In [10]:
```
import pandas as pd
import numpy as np
import text_normalizer as tn
import model_evaluation_utils as meu



np.set_printoptions(precision=2, linewidth=80)
```

In [12]:
```
dataset = pd.read_csv(r'movie_reviews.csv')
```

In [14]:
```
reviews = np.array(dataset['review'])
sentiments = np.array(dataset['sentiment'])

# build train and test datasets
train_reviews = reviews[:35000]
train_sentiments = sentiments[:35000]
test_reviews = reviews[35000:]
test_sentiments = sentiments[35000:]

# normalize datasets
norm_train_reviews = tn.normalize_corpus(train_reviews)
norm_test_reviews = tn.normalize_corpus(test_reviews)
```

In [15]:
```
tokenized_train = [tn.tokenizer.tokenize(text) for text in norm_train_reviews]
tokenized_test = [tn.tokenizer.tokenize(text) for text in norm_test_reviews]
```

In [16]:
```
from collections import Counter

# build word to index vocabulary
token_counter = Counter([token for review in tokenized_train for token in review
vocab_map = {item[0]: index+1 for index, item in enumerate(dict(token_counter).i
max_index = np.max(list(vocab_map.values()))
vocab_map['PAD_INDEX'] = 0
vocab_map['NOT_FOUND_INDEX'] = max_index+1
vocab_size = len(vocab_map)
# view vocabulary size and part of the vocabulary map
print('Vocabulary Size:', vocab_size)
print('Sample slice of vocabulary map:', dict(list(vocab_map.items())[10:20]))
```

```
Vocabulary Size: 77811
Sample slice of vocabulary map: {'bore': 11, 'terribly': 12, 'predictable': 13,
'interesting': 14, 'start': 15, 'middle': 16, 'film': 17, 'little': 18, 'socia
l': 19, 'drama': 20}
```

In [17]:
```
from keras.preprocessing import sequence
from sklearn.preprocessing import LabelEncoder

# get max length of train corpus and initialize label encoder
le = LabelEncoder()
num_classes=2 # positive -> 1, negative -> 0
max_len = np.max([len(review) for review in tokenized_train])

## Train reviews data corpus
# Convert tokenized text reviews to numeric vectors
train_X = [[vocab_map[token] for token in tokenized_review] for tokenized_review
train_X = sequence.pad_sequences(train_X, maxlen=max_len) # pad
## Train prediction class labels
# Convert text sentiment labels (negative\positive) to binary encodings (0/1)
train_y = le.fit_transform(train_sentiments)

## Test reviews data corpus
# Convert tokenized text reviews to numeric vectors
test_X = [[vocab_map[token] if vocab_map.get(token) else vocab_map['NOT_FOUND_IND
           for token in tokenized_review]
              for tokenized_review in tokenized_test]
test_X = sequence.pad_sequences(test_X, maxlen=max_len)
## Test prediction class labels
# Convert text sentiment labels (negative\positive) to binary encodings (0/1)
test_y = le.transform(test_sentiments)

# view vector shapes
print('Max length of train review vectors:', max_len)
print('Train review vectors shape:', train_X.shape, ' Test review vectors shape:
```

```
Using TensorFlow backend.

Max length of train review vectors: 1115
Train review vectors shape: (35000, 1115)  Test review vectors shape: (15000, 1
115)
```

In [18]:
```
from keras.models import Sequential
from keras.layers import Dense, Embedding, Dropout, SpatialDropout1D
from keras.layers import LSTM

EMBEDDING_DIM = 128 # dimension for dense embeddings for each token
LSTM_DIM = 64 # total LSTM units

model = Sequential()
model.add(Embedding(input_dim=vocab_size, output_dim=EMBEDDING_DIM, input_length=
model.add(SpatialDropout1D(0.2))
model.add(LSTM(LSTM_DIM, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(1, activation="sigmoid"))

model.compile(loss="binary_crossentropy", optimizer="adam",
              metrics=["accuracy"])
```

```
WARNING:tensorflow:From c:\users\mglewis\appdata\local\programs\python\python37
\lib\site-packages\tensorflow\python\framework\op_def_library.py:263: colocate_
with (from tensorflow.python.framework.ops) is deprecated and will be removed i
n a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From c:\users\mglewis\appdata\local\programs\python\python37
\lib\site-packages\keras\backend\tensorflow_backend.py:3445: calling dropout (f
rom tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be remo
ved in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - kee
p_prob`.
```

In [19]:
```
print(model.summary())
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 1115, 128)         9959808
_________________________________________________________________
spatial_dropout1d_1 (Spatial (None, 1115, 128)         0
_________________________________________________________________
lstm_1 (LSTM)                (None, 64)                49408
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 10,009,281
Trainable params: 10,009,281
Non-trainable params: 0
_________________________________________________________________
None
```

In [ ]:
```
from IPython.display import SVG
from keras.utils.vis_utils import model_to_dot

SVG(model_to_dot(model, show_shapes=True, show_layer_names=False,
                 rankdir='LR').create(prog='dot', format='svg'))
```

In [23]:
```
batch_size = 100
model.fit(train_X, train_y, epochs=5, batch_size=batch_size,
          shuffle=True, validation_split=0.1, verbose=1)
```

```
1s - loss: 0.0736 - acc: 0.97 - ETA: 50s - loss: 0.0736 - acc: 0.97 - ETA: 48
s - loss: 0.0736 - acc: 0.97 - ETA: 46s - loss: 0.0736 - acc: 0.97 - ETA: 44s
- loss: 0.0736 - acc: 0.97 - ETA: 43s - loss: 0.0734 - acc: 0.97 - ETA: 41s -
loss: 0.0735 - acc: 0.97 - ETA: 39s - loss: 0.0737 - acc: 0.97 - ETA: 38s - l
oss: 0.0738 - acc: 0.97 - ETA: 36s - loss: 0.0740 - acc: 0.97 - ETA: 34s - lo
ss: 0.0738 - acc: 0.97 - ETA: 32s - loss: 0.0739 - acc: 0.97 - ETA: 31s - los
s: 0.0738 - acc: 0.97 - ETA: 29s - loss: 0.0741 - acc: 0.97 - ETA: 27s - los
s: 0.0741 - acc: 0.97 - ETA: 25s - loss: 0.0743 - acc: 0.97 - ETA: 24s - los
s: 0.0742 - acc: 0.97 - ETA: 22s - loss: 0.0741 - acc: 0.97 - ETA: 20s - los
s: 0.0745 - acc: 0.97 - ETA: 19s - loss: 0.0748 - acc: 0.97 - ETA: 17s - los
s: 0.0750 - acc: 0.97 - ETA: 15s - loss: 0.0748 - acc: 0.97 - ETA: 13s - los
s: 0.0747 - acc: 0.97 - ETA: 12s - loss: 0.0746 - acc: 0.97 - ETA: 10s - los
s: 0.0745 - acc: 0.97 - ETA: 8s - loss: 0.0745 - acc: 0.9752 - ETA: 6s - los
s: 0.0744 - acc: 0.975 - ETA: 5s - loss: 0.0744 - acc: 0.975 - ETA: 3s - los

s: 0.0744 - acc: 0.975 - ETA: 1s - loss: 0.0746 - acc: 0.975 - 554s 18ms/step
- loss: 0.0748 - acc: 0.9750 - val_loss: 0.3772 - val_acc: 0.8786
```

Out[23]:
```
<keras.callbacks.History at 0x230b678d8d0>
```

In [24]:
```
pred_test = model.predict_classes(test_X)
predictions = le.inverse_transform(pred_test.flatten())
```

In [25]:
```
meu.display_model_performance_metrics(true_labels=test_sentiments, predicted_labe
                                      classes=['positive', 'negative'])
```

```
Model Performance metrics:
------------------------------
Accuracy: 0.8789
Precision: 0.8789
Recall: 0.8789
F1 Score: 0.8789

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    positive       0.88      0.88      0.88      7587
    negative       0.88      0.88      0.88      7413

   micro avg       0.88      0.88      0.88     15000
   macro avg       0.88      0.88      0.88     15000
weighted avg       0.88      0.88      0.88     15000


Prediction Confusion Matrix:
------------------------------

C:\code\ML York\ML1010_InClass-master\ML1010_InClass-master\Week 5\model_evalua
tion_utils.py:61: FutureWarning: the 'labels' keyword is deprecated, use 'code
s' instead
  labels=level_labels),
C:\code\ML York\ML1010_InClass-master\ML1010_InClass-master\Week 5\model_evalua
tion_utils.py:63: FutureWarning: the 'labels' keyword is deprecated, use 'code
s' instead
  labels=level_labels))

                Predicted:
                  positive negative
Actual: positive      6680      907
        negative       910     6503
```

In [ ]:
```

```