In [1]:

```
!gdown --id 1OurDQUtbWQacvT32HMqFL7vIUrSMllOp
```

```
Downloading...
From: https://drive.google.com/uc?id=1OurDQUtbWQacvT32HMqFL7vIUrSMllOp
To: /content/preprocessed_data.csv
100% 300k/300k [00:00<00:00, 9.50MB/s]
```

In [2]:

```
!pip install kaggle
```

```
Requirement already satisfied: kaggle in /usr/local/lib/python3.7/dist-packages (1.5.12)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.7/dist-packages (from kaggle) (5.
0.2)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from kaggle) (4.41.1)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from kaggle) (2.23.0)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.7/dist-packages (from kaggle) (1.24.3)
Requirement already satisfied: certifi in /usr/local/lib/python3.7/dist-packages (from kaggle) (2021.5.30
)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.7/dist-packages (from kaggle) (2
.8.1)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.7/dist-packages (from kaggle) (1.15.0)
Requirement already satisfied: text-unidecode>=1.3 in /usr/local/lib/python3.7/dist-packages (from python
-slugify->kaggle) (1.3)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests
->kaggle) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->kag
gle) (2.10)
```

In [7]:

```
!mkdir ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 /root/.kaggle/kaggle.json
!kaggle datasets download -d yekenot/fasttext-crawl-300d-2m
```

```
mkdir: cannot create directory ‘/root/.kaggle’: File exists
Downloading fasttext-crawl-300d-2m.zip to /content
 99% 1.42G/1.44G [00:06<00:00, 238MB/s]
100% 1.44G/1.44G [00:06<00:00, 238MB/s]
```

In [8]:

```
!7z e fasttext-crawl-300d-2m.zip -o/content -r
```

```
7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,64 bits,2 CPUs Intel(R) Xeon(R) CPU @
2.20GHz (406F0),ASM,AES-NI)

Scanning the drive for archives:
  0M Scan                           1 file, 1545551987 bytes (1474 MiB)

Extracting archive: fasttext-crawl-300d-2m.zip
--
Path = fasttext-crawl-300d-2m.zip
Type = zip
Physical Size = 1545551987

  0%              0% - crawl-300d-2M.vec
                           1% - crawl-300d-2M.vec
                  2% - crawl-300d-2M.vec
                           3% - crawl-300d-2M.vec
                  4% - crawl-300d-2M.vec
                           5% - crawl-300d-2M.vec
                  6% - crawl-300d-2M.vec
                           7% - crawl-300d-2M.vec
                  8% - crawl-300d-2M.vec
                           9% - crawl-300d-2M.vec
                 10% - crawl-300d-2M.vec
                          11% - crawl-300d-2M.vec
                          12% - crawl-300d-2M.vec
                          13% - crawl-300d-2M.vec
                          14% - crawl-300d-2M.vec
                          15% - crawl-300d-2M.vec
                          16% - crawl-300d-2M.vec
                          17% - crawl-300d-2M.vec
                          18% - crawl-300d-2M.vec
                          19% - crawl-300d-2M.vec
                          20% - crawl-300d-2M.vec
                          21% - crawl-300d-2M.vec
                          22% - crawl-300d-2M.vec
```

23% - crawl-300d-2M.vec
24% - crawl-300d-2M.vec
25% - crawl-300d-2M.vec
26% - crawl-300d-2M.vec
27% - crawl-300d-2M.vec
28% - crawl-300d-2M.vec
29% - crawl-300d-2M.vec
30% - crawl-300d-2M.vec
31% - crawl-300d-2M.vec
32% - crawl-300d-2M.vec
33% - crawl-300d-2M.vec
34% - crawl-300d-2M.vec
35% - crawl-300d-2M.vec
36% - crawl-300d-2M.vec
37% - crawl-300d-2M.vec
38% - crawl-300d-2M.vec
39% - crawl-300d-2M.vec
40% - crawl-300d-2M.vec
41% - crawl-300d-2M.vec
42% - crawl-300d-2M.vec
43% - crawl-300d-2M.vec
44% - crawl-300d-2M.vec
45% - crawl-300d-2M.vec
46% - crawl-300d-2M.vec
47% - crawl-300d-2M.vec
48% - crawl-300d-2M.vec
49% - crawl-300d-2M.vec
50% - crawl-300d-2M.vec
51% - crawl-300d-2M.vec
52% - crawl-300d-2M.vec
53% - crawl-300d-2M.vec
54% - crawl-300d-2M.vec
55% - crawl-300d-2M.vec
56% - crawl-300d-2M.vec
57% - crawl-300d-2M.vec
58% - crawl-300d-2M.vec
59% - crawl-300d-2M.vec
60% - crawl-300d-2M.vec
61% - crawl-300d-2M.vec
62% - crawl-300d-2M.vec
63% - crawl-300d-2M.vec
64% - crawl-300d-2M.vec
65% - crawl-300d-2M.vec
66% - crawl-300d-2M.vec
67% - crawl-300d-2M.vec
68% - crawl-300d-2M.vec
69% - crawl-300d-2M.vec
70% - crawl-300d-2M.vec
71% - crawl-300d-2M.vec
72% - crawl-300d-2M.vec
73% - crawl-300d-2M.vec
74% - crawl-300d-2M.vec
75% - crawl-300d-2M.vec
76% - crawl-300d-2M.vec
77% - crawl-300d-2M.vec
78% - crawl-300d-2M.vec
79% - crawl-300d-2M.vec
80% - crawl-300d-2M.vec
81% - crawl-300d-2M.vec
82% - crawl-300d-2M.vec
83% - crawl-300d-2M.vec
84% - crawl-300d-2M.vec
85% - crawl-300d-2M.vec
86% - crawl-300d-2M.vec
87% - crawl-300d-2M.vec
88% - crawl-300d-2M.vec
89% - crawl-300d-2M.vec
90% - crawl-300d-2M.vec
91% - crawl-300d-2M.vec
92% - crawl-300d-2M.vec
93% - crawl-300d-2M.vec
94% - crawl-300d-2M.vec
95% - crawl-300d-2M.vec
96% - crawl-300d-2M.vec
97% - crawl-300d-2M.vec
98% - crawl-300d-2M.vec
99% - crawl-300d-2M.vec

```
                    100% - crawl-300d-2M.vec
                    100% 1                  Everything is Ok

Size:       4516698366
Compressed: 1545551987
```

In [11]:

```
# Reading glove vectors in python: https://stackoverflow.com/a/38230349/4084039
def fasttextModel(gloveFile):
    print ("Loading Fasttext Model")
    f = open(gloveFile,'r', encoding="utf8")
    model = {}#for storing word and the corresponding embedding vector for that word
    for line in f:
        splitLine = line.split()#splitting the line and storing it in a list
        word = splitLine[0]#getting the first element and storing it in word
        embedding = np.array([float(val) for val in splitLine[1:]])#obtaining corresponding vector for th
        model[word] = embedding#storing word as key and embedding vector for that word as value
    print ("Done.",len(model)," words loaded!")
    return model
model = fasttextModel('/content/crawl-300d-2M.vec')
```

```
Loading Fasttext Model
Done. 2000000  words loaded!
```

In [12]:

```
#Importing necessary libraries
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
```

In [13]:

```
df=pd.read_csv('preprocessed_data.csv')#creating dataframe using preprocessed_data.csv
```

In [14]:

```
df.head(4)
```

Out[14]:

	Unnamed: 0	source	target
0	0	U wan me to "chop" seat 4 u nt?\n	Do you want me to reserve seat for you or not?\n
1	1	Yup. U reaching. We order some durian pastry a...	Yeap. You reaching? We ordered some Durian pas...
2	2	They become more ex oredi... Mine is like 25.....	They become more expensive already. Mine is li...
3	3	I'm thai. what do u do?\n	I'm Thai. What do you do?\n

In [15]:

```
def preprocess(x):#for removing last character
  x=x[:-1]
  return x
```

In [16]:

```
df['source']=df['source'].apply(preprocess)#preprocessing the source data
df['target']=df['target'].apply(preprocess)#preprocessing the target data
```

In [17]:

```
df=df[['source','target']]
df.head()
```

Out[17]:

	source	target
0	U wan me to "chop" seat 4 u nt?	Do you want me to reserve seat for you or not?
1	Yup. U reaching. We order some durian pastry a...	Yeap. You reaching? We ordered some Durian pas...
2	They become more ex oredi... Mine is like 25.....	They become more expensive already. Mine is li...
3	I'm thai. what do u do?	I'm Thai. What do you do?
4	Hi! How did your week go? Haven heard from you...	Hi! How did your week go? Haven't heard from y...

In [18]:

```
df.shape
```

Out[18]:

```
(2000, 2)
```

In [19]:

```
df=df[df['source'].apply(len)<170]#removing the datapoints where the source sentence length is greater th
df=df[df['target'].apply(len)<200]#removing the datapoints where the target sentence length is greater th
```

In [20]:

```
df.shape
```

Out[20]:

```
(1990, 2)
```

In [21]:

```
from sklearn.model_selection import train_test_split
X=df['source']
y=df['target']
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.01)#splitting the data in the ratio 99:1
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(1970,)
(20,)
(1970,)
(20,)
```

Target:

In [22]:

```
target_tokenizer= Tokenizer()#creating tokenization
target_tokenizer.fit_on_texts(y_train)#fitting on y_train
target_vocab_size= len(target_tokenizer.word_index) + 1#target vocab size
print(len(target_tokenizer.word_index))
```

```
3034
```

In [23]:

```
target_encoded_docs_train = target_tokenizer.texts_to_sequences(y_train)#converting text to integers
target_encoded_docs_test = target_tokenizer.texts_to_sequences(y_test)#converting text to integers
```

In [24]:

```
target_padded_docs_train = pad_sequences(target_encoded_docs_train,padding='post')#padding the sequence w
```

In [25]:

```
target_padded_docs_train.shape
```

Out[25]:

```
(1970, 43)
```

In [26]:

```
target_padded_docs_test = pad_sequences(target_encoded_docs_test,maxlen=target_padded_docs_train.shape[1]
```

In [27]:

```
target_padded_docs_test.shape
```

Out[27]:

```
(20, 43)
```

Source:

In [28]:

```
source_tokenizer= Tokenizer()#creating tokenziation
source_tokenizer.fit_on_texts(X_train)#fitting on X_train
source_vocab_size= len(source_tokenizer.word_index) + 1#source vocab size
print(len(source_tokenizer.word_index))
```

```
3706
```

In [29]:

```
source_encoded_docs_train = source_tokenizer.texts_to_sequences(X_train)#converting text to integers
source_encoded_docs_test = source_tokenizer.texts_to_sequences(X_test)#converting text to integers
```

In [30]:

```
source_padded_docs_train = pad_sequences(source_encoded_docs_train,padding='post')#padding the sequence w
```

In [31]:

```
source_padded_docs_train.shape
```

Out[31]:

```
(1970, 39)
```

In [32]:

```
source_padded_docs_test = pad_sequences(source_encoded_docs_test,maxlen=source_padded_docs_train.shape[1]
```

In [33]:

```
source_padded_docs_test.shape
```

Out[33]:

```
(20, 39)
```

In [34]:

```
#we are reshaping the data because sparse_categorical_crossentropy accepts three dimensional

target_padded_docs_train=target_padded_docs_train.reshape((*target_padded_docs_train.shape,1))
target_padded_docs_test=target_padded_docs_test.reshape((*target_padded_docs_test.shape,1))
```

In [35]:

```
print(target_padded_docs_train.shape)
print(target_padded_docs_test.shape)
```

```
(1970, 43, 1)
(20, 43, 1)
```

In [36]:

```
#we are reshaping the data because sparse_categorical_crossentropy accepts three dimensional

source_padded_docs_train=source_padded_docs_train.reshape((*source_padded_docs_train.shape,1))
source_padded_docs_test=source_padded_docs_test.reshape((*source_padded_docs_test.shape,1))
```

In [37]:

```
print(source_padded_docs_train.shape)
print(source_padded_docs_test.shape)
```

```
(1970, 39, 1)
(20, 39, 1)
```

In [38]:

```
#creating an embedding matrix
embedding_matrix = np.zeros((source_vocab_size, 300))
for word, i in source_tokenizer.word_index.items():
    embedding_vector = model.get(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector
```

Model1:

In [45]:

```
input=tf.keras.layers.Input(shape=(39,))
embed=tf.keras.layers.Embedding(source_vocab_size,300,weights=[embedding_matrix],input_length=source_padc

enc=tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(100))(embed)
repeat=tf.keras.layers.RepeatVector(43)(enc)

dec=tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(128, return_sequences=True))(repeat)
dense=tf.keras.layers.TimeDistributed(tf.keras.layers.Dense(1024, activation='relu'))(dec)
drop=tf.keras.layers.Dropout(0.5)(dense)
output=tf.keras.layers.TimeDistributed(tf.keras.layers.Dense(target_vocab_size, activation='softmax'))(dr

model=tf.keras.models.Model(inputs=input,outputs=output)
model.summary()
```

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_4 (InputLayer)         [(None, 39)]              0
_________________________________________________________________
embedding_3 (Embedding)      (None, 39, 300)           1112100
_________________________________________________________________
bidirectional_6 (Bidirection (None, 200)               320800
_________________________________________________________________
repeat_vector_3 (RepeatVecto (None, 43, 200)           0
_________________________________________________________________
bidirectional_7 (Bidirection (None, 43, 256)           336896
_________________________________________________________________
time_distributed_6 (TimeDist (None, 43, 1024)          263168
_________________________________________________________________
dropout_3 (Dropout)          (None, 43, 1024)          0
_________________________________________________________________
time_distributed_7 (TimeDist (None, 43, 3035)          3110875
=================================================================
Total params: 5,143,839
Trainable params: 4,031,739
Non-trainable params: 1,112,100
_________________________________________________________________
```

In [46]:

```
# Compile model
```

```
model.compile(optimizer=tf.keras.optimizers.Adam(0.01),
              loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

In [47]:

```
model.fit(source_padded_docs_train,target_padded_docs_train,batch_size=1024,epochs=50,
          validation_data=(source_padded_docs_test,target_padded_docs_test))
```

```
Epoch 1/50
2/2 [==============================] - 6s 2s/step - loss: 7.2103 - accuracy: 0.3273 - val_loss: 4.5655 -
val_accuracy: 0.7000
Epoch 2/50
2/2 [==============================] - 1s 417ms/step - loss: 4.4471 - accuracy: 0.6735 - val_loss:
7.5582 - val_accuracy: 0.0186
Epoch 3/50
2/2 [==============================] - 1s 414ms/step - loss: 6.9582 - accuracy: 0.0076 - val_loss:
3.5651 - val_accuracy: 0.7000
Epoch 4/50
2/2 [==============================] - 1s 414ms/step - loss: 3.6384 - accuracy: 0.6735 - val_loss:
3.7226 - val_accuracy: 0.7000
Epoch 5/50
2/2 [==============================] - 1s 418ms/step - loss: 4.4497 - accuracy: 0.6735 - val_loss:
4.3426 - val_accuracy: 0.7000
Epoch 6/50
2/2 [==============================] - 1s 416ms/step - loss: 4.4265 - accuracy: 0.6735 - val_loss:
2.6218 - val_accuracy: 0.7000
Epoch 7/50
2/2 [==============================] - 1s 414ms/step - loss: 3.4221 - accuracy: 0.6735 - val_loss:
2.5167 - val_accuracy: 0.7000
Epoch 8/50
2/2 [==============================] - 1s 413ms/step - loss: 2.9227 - accuracy: 0.6735 - val_loss:
2.8996 - val_accuracy: 0.7000
Epoch 9/50
2/2 [==============================] - 1s 419ms/step - loss: 3.1725 - accuracy: 0.6735 - val_loss:
2.4779 - val_accuracy: 0.7000
Epoch 10/50
2/2 [==============================] - 1s 414ms/step - loss: 2.7962 - accuracy: 0.6735 - val_loss:
2.7062 - val_accuracy: 0.7000
Epoch 11/50
2/2 [==============================] - 1s 415ms/step - loss: 2.9096 - accuracy: 0.6735 - val_loss:
2.3609 - val_accuracy: 0.7000
Epoch 12/50
2/2 [==============================] - 1s 421ms/step - loss: 2.6581 - accuracy: 0.6735 - val_loss:
2.4123 - val_accuracy: 0.7000
Epoch 13/50
2/2 [==============================] - 1s 418ms/step - loss: 2.6712 - accuracy: 0.6735 - val_loss:
2.2605 - val_accuracy: 0.7000
Epoch 14/50
2/2 [==============================] - 1s 418ms/step - loss: 2.5784 - accuracy: 0.6735 - val_loss:
2.2887 - val_accuracy: 0.7000
Epoch 15/50
2/2 [==============================] - 1s 421ms/step - loss: 2.5201 - accuracy: 0.6735 - val_loss:
2.2496 - val_accuracy: 0.7000
Epoch 16/50
2/2 [==============================] - 1s 426ms/step - loss: 2.4976 - accuracy: 0.6735 - val_loss:
2.1687 - val_accuracy: 0.7000
Epoch 17/50
2/2 [==============================] - 1s 415ms/step - loss: 2.4271 - accuracy: 0.6735 - val_loss:
2.1718 - val_accuracy: 0.7000
Epoch 18/50
2/2 [==============================] - 1s 415ms/step - loss: 2.3965 - accuracy: 0.6735 - val_loss:
2.1007 - val_accuracy: 0.7000
Epoch 19/50
2/2 [==============================] - 1s 419ms/step - loss: 2.3583 - accuracy: 0.6735 - val_loss:
2.0527 - val_accuracy: 0.7000
Epoch 20/50
2/2 [==============================] - 1s 416ms/step - loss: 2.3062 - accuracy: 0.6735 - val_loss:
2.0345 - val_accuracy: 0.7000
Epoch 21/50
2/2 [==============================] - 1s 416ms/step - loss: 2.2815 - accuracy: 0.6735 - val_loss:
2.0114 - val_accuracy: 0.7000
Epoch 22/50
2/2 [==============================] - 1s 417ms/step - loss: 2.2571 - accuracy: 0.6735 - val_loss:
1.9703 - val_accuracy: 0.7000
Epoch 23/50
2/2 [==============================] - 1s 414ms/step - loss: 2.2284 - accuracy: 0.6739 - val_loss:
1.9568 - val_accuracy: 0.7000
Epoch 24/50
2/2 [==============================] - 1s 419ms/step - loss: 2.2093 - accuracy: 0.6741 - val_loss:
```

1.9345 - val_accuracy: 0.7000
Epoch 25/50
2/2 [==============================] - 1s 423ms/step - loss: 2.1902 - accuracy: 0.6762 - val_loss:
1.9293 - val_accuracy: 0.7000
Epoch 26/50
2/2 [==============================] - 1s 421ms/step - loss: 2.1758 - accuracy: 0.6768 - val_loss:
1.9102 - val_accuracy: 0.7012
Epoch 27/50
2/2 [==============================] - 1s 421ms/step - loss: 2.1686 - accuracy: 0.6782 - val_loss:
1.9100 - val_accuracy: 0.7012
Epoch 28/50
2/2 [==============================] - 1s 423ms/step - loss: 2.1589 - accuracy: 0.6787 - val_loss:
1.8960 - val_accuracy: 0.7023
Epoch 29/50
2/2 [==============================] - 1s 419ms/step - loss: 2.1424 - accuracy: 0.6781 - val_loss:
1.8937 - val_accuracy: 0.7047
Epoch 30/50
2/2 [==============================] - 1s 418ms/step - loss: 2.1386 - accuracy: 0.6788 - val_loss:
1.8799 - val_accuracy: 0.7047
Epoch 31/50
2/2 [==============================] - 1s 425ms/step - loss: 2.1208 - accuracy: 0.6793 - val_loss:
1.8815 - val_accuracy: 0.7070
Epoch 32/50
2/2 [==============================] - 1s 416ms/step - loss: 2.1138 - accuracy: 0.6806 - val_loss:
1.8644 - val_accuracy: 0.7093
Epoch 33/50
2/2 [==============================] - 1s 421ms/step - loss: 2.1047 - accuracy: 0.6813 - val_loss:
1.8633 - val_accuracy: 0.7128
Epoch 34/50
2/2 [==============================] - 1s 421ms/step - loss: 2.1053 - accuracy: 0.6822 - val_loss:
1.8733 - val_accuracy: 0.7116
Epoch 35/50
2/2 [==============================] - 1s 420ms/step - loss: 2.1264 - accuracy: 0.6831 - val_loss:
1.8731 - val_accuracy: 0.7128
Epoch 36/50
2/2 [==============================] - 1s 424ms/step - loss: 2.1060 - accuracy: 0.6839 - val_loss:
1.8930 - val_accuracy: 0.7105
Epoch 37/50
2/2 [==============================] - 1s 426ms/step - loss: 2.1055 - accuracy: 0.6839 - val_loss:
1.8601 - val_accuracy: 0.7116
Epoch 38/50
2/2 [==============================] - 1s 430ms/step - loss: 2.0861 - accuracy: 0.6822 - val_loss:
1.8554 - val_accuracy: 0.7128
Epoch 39/50
2/2 [==============================] - 1s 428ms/step - loss: 2.0800 - accuracy: 0.6836 - val_loss:
1.8969 - val_accuracy: 0.7105
Epoch 40/50
2/2 [==============================] - 1s 425ms/step - loss: 2.0785 - accuracy: 0.6830 - val_loss:
1.8517 - val_accuracy: 0.7116
Epoch 41/50
2/2 [==============================] - 1s 425ms/step - loss: 2.0739 - accuracy: 0.6841 - val_loss:
1.8609 - val_accuracy: 0.7140
Epoch 42/50
2/2 [==============================] - 1s 424ms/step - loss: 2.0634 - accuracy: 0.6850 - val_loss:
1.8673 - val_accuracy: 0.7128
Epoch 43/50
2/2 [==============================] - 1s 423ms/step - loss: 2.0588 - accuracy: 0.6843 - val_loss:
1.8524 - val_accuracy: 0.7128
Epoch 44/50
2/2 [==============================] - 1s 429ms/step - loss: 2.0561 - accuracy: 0.6860 - val_loss:
1.8803 - val_accuracy: 0.7128
Epoch 45/50
2/2 [==============================] - 1s 434ms/step - loss: 2.0609 - accuracy: 0.6858 - val_loss:
1.8724 - val_accuracy: 0.7116
Epoch 46/50
2/2 [==============================] - 1s 426ms/step - loss: 2.0507 - accuracy: 0.6854 - val_loss:
1.8573 - val_accuracy: 0.7151
Epoch 47/50
2/2 [==============================] - 1s 427ms/step - loss: 2.0548 - accuracy: 0.6862 - val_loss:
1.8646 - val_accuracy: 0.7140
Epoch 48/50
2/2 [==============================] - 1s 425ms/step - loss: 2.0440 - accuracy: 0.6869 - val_loss:
1.8954 - val_accuracy: 0.7140
Epoch 49/50
2/2 [==============================] - 1s 430ms/step - loss: 2.0481 - accuracy: 0.6864 - val_loss:
1.8653 - val_accuracy: 0.7140
Epoch 50/50

```
2/2 [==============================] - 1s 425ms/step - loss: 2.0407 - accuracy: 0.6862 - val_loss:
1.8567 - val_accuracy: 0.7128
```

Out[47]:

```
<tensorflow.python.keras.callbacks.History at 0x7ffa5c5d0d50>
```

In [48]:

```
x=model.predict(source_padded_docs_test[1:2])[0]
```

In [49]:

```
#https://machinelearningmastery.com

index_to_words = {id: word for word, id in target_tokenizer.word_index.items()}
index_to_words[0] = '<PAD>'

' '.join([index_to_words[prediction] for prediction in np.argmax(x, 1)])
```

Out[49]:

```
'i i i i i you you you you you you you you you you <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD>
<PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PA
D> <PAD>'
```

In [50]:

```
print(y_test[:1])
```

```
443    Hi. Everybody! Me. I'm fine. And wish to chat ...
Name: target, dtype: object
```

In [51]:

```
X_test[:1]
```

Out[51]:

```
443    Hi.......everybody! Me.....i'm fine... ñ wish ...
Name: source, dtype: object
```

In [52]:

```
def prediction(x):

  index_to_words = {id: word for word, id in target_tokenizer.word_index.items()}
  index_to_words[0] = '<PAD>'

  y=' '.join([index_to_words[prediction] for prediction in np.argmax(x, 1)])
  return y
for i in range(20):
  print("Input text: ")
  a=list(X_test[i:i+1])
  print(a[0])

  print("Actual Output: ")
  b=list(y_test[i:i+1])
  print(b[0])

  print("Predicted Output: ")
  x=model.predict(source_padded_docs_test[i:i+1])
  y=prediction(x[0])
  y=y.split(' ')
  y_lst=[]
  for i in y:
    if i=='<PAD>':
      continue
    else:
      y_lst.append(i)
  print(' '.join(y_lst))
  print('>'*180)
```

```
Input text:
Hi.......everybody! Me.....i'm fine... ñ wish to chat wif ü guys. Can sms to 96792371. K....
Actual Output:
Hi. Everybody! Me. I'm fine. And wish to chat with you guys. You can message to 96792371. OK.
Predicted Output:
i i i i i you you you you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
U still wan me 2 reg e gown 4 u? But need ur add, IC n matric. Then e 3 measurement.
Actual Output:
Do you still want me to register the gown for you? But I will need your address, IC and matriculation nu
mber. And also the 3 measurement.
Predicted Output:
i i i i i you you you you you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
You study wat course?
Actual Output:
What course do you study?
Predicted Output:
you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
I'm going 2 orchard now already me reaching soon. U reaching?
Actual Output:
I'm going to Orchard now already. I'm reaching soon. Are you reaching?
Predicted Output:
i i i you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
i have a little angel flying around with a hammer. each person he hits gets a little bit of my love and
concern... i think he'll beat the crap out of you.. =p
Actual Output:
I have a little angel flying around with a hammer. Each person he hits gets a little bit of my love and
concern. I think he'll beat the crap out of you.
Predicted Output:
i i i i i i i i i i you you you you you you you you you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Hmm... How early? I think i'll be there about 11... Eh... E stocks qn, i dun understand how he get e
interest... Sigh...
Actual Output:
Hm. How early? I think I'll be there about 11. The stocks question, I don't understand how he get the in
terest. Sigh.
Predicted Output:
i i i i i i i you you you you you you you you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
oh... yup, doin account tut now..   i think bukit panjang is goin to rain soon. hav u decided to go out a
t wad time?
Actual Output:
Oh yes, doing the accounting tutorial now. I think Bukit Panjang is going to rain soon. What time have yo
u decided to go out?
Predicted Output:
i i i i i i i i you you you you you you you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Yup... I will call you in a while? I'm booking my lessons online
Actual Output:
Yeap. I will call you in a while? I'm booking my lessons online.
Predicted Output:
i i i i you you you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Yupz... Juz went to e gym w fren bt she left first...
Actual Output:
Just went to the gym with my friend but she left first.
Predicted Output:
i i i you you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
ür exams when start?
Actual Output:
When will your exams start?
Predicted Output:
you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Haha... Ü mean mich din join her meh?
Actual Output:
Haha. You mean Mich Din joins her?
Predicted Output:
i i you you you you you
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Rain,u busy nw? Y so quiet? Speak up.
Actual Output:
Rain, you are busy now? Why so quiet? Speak up.
Predicted Output:
i i you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
I'm worried abt my hair leh
Actual Output:
I'm worried about my hair.
Predicted Output:
you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Hey, are u still in school? Wanna go lunch with me n wei yi?
Actual Output:
Hey, are you still in school? Do you want to go lunch with me and Wei Yi?
Predicted Output:
i i i i you you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
But he neva ask me wat... Go which toni n guy da academy or wat...
Actual Output:
But he never asks me what. Go which Toni & Guy the academy or what?
Predicted Output:
i i i i you you you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Hey... So how life in e uni?
Actual Output:
Hey. So how's life in the university?
Predicted Output:
i you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Of cos got wash lah... I very sure...
Actual Output:
Of course got wash. I am very sure.
Predicted Output:
i you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
The rest free nt? Den u wan to go chiong after dat?
Actual Output:
Are the rest free or not? Then you want to go rushing after that?
Predicted Output:
i i i you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Okay... Thank you... You sleep early too
Actual Output:
Okay. Thank you. You should sleep early too.
Predicted Output:
i you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Huh muz send resume but i dun haf lei... I lazy 2 prepare lei...
Actual Output:
Must send resume but I don't have it. I am lazy to prepare.
Predicted Output:
i i i you you you you you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

In [53]:

```
import nltk.translate.bleu_score as bleu

bleu_score=[]
for i in range(20):
  b=list(y_test[i:i+1])
  x=model.predict(source_padded_docs_test[i:i+1])
  y=prediction(x[0])
  y=y.split(' ')
  y_lst=[]
  for i in y:
    if i=='<PAD>':
      continue
    else:
      y_lst.append(i)
  bleu_score.append(bleu.sentence_bleu([b[0].split(),],y_lst))
print(bleu_score)
print("The Average Bleu Score is: ",sum(bleu_score)/20)
```

```
/usr/local/lib/python3.7/dist-packages/nltk/translate/bleu_score.py:490: UserWarning:
Corpus/Sentence contains 0 counts of 2-gram overlaps.
BLEU scores might be undesirable; use SmoothingFunction().
  warnings.warn(_msg)
[0.3884935863283276, 0.24405885259598942, 0.39011264866539486, 0.41368954504257255, 0, 0, 0.3871538698781
7624, 0.49432603195143127, 0, 0, 0, 0.4004970149398301, 0, 0.37846125781090306, 0, 0, 0, 0.33125669191122
536, 0, 0]
The Average Bleu Score is:  0.1714024749561925
```

In [ ]: