Chen_Pohao_Final_Project

Name: Po Hao Chen
Github Username: pohaoc29
USC ID: 4213309111

1. Text Classification

(a)

import packages

In [1]:
```
import os,sys
import numpy as np
import math
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import re
from tensorflow import keras
import warnings
warnings.filterwarnings("ignore")
```

(b) Data Exploration and Pre-processing

(i) (ii) Data pre-processing and labeling

In [2]:
```
Ppath = '../Data/pos'
Pfiles = sorted(os.listdir(Ppath))

Npath = '../Data/neg'
Nfiles = sorted(os.listdir(Npath))

Ptxts = list()
Ntxts = list()

for file in Pfiles:
    Pfname = Ppath+'/'+file
    #print(Pfname)
    with open(Pfname, 'r') as f:
        text = f.read()
        res = re.sub(r'[\W\_\d]', ' ', text).split()
        #result = re.sub(r'[\s+]', ' ', res).strip()
        Ptxts.append(res)

pos_df = pd.DataFrame({'text': Ptxts, 'label': list([1 for i in range(

for file in Nfiles:
    Nfname = Npath+'/'+file
    #print(Nfname)
    with open(Nfname, 'r') as f:
        text = f.read()
        res = re.sub(r'[\W\_\d]', ' ', text).split()
        #result = re.sub(r'[\s+]', ' ', res)
        Ntxts.append(res)


neg_df = pd.DataFrame({'text': Ntxts, 'label': list([-1 for i in range
```

In [3]:
```
pos_df
```

Out[3]:

	text	label
0	[films, adapted, from, comic, books, have, had...	1
1	[every, now, and, then, a, movie, comes, along...	1
2	[you, ve, got, mail, works, alot, better, than...	1
3	[jaws, is, a, rare, film, that, grabs, your, a...	1
4	[moviemaking, is, a, lot, like, being, the, ge...	1
...	...	...
995	[wow, what, a, movie, it, s, everything, a, mo...	1
996	[richard, gere, can, be, a, commanding, actor,...	1
997	[glory, starring, matthew, broderick, denzel, ...	1
998	[steven, spielberg, s, second, epic, film, on,...	1
999	[truman, true, man, burbank, is, the, perfect,...	1

1000 rows × 2 columns

In [4]:
```
neg_df
```

Out[4]:

	text	label
0	[plot, two, teen, couples, go, to, a, church, ...	-1
1	[the, happy, bastard, s, quick, movie, review,...	-1
2	[it, is, movies, like, these, that, make, a, j...	-1
3	[quest, for, camelot, is, warner, bros, first,...	-1
4	[synopsis, a, mentally, unstable, man, undergo...	-1
...	...	...
995	[if, anything, stigmata, should, be, taken, as...	-1
996	[john, boorman, s, zardoz, is, a, goofy, cinem...	-1
997	[the, kids, in, the, hall, are, an, acquired, ...	-1
998	[there, was, a, time, when, john, carpenter, w...	-1
999	[two, party, guys, bob, their, heads, to, hadd...	-1

1000 rows × 2 columns

In [5]:
```
All_df = pd.concat([pos_df,neg_df])
All_df
```

Out[5]:

	text	label
0	[films, adapted, from, comic, books, have, had...	1
1	[every, now, and, then, a, movie, comes, along...	1
2	[you, ve, got, mail, works, alot, better, than...	1
3	[jaws, is, a, rare, film, that, grabs, your, a...	1
4	[moviemaking, is, a, lot, like, being, the, ge...	1
...	...	...
995	[if, anything, stigmata, should, be, taken, as...	-1
996	[john, boorman, s, zardoz, is, a, goofy, cinem...	-1
997	[the, kids, in, the, hall, are, an, acquired, ...	-1
998	[there, was, a, time, when, john, carpenter, w...	-1
999	[two, party, guys, bob, their, heads, to, hadd...	-1

2000 rows × 2 columns

(iii) Split the train and test data

In [6]:
```
train_df = pd.concat([pos_df[:700],neg_df[:700]])
train_df
```

Out[6]:

	text	label
0	[films, adapted, from, comic, books, have, had...	1
1	[every, now, and, then, a, movie, comes, along...	1
2	[you, ve, got, mail, works, alot, better, than...	1
3	[jaws, is, a, rare, film, that, grabs, your, a...	1
4	[moviemaking, is, a, lot, like, being, the, ge...	1
...	...	...
695	[house, on, haunted, hill, starring, taye, dig...	-1
696	[fit, for, a, ghoul, s, night, out, fat, girl,...	-1
697	[marie, couldn, t, talk, paulie, the, parrot, ...	-1
698	[well, here, s, a, distasteful, thoroughly, am...	-1
699	[okay, i, just, don, t, know, why, but, i, see...	-1

1400 rows × 2 columns

In [7]:
```
test_df = pd.concat([pos_df[700:],neg_df[700:]])
test_df
```

Out[7]:

	text	label
700	[let, me, start, off, by, saying, that, leadin...	1
701	[seen, september, at, p, m, at, the, sony, nic...	1
702	[the, characters, in, palmetto, collectively, ...	1
703	[you, ve, got, mail, is, the, very, definition...	1
704	[with, the, sudden, liberal, emergence, of, pe...	1
...	...	...
995	[if, anything, stigmata, should, be, taken, as...	-1
996	[john, boorman, s, zardoz, is, a, goofy, cinem...	-1
997	[the, kids, in, the, hall, are, an, acquired, ...	-1
998	[there, was, a, time, when, john, carpenter, w...	-1
999	[two, party, guys, bob, their, heads, to, hadd...	-1

(iv) Count the number of unique words

In [8]:
```
count_dict = dict()

for text in All_df['text']:
    for word in text:
        if word not in count_dict:
            count_dict[word] = 1
        else:
            count_dict[word] += 1
#print(count_dict)
```

In [9]:
```
print("The number of unique words is:", len(count_dict))
```

```
The number of unique words is: 38911
```

The number of each unique word:

In [10]:
```
pd.set_option('display.max_rows', None)

count_word_df = pd.DataFrame(count_dict.items(), columns=['unique word
count_word_df
```

Out[10]:

	unique word	count
0	films	1536
1	adapted	46
2	from	4999
3	comic	389
4	books	78
5	have	4902
6	had	1546
7	plenty	134
8	of	34126
9	success	216
10	whether	217

(v) Calculate average and standard deviation of review length

In [11]:
```
review_length = list()
for text in All_df['text']:
    review_length.append(len(text))

print("The average of review length is:", np.mean(review_length))
print("The standard deviation of revew length is:", np.std(review_leng
```

```
The average of review length is: 665.636
The standard deviation of revew length is: 293.66091245516486
```

(vi) Plot the histogram of review lengths

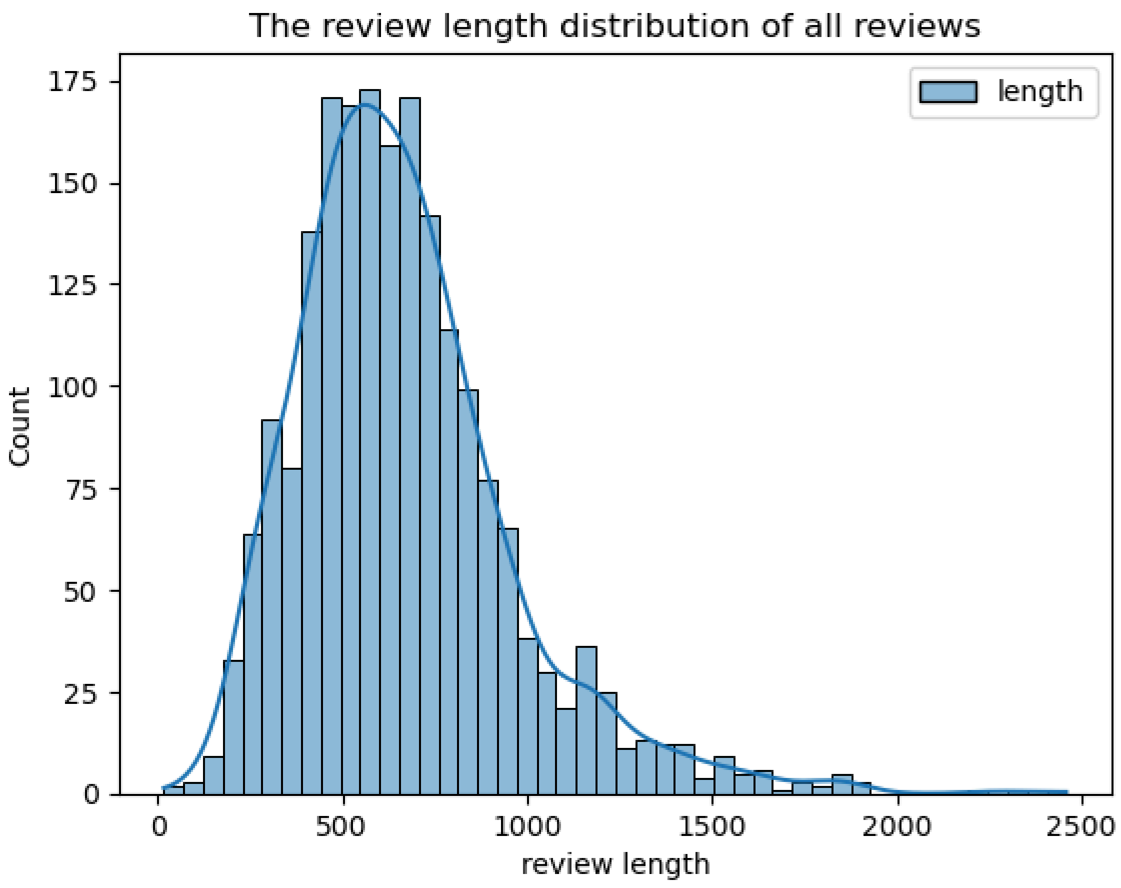
In [12]:
```
All_df['length'] = review_length
All_df
```

Out[12]:

	text	label	length
0	[films, adapted, from, comic, books, have, had...	1	707
1	[every, now, and, then, a, movie, comes, along...	1	672
2	[you, ve, got, mail, works, alot, better, than...	1	431
3	[jaws, is, a, rare, film, that, grabs, your, a...	1	1018
4	[moviemaking, is, a, lot, like, being, the, ge...	1	673
5	[on, june, a, self, taught, idealistic, yet, p...	1	871
6	[apparently, director, tony, kaye, had, a, maj...	1	806
7	[one, of, my, colleagues, was, surprised, when...	1	636
8	[after, bloody, clashes, and, independence, wo...	1	262
9	[the, american, action, film, has, been, slowl...	1	433
10	[after, watching, rat, race, last, week, i, no...	1	849

In [13]:
```
sns.histplot(All_df.drop(['label'], axis=1), kde=True)
plt.xlabel('review length')
plt.title('The review length distribution of all reviews')
```

Out[13]:
```
Text(0.5, 1.0, 'The review length distribution of all reviews')
```



(vii) Tokenization

In [14]:
```
# Reference source: Tokenize.webarchive
from keras.preprocessing.text import Tokenizer

#word_counts = list()
word_index = list()
#one_hot_encoded = list()
text_rank_seq = list()

t = Tokenizer()

t.fit_on_texts(train_df['text'])#(All_df['text'])
print(t.word_index)

for text in All_df['text']:
    #print(t.word_counts)
    #word_counts.append(t.word_counts)

    #word_index.append(t.word_index)
    seq = t.texts_to_sequences(text)
    print(seq)
    text_rank_seq.append(seq)

    # encoded_docs = t.texts_to_matrix(text, mode='count')
    # print(encoded_docs)
    # one_hot_encoded.append(encoded_docs)
```

```
{'the': 1, 'a': 2, 'and': 3, 'of': 4, 'to': 5, 'is': 6, 'in': 7,
's': 8, 'it': 9, 'that': 10, 'as': 11, 'with': 12, 'for': 13, 'his':
14, 'film': 15, 'this': 16, 'i': 17, 'he': 18, 'but': 19, 'on': 20,
'are': 21, 't': 22, 'by': 23, 'be': 24, 'one': 25, 'an': 26, 'who':
27, 'movie': 28, 'not': 29, 'you': 30, 'at': 31, 'was': 32, 'from':
33, 'have': 34, 'they': 35, 'has': 36, 'her': 37, 'all': 38, 'ther
e': 39, 'like': 40, 'out': 41, 'so': 42, 'about': 43, 'more': 44, 'u
p': 45, 'what': 46, 'when': 47, 'which': 48, 'or': 49, 'she': 50, 't
heir': 51, 'some': 52, 'just': 53, 'can': 54, 'if': 55, 'we': 56, 'i
nto': 57, 'him': 58, 'even': 59, 'no': 60, 'only': 61, 'than': 62,
'time': 63, 'good': 64, 'most': 65, 'its': 66, 'will': 67, 'story':
68, 'would': 69, 'been': 70, 'much': 71, 'character': 72, 'get': 73,
'also': 74, 'other': 75, 'do': 76, 'well': 77, 'first': 78, 'two': 7
9, 'very': 80, 'characters': 81, 'them': 82, 'see': 83, 'after': 84,
'way': 85, 'because': 86, 'make': 87, 'life': 88, 'too': 89, 'does':
90, 'really': 91, 'off': 92, 'plot': 93, 'little': 94, 'had': 95, 'a
ny': 96, 'films': 97, 'while': 98, 'how': 99, 'where': 100, 'then':
101, 'me': 102, 'people': 103, 'over': 104, 'man': 105, 'could': 10
6, 'scene': 107, 'bad': 108, 'best': 109, 'my': 110, 'never': 111,
```

(viii) Select the reviews have a length below L

If L=70%

In [15]:
```
All_df['length'].quantile(0.7)
```

Out[15]:
```
759.3
```

In [16]:
```
L70_df = All_df[All_df['length'] < All_df['length'].quantile(0.7)]
L70_df
```

Out[16]:

	text	label	length
0	[films, adapted, from, comic, books, have, had...	1	707
1	[every, now, and, then, a, movie, comes, along...	1	672
2	[you, ve, got, mail, works, alot, better, than...	1	431
4	[moviemaking, is, a, lot, like, being, the, ge...	1	673
7	[one, of, my, colleagues, was, surprised, when...	1	636
8	[after, bloody, clashes, and, independence, wo...	1	262
9	[the, american, action, film, has, been, slowl...	1	433
11	[i, ve, noticed, something, lately, that, i, v...	1	700
12	[synopsis, bobby, garfield, yelchin, lives, in...	1	312
13	[synopsis, in, this, movie, steven, spielberg,...	1	293
15	[plot, a, young, man, who, loves, heavy, metal...	1	587

In [17]:
```
# Decided threshold is whether 759 or 760
threshold = 760
L70_df = All_df[All_df['length'] < threshold]
L70_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1400 entries, 0 to 999
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    1400 non-null   object
 1   label   1400 non-null   int64 
 2   length  1400 non-null   int64 
dtypes: int64(2), object(1)
memory usage: 43.8+ KB
```

In [18]:
```
rest_df = All_df[All_df['length'] > All_df['length'].quantile(0.7)]
rest_df
```

Out[18]:

	text	label	length
3	[jaws, is, a, rare, film, that, grabs, your, a...	1	1018
5	[on, june, a, self, taught, idealistic, yet, p...	1	871
6	[apparently, director, tony, kaye, had, a, maj...	1	806
10	[after, watching, rat, race, last, week, i, no...	1	849
14	[the, police, negotiator, is, the, person, wit...	1	1140
17	[the, ultimate, match, up, between, good, and,...	1	835
23	[when, bulworth, ended, i, allowed, myself, a,...	1	782
24	[call, for, the, cliche, police, if, you, must...	1	934
25	[hilarious, ultra, low, budget, comedy, from, ...	1	772
27	[the, most, common, and, in, many, cases, the,...	1	1147
28	[the, blair, witch, project, was, perhaps, one...	1	997

I've tried both 70% and 90% as a threshold, and their performance is nearly the same in my model; using 70% as a threshold seems to have a little better performance, so I choose 70% to do the following steps.

(ix) Truncating and zero-padding

In [19]:
```
for i in range(len(text_rank_seq)):
    for j in range(len(text_rank_seq[i])):
        if len(text_rank_seq[i][j]) == 0:
            text_rank_seq[i][j] = [0]
```

In [20]:
```
import tensorflow as tf
from tensorflow.keras.preprocessing.sequence import pad_sequences

# Truncate reviews longer than L
padded_seq = tf.keras.utils.pad_sequences(text_rank_seq, dtype='int32'
```

(c) Word Enbeddings

(i) - (ii) embedding and fletten

In [21]:
```
from keras import Sequential,layers
from keras.layers import Embedding, Flatten
# L
max_length = threshold

input_array = padded_seq
input_array[input_array > 5000] = 0

model = Sequential()
model.add(Embedding(5001, 32, input_length=max_length))
model.add(Flatten())

output_array = model.predict(input_array)
print(model.summary())
#print(output_array)
```

```
63/63 [==============================] - 0s 833us/step
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 embedding (Embedding)       (None, 760, 32)           160032    
                                                                 
 flatten (Flatten)           (None, 24320)             0         
                                                                 
=================================================================
Total params: 160,032
Trainable params: 160,032
Non-trainable params: 0
_________________________________________________________________
None

2024-05-05 18:24:36.600120: W tensorflow/tsl/platform/profile_utils/
cpu_utils.cc:128] Failed to get CPU frequency: 0 Hz
```

In [22]:
```
print(output_array)
```

```
[[ 0.03655313  0.02482773 -0.03938542 ... -0.03580993  0.0145128
  -0.04934711]
 [ 0.00483906  0.04944864  0.04300383 ... -0.03580993  0.0145128
  -0.04934711]
 [ 0.03773287 -0.03392153 -0.02311529 ... -0.03580993  0.0145128
  -0.04934711]
 ...
 [-0.01970756  0.00251323 -0.02096295 ... -0.03580993  0.0145128
  -0.04934711]
 [ 0.03555525  0.01667458 -0.0403561  ... -0.03580993  0.0145128
  -0.04934711]
 [ 0.04152014  0.04267934  0.03308436 ... -0.03580993  0.0145128
  -0.04934711]]
```

In [23]:
```
len(output_array), len(output_array[0])
```

Out[23]:
```
(2000, 24320)
```

(d) Multi_Layer Perceptron

(i) Train MLP model

Split the train, test dataset

In [24]:
```
#padded_seq 2000*760

train_metrix = np.vstack((output_array[0:700], output_array[1000:1700]
print(len(train_metrix))
train_label = np.array([1 for i in range(700)] + [0 for i in range(70
print(len(train_label))

test_metrix = np.vstack((output_array[700:1000], output_array[1700:]))
print(len(test_metrix))
test_label = np.array([1 for i in range(300)] + [0 for i in range(300)
print(len(test_label))
```

```
1400
1400
600
600
```

Build the model

In [25]:
```
#source: keras.io doc
epochs = 2
batch_size = 10

MLPmodel = keras.Sequential()


mlp1 = keras.Sequential(
            [
                layers.Dense(units=50, activation='relu'),
                layers.Dropout(rate=0.2),
            ]
)

mlp2 = keras.Sequential(
            [
                layers.Dense(units=50, activation='relu'),
                layers.Dropout(rate=0.5),
            ]
)

mlp3 = keras.Sequential(
            [
                layers.Dense(units=50, activation='relu'),
                layers.Dropout(rate=0.5),
            ]
)

MLPmodel.add(mlp1)
MLPmodel.add(mlp2)
MLPmodel.add(mlp3)
MLPmodel.add(layers.Dense(units=1, activation='sigmoid'))

MLPmodel.compile(
    optimizer = 'adam',#keras.optimizers.legacy.Adam(),
    loss = 'binary_crossentropy',#keras.losses.BinaryCrossentropy(),
    metrics=[
            'accuracy'#keras.metrics.Accuracy(name="acc")
            #keras.metrics.SparseTopKCategoricalAccuracy(5, name="top5
        ],
)


MLPmodel.fit(train_metrix, train_label, epochs=epochs, batch_size=batc
```

```
Epoch 1/2
140/140 [==============================] - 1s 2ms/step - loss: 0.698
3 - accuracy: 0.5157
Epoch 2/2
140/140 [==============================] - 0s 2ms/step - loss: 0.689
0 - accuracy: 0.5393
```

Out[25]:
```
<keras.callbacks.History at 0x2bf47e380>
```

(ii) Train/Test accuracy

In [26]:
```
train_loss, train_accuracy =  MLPmodel.evaluate(train_metrix, train_la
print(f"Train accuracy: {round(train_accuracy * 100, 2)}%")

test_loss, test_accuracy = MLPmodel.evaluate(test_metrix, test_label)
print(f"Test accuracy: {round(test_accuracy * 100, 2)}%")
```

```
44/44 [==============================] - 0s 1ms/step - loss: 0.6657
- accuracy: 0.6250
Train accuracy: 62.5%
19/19 [==============================] - 0s 1ms/step - loss: 0.6917
- accuracy: 0.5067
Test accuracy: 50.67%
```

(e) 1-D CNN

(i)

In [27]:
```
#padded_seq 2000*760

all_seq = padded_seq
all_seq[all_seq > 5000] = 0

train_X = np.vstack((all_seq[0:700], all_seq[1000:1700]))
print(len(train_X))
train_label = np.array([1 for i in range(700)] + [0 for i in range(70
print(len(train_label))

test_X = np.vstack((all_seq[700:1000], all_seq[1700:]))
print(len(test_X))
test_label = np.array([1 for i in range(300)] + [0 for i in range(300)
print(len(test_label))
```

```
1400
1400
600
600
```

In [28]:
```
epochs = 2
batch_size = 10

# L
max_length = threshold


CNNmodel = keras.Sequential()
CNNmodel.add(Embedding(5001, 32, input_length=threshold))


CNNmodel.add(keras.layers.Conv1D(32, 3, activation='relu'))
CNNmodel.add(keras.layers.MaxPooling1D(pool_size=2, strides=2))

CNNmodel.add(Flatten())

# MLP 3 layers
mlp1 = keras.Sequential(
            [
                layers.Dense(units=50, activation='relu'),
                layers.Dropout(rate=0.2),
            ]
)

mlp2 = keras.Sequential(
            [
                layers.Dense(units=50, activation='relu'),
                layers.Dropout(rate=0.5),
            ]
)

mlp3 = keras.Sequential(
            [
                layers.Dense(units=50, activation='relu'),
                layers.Dropout(rate=0.5),
            ]
)

CNNmodel.add(mlp1)
CNNmodel.add(mlp2)
CNNmodel.add(mlp3)
CNNmodel.add(layers.Dense(units=1, activation='sigmoid'))

CNNmodel.compile(
    optimizer = 'adam',#keras.optimizers.legacy.Adam(),
    loss = 'binary_crossentropy',#keras.losses.BinaryCrossentropy(),
    metrics=[
            'accuracy'#keras.metrics.Accuracy(name="acc"),
            #keras.metrics.SparseTopKCategoricalAccuracy(5, name="top5
        ],
)

print(CNNmodel.summary())

#train_label_exp = tf.expand_dims(train_label, 1)
CNNmodel.fit(train_X, train_label, epochs=epochs, batch_size=batch_siz
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 760, 32)           160032

 conv1d (Conv1D)             (None, 758, 32)           3104

 max_pooling1d (MaxPooling1D  (None, 379, 32)          0
 )

 flatten_1 (Flatten)         (None, 12128)             0

 sequential_6 (Sequential)   (None, 50)                606450

 sequential_7 (Sequential)   (None, 50)                2550

 sequential_8 (Sequential)   (None, 50)                2550

 dense_7 (Dense)             (None, 1)                 51

=================================================================
Total params: 774,737
Trainable params: 774,737
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/2
140/140 [==============================] - 1s 5ms/step - loss: 0.694
8 - accuracy: 0.5071
Epoch 2/2
140/140 [==============================] - 1s 4ms/step - loss: 0.690
9 - accuracy: 0.5286
```

Out[28]:
```
<keras.callbacks.History at 0x2d6cf9db0>
```

(ii) Trian/test accuracy

In [29]:
```
train_loss, train_accuracy =  CNNmodel.evaluate(train_X, train_label)
print(f"Train accuracy: {round(train_accuracy * 100, 2)}%")

test_loss, test_accuracy = CNNmodel.evaluate(test_X, test_label)
print(f"Test accuracy: {round(test_accuracy * 100, 2)}%")
```

```
44/44 [==============================] - 0s 2ms/step - loss: 0.6714
- accuracy: 0.6471
Train accuracy: 64.71%
19/19 [==============================] - 0s 2ms/step - loss: 0.6878
- accuracy: 0.5500
Test accuracy: 55.0%
```

(f) LSTM RNN

(i)

In [30]:
```
#padded_seq 2000*760

all_seq = padded_seq
all_seq[all_seq > 5000] = 0

train_X = np.vstack((all_seq[0:700], all_seq[1000:1700]))
print(len(train_X))
train_label = np.array([1 for i in range(700)] + [0 for i in range(70
print(len(train_label))

test_X = np.vstack((all_seq[700:1000], all_seq[1700:]))
print(len(test_X))
test_label = np.array([1 for i in range(300)] + [0 for i in range(300)
print(len(test_label))
```

```
1400
1400
600
600
```

In [31]:
```
epochs = 10
batch_size = 10


LSTM_model = keras.Sequential()

LSTM_model.add(layers.Embedding(5001, 32, input_length=threshold))
LSTM_model.add(layers.LSTM(32,  dropout=0.2))
LSTM_model.add(layers.Dense(256, activation='relu'))
LSTM_model.add(layers.Dropout(rate=0.2))
LSTM_model.add(layers.Dense(units=1, activation='sigmoid'))

LSTM_model.compile(
    optimizer = 'adam',#keras.optimizers.legacy.Adam(),
    loss = 'binary_crossentropy',#keras.losses.BinaryCrossentropy(),
    metrics=[
            'accuracy'#keras.metrics.Accuracy(name="acc"),
            #keras.metrics.SparseTopKCategoricalAccuracy(5, name="top5
        ],
)

print(LSTM_model.summary())

#train_label_exp = tf.expand_dims(train_label, 1)
LSTM_model.fit(train_X, train_label, epochs=epochs, batch_size=batch_s
```

```
Model: "sequential_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_2 (Embedding)     (None, 760, 32)           160032

 lstm (LSTM)                 (None, 32)                8320

 dense_8 (Dense)             (None, 256)               8448

 dropout_6 (Dropout)         (None, 256)               0

 dense_9 (Dense)             (None, 1)                 257

=================================================================
Total params: 177,057
Trainable params: 177,057
Non-trainable params: 0
_________________________________________________________________
None
Epoch 1/10
```

```
2024-05-05 18:24:39.798573: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_2_grad/concat/split_2/split_dim' with dtype int32
         [[{{node gradients/split_2_grad/concat/split_2/split_di
m}}]]
2024-05-05 18:24:39.799275: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_grad/concat/split/split_dim' with dtype int32
         [[{{node gradients/split_grad/concat/split/split_dim}}]]
2024-05-05 18:24:39.799804: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_1_grad/concat/split_1/split_dim' with dtype int32
         [[{{node gradients/split_1_grad/concat/split_1/split_di
m}}]]
2024-05-05 18:24:39.919953: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_2_grad/concat/split_2/split_dim' with dtype int32
         [[{{node gradients/split_2_grad/concat/split_2/split_di
m}}]]
2024-05-05 18:24:39.920431: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_grad/concat/split/split_dim' with dtype int32
         [[{{node gradients/split_grad/concat/split/split_dim}}]]
2024-05-05 18:24:39.920842: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_1_grad/concat/split_1/split_dim' with dtype int32
         [[{{node gradients/split_1_grad/concat/split_1/split_di
m}}]]
2024-05-05 18:24:40.106246: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_2_grad/concat/split_2/split_dim' with dtype int32
         [[{{node gradients/split_2_grad/concat/split_2/split_di
m}}]]
2024-05-05 18:24:40.107086: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_grad/concat/split/split_dim' with dtype int32
         [[{{node gradients/split_grad/concat/split/split_dim}}]]
2024-05-05 18:24:40.107761: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_1_grad/concat/split_1/split_dim' with dtype int32
         [[{{node gradients/split_1_grad/concat/split_1/split_di
m}}]]
```

```
140/140 [==============================] - 12s 80ms/step - loss: 0.6
932 - accuracy: 0.5050
Epoch 2/10
140/140 [==============================] - 11s 79ms/step - loss: 0.6
878 - accuracy: 0.5586
Epoch 3/10
140/140 [==============================] - 11s 79ms/step - loss: 0.6
194 - accuracy: 0.6136
Epoch 4/10
140/140 [==============================] - 11s 81ms/step - loss: 0.5
575 - accuracy: 0.6686
Epoch 5/10
140/140 [==============================] - 12s 85ms/step - loss: 0.5
010 - accuracy: 0.6814
Epoch 6/10
140/140 [==============================] - 12s 84ms/step - loss: 0.4
851 - accuracy: 0.6679
Epoch 7/10
140/140 [==============================] - 11s 80ms/step - loss: 0.4
800 - accuracy: 0.6807
Epoch 8/10
140/140 [==============================] - 11s 78ms/step - loss: 0.4
792 - accuracy: 0.6850
Epoch 9/10
140/140 [==============================] - 11s 81ms/step - loss: 0.4
837 - accuracy: 0.6764
Epoch 10/10
140/140 [==============================] - 11s 79ms/step - loss: 0.4
749 - accuracy: 0.6864
```

```
Out[31]: <keras.callbacks.History at 0x29f7417b0>
```

(ii) Train/Test accuracy

In [32]:
```
train_loss, train_accuracy =  LSTM_model.evaluate(train_X, train_label
print(f"Train accuracy: {round(train_accuracy * 100, 2)}%")

#test_label_exp = tf.expand_dims(test_label, 1)

test_loss, test_accuracy = LSTM_model.evaluate(test_X, test_label)
print(f"Test accuracy: {round(test_accuracy * 100, 2)}%")
```

```
 4/44 [=>............................] - ETA: 0s - loss: 0.4774 - ac
curacy: 0.3750

2024-05-05 18:26:33.276006: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_2_grad/concat/split_2/split_dim' with dtype int32
         [[{{node gradients/split_2_grad/concat/split_2/split_di
m}}]]
2024-05-05 18:26:33.276655: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_grad/concat/split/split_dim' with dtype int32
         [[{{node gradients/split_grad/concat/split/split_dim}}]]
2024-05-05 18:26:33.277033: I tensorflow/core/common_runtime/executo
r.cc:1197] [/device:CPU:0] (DEBUG INFO) Executor start aborting (thi
s does not indicate an error and you can ignore this message): INVAL
ID_ARGUMENT: You must feed a value for placeholder tensor 'gradient
s/split_1_grad/concat/split_1/split_dim' with dtype int32
         [[{{node gradients/split_1_grad/concat/split_1/split_di
m}}]]

44/44 [==============================] - 1s 22ms/step - loss: 0.4715
- accuracy: 0.6879
Train accuracy: 68.79%
19/19 [==============================] - 0s 23ms/step - loss: 1.3404
- accuracy: 0.5350
Test accuracy: 53.5%
```

In [ ]: