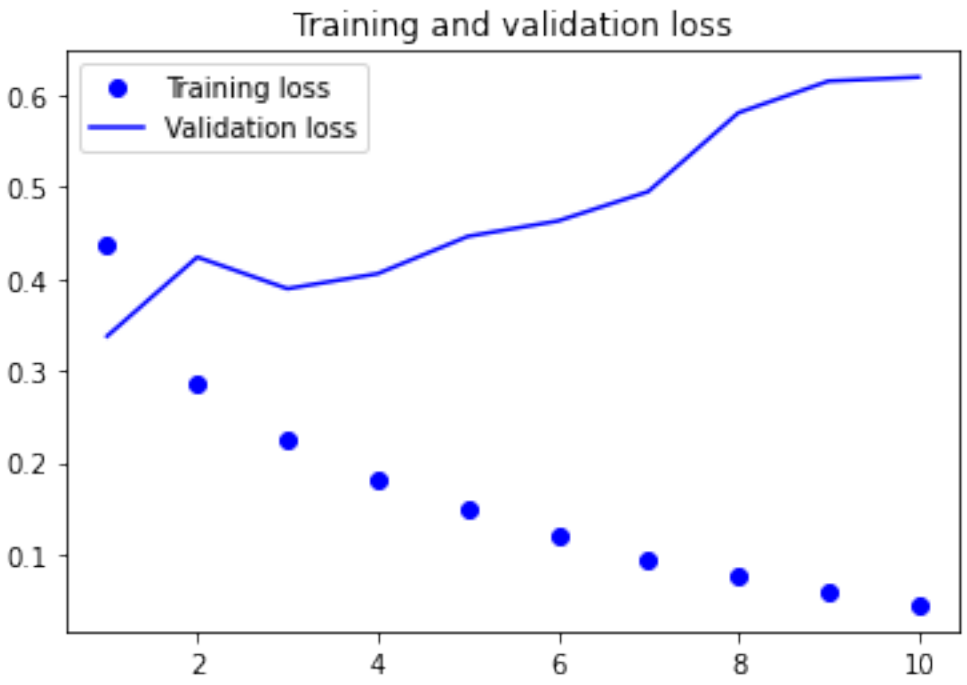
```
dtype int32
         [[{{node gradients/split_1_grad/concat/split_1/split_dim}}]]
2023-05-14 13:38:45.883752: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_2_grad/concat/split_2/split_dim' with
dtype int32
         [[{{node gradients/split_2_grad/concat/split_2/split_dim}}]]
2023-05-14 13:38:45.887249: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_grad/concat/split/split_dim' with dtype
int32
         [[{{node gradients/split_grad/concat/split/split_dim}}]]
2023-05-14 13:38:45.889830: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_1_grad/concat/split_1/split_dim' with
dtype int32
         [[{{node gradients/split_1_grad/concat/split_1/split_dim}}]]
2023-05-14 13:38:46.933764: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_2_grad/concat/split_2/split_dim' with
dtype int32
         [[{{node gradients/split_2_grad/concat/split_2/split_dim}}]]
2023-05-14 13:38:46.937256: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_grad/concat/split/split_dim' with dtype
int32
         [[{{node gradients/split_grad/concat/split/split_dim}}]]
2023-05-14 13:38:46.939986: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_1_grad/concat/split_1/split_dim' with
dtype int32
         [[{{node gradients/split_1_grad/concat/split_1/split_dim}}]]

782/782 [==============================] - ETA: 0s - loss: 0.4363 - acc: 0.7949

2023-05-14 13:39:45.758918: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_2_grad/concat/split_2/split_dim' with
dtype int32
         [[{{node gradients/split_2_grad/concat/split_2/split_dim}}]]
2023-05-14 13:39:45.761675: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
```

```
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_grad/concat/split/split_dim' with dtype
int32
         [[{{node gradients/split_grad/concat/split/split_dim}}]]
2023-05-14 13:39:45.764294: I tensorflow/core/common_runtime/executor.cc:1197]
[/device:CPU:0] (DEBUG INFO) Executor start aborting (this does not indicate an
error and you can ignore this message): INVALID_ARGUMENT: You must feed a value
for placeholder tensor 'gradients/split_1_grad/concat/split_1/split_dim' with
dtype int32
         [[{{node gradients/split_1_grad/concat/split_1/split_dim}}]]
782/782 [==============================] - 64s 79ms/step - loss: 0.4363 - acc:
0.7949 - val_loss: 0.3377 - val_acc: 0.8510
Epoch 2/10
782/782 [==============================] - 62s 79ms/step - loss: 0.2872 - acc:
0.8844 - val_loss: 0.4238 - val_acc: 0.8306
Epoch 3/10
782/782 [==============================] - 62s 80ms/step - loss: 0.2263 - acc:
0.9138 - val_loss: 0.3894 - val_acc: 0.8476
Epoch 4/10
782/782 [==============================] - 63s 81ms/step - loss: 0.1828 - acc:
0.9342 - val_loss: 0.4057 - val_acc: 0.8434
Epoch 5/10
782/782 [==============================] - 64s 82ms/step - loss: 0.1492 - acc:
0.9488 - val_loss: 0.4463 - val_acc: 0.8470
Epoch 6/10
782/782 [==============================] - 62s 80ms/step - loss: 0.1193 - acc:
0.9600 - val_loss: 0.4631 - val_acc: 0.8430
Epoch 7/10
782/782 [==============================] - 62s 79ms/step - loss: 0.0953 - acc:
0.9693 - val_loss: 0.4950 - val_acc: 0.8421
Epoch 8/10
782/782 [==============================] - 63s 80ms/step - loss: 0.0769 - acc:
0.9761 - val_loss: 0.5808 - val_acc: 0.8150
Epoch 9/10
782/782 [==============================] - 63s 81ms/step - loss: 0.0581 - acc:
0.9823 - val_loss: 0.6153 - val_acc: 0.8310
Epoch 10/10
782/782 [==============================] - 63s 80ms/step - loss: 0.0452 - acc:
0.9861 - val_loss: 0.6196 - val_acc: 0.8236
391/391 [==============================] - 5s 13ms/step - loss: 0.6208 - acc:
0.8224
Model Accuracy: 0.8223999738693237
```

[12]:
```python
import matplotlib.pyplot as plt

# Creating training and validation loss and accuracy curves
acc = history.history['acc']
```

```
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(acc) + 1)
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()
plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()
plt.show()
```





[ ]:

0.6 10.4 1D Convnet

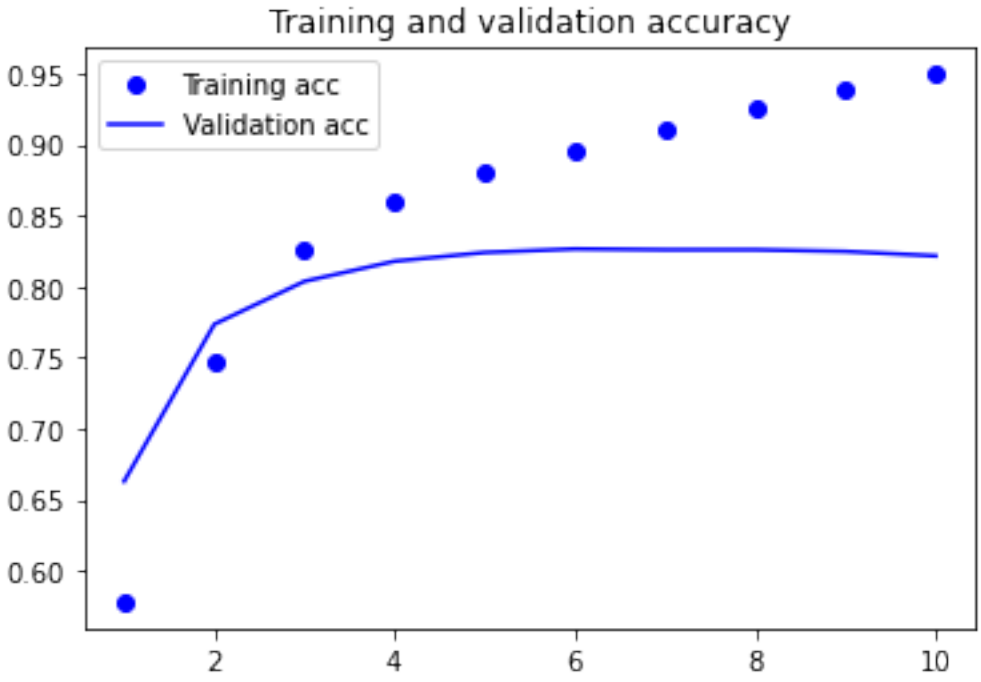
[13]:
```
from keras.optimizers import RMSprop
from keras import layers

# model architecture
model = Sequential()
model.add(Embedding(max_words, embedding_dim, input_length= max_length))
model.add(layers.Conv1D(32, 7, activation='relu')) # Adding 1D Convnet layer
model.add(layers.MaxPooling1D(5))
model.add(layers.Conv1D(32, 7, activation='relu')) # Adding 1D Convnet layer
model.add(layers.GlobalMaxPooling1D())
model.add(layers.Dense(1))

model.compile(optimizer=RMSprop(lr=1e-4),
              loss='binary_crossentropy',
              metrics=['acc'])

# training the model
history = model.fit(x_train_padded, y_train,
                    epochs=10,
                    batch_size=32,
```

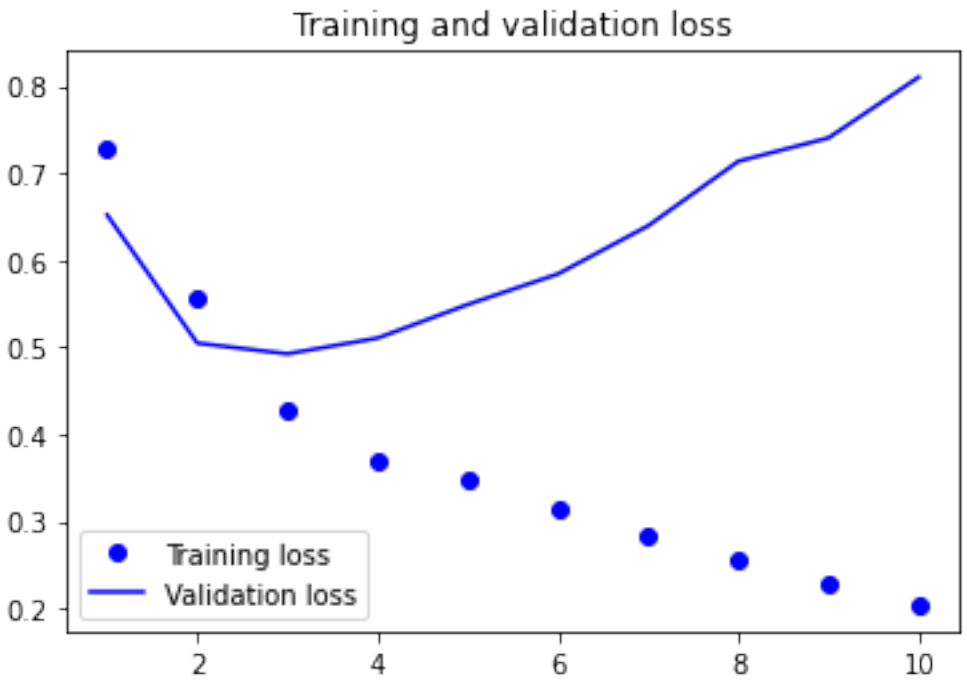
```
                    validation_data=(x_val_padded, y_val))

test_loss, test_acc = model.evaluate(x_test_padded, y_test)
print("Model Accuracy:", test_acc)
```

```
Epoch 1/10
782/782 [==============================] - 43s 54ms/step - loss: 0.7280 - acc:
0.5778 - val_loss: 0.6518 - val_acc: 0.6628
Epoch 2/10
782/782 [==============================] - 42s 54ms/step - loss: 0.5574 - acc:
0.7474 - val_loss: 0.5046 - val_acc: 0.7734
Epoch 3/10
782/782 [==============================] - 42s 54ms/step - loss: 0.4284 - acc:
0.8252 - val_loss: 0.4924 - val_acc: 0.8037
Epoch 4/10
782/782 [==============================] - 43s 55ms/step - loss: 0.3702 - acc:
0.8602 - val_loss: 0.5104 - val_acc: 0.8178
Epoch 5/10
782/782 [==============================] - 44s 56ms/step - loss: 0.3470 - acc:
0.8806 - val_loss: 0.5490 - val_acc: 0.8240
Epoch 6/10
782/782 [==============================] - 42s 53ms/step - loss: 0.3151 - acc:
0.8965 - val_loss: 0.5841 - val_acc: 0.8265
Epoch 7/10
782/782 [==============================] - 41s 53ms/step - loss: 0.2827 - acc:
0.9112 - val_loss: 0.6394 - val_acc: 0.8259
Epoch 8/10
782/782 [==============================] - 42s 54ms/step - loss: 0.2559 - acc:
0.9257 - val_loss: 0.7134 - val_acc: 0.8259
Epoch 9/10
782/782 [==============================] - 43s 55ms/step - loss: 0.2278 - acc:
0.9392 - val_loss: 0.7402 - val_acc: 0.8247
Epoch 10/10
782/782 [==============================] - 42s 54ms/step - loss: 0.2040 - acc:
0.9496 - val_loss: 0.8100 - val_acc: 0.8216
391/391 [==============================] - 2s 5ms/step - loss: 0.8159 - acc:
0.8208
Model Accuracy: 0.8208000063896179
```

[14]:
```
import matplotlib.pyplot as plt

# Creating training and validation loss and accuracy curves
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(acc) + 1)
```

```
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()
plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()
plt.show()
```




Training and validation loss
0.8
0.7
0.6
0.5
0.4
0.3
0.2
2
4
6
8
10
Training loss
Validation loss

[ ]: