In [17]:

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

```
Downloading...
From: https://drive.google.com/uc?id=1OurDQUtbWQacvT32HMqFL7vIUrSMllOp
To: /content/preprocessed_data.csv
100% 300k/300k [00:00<00:00, 4.73MB/s]
```

In [18]:

```
!pip install kaggle
```

```
Requirement already satisfied: kaggle in /usr/local/lib/python3.7/dist-packages (1.5.12)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from kaggle) (2.23.0)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.7/dist-packages (from kaggle) (5.
0.2)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.7/dist-packages (from kaggle) (1.15.0)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.7/dist-packages (from kaggle) (2
.8.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.7/dist-packages (from kaggle) (2021.5.30
)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.7/dist-packages (from kaggle) (1.24.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from kaggle) (4.41.1)
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)
Requirement already satisfied: text-unidecode>=1.3 in /usr/local/lib/python3.7/dist-packages (from python
-slugify->kaggle) (1.3)
```

In [19]:

```
!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
100% 1.44G/1.44G [00:17<00:00, 64.2MB/s]
100% 1.44G/1.44G [00:17<00:00, 86.6MB/s]
```

In [20]:

```
!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 @
2.00GHz (50653),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
                         Everything is Ok

Size:       4516698366
Compressed: 1545551987
```

In [21]:

```
#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 [22]:

```
# 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 [23]:

```
df=pd.read_csv('preprocessed_data.csv')#reading data into DataFrame
```

In [24]:

```
df.head(4)#displaying top 4 datapoints
```

Out[24]:

	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 [25]:

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

In [26]:

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

In [27]:

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

Out[27]:

	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 [28]:

```
df.shape#shape of DataFrame
```

Out[28]:

```
(2000, 2)
```

In [29]:

```
df=df[df['source'].apply(len)<170]#removing source sentences of length greater than or equal to 170
df=df[df['target'].apply(len)<200]#removing target sentences of length greater than or equal to 200
```

In [30]:

```
df.shape#shape of DataFrame
```

Out[30]:

```
(1990, 2)
```

In [31]:

```
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,)
```

In [32]:

```
X_train.to_csv('X_train.csv')
y_train.to_csv('y_train.csv')
X_test.to_csv('X_test.csv')
y_test.to_csv('y_test.csv')
```

Target:

In [33]:

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

```
3033
```

In [34]:

```
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 [35]:

```
target_padded_docs_train = pad_sequences(target_encoded_docs_train,padding='post')#padding to maxlength
```

In [36]:

```
target_padded_docs_train.shape
```

Out[36]:

```
(1970, 43)
```

In [37]:

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

In [38]:

```
target_padded_docs_test.shape
```

Out[38]:

```
(20, 43)
```

Source:

In [39]:

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

```
3698
```

In [40]:

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

In [41]:

```
source_padded_docs_train = pad_sequences(source_encoded_docs_train,maxlen=target_padded_docs_train.shape[
```

In [42]:

```
source_padded_docs_train.shape
```

Out[42]:

```
(1970, 43)
```

In [43]:

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

In [44]:

```
source_padded_docs_test.shape
```

Out[44]:

```
(20, 43)
```

In [45]:

```
#we are reshaping the dataset because the sparese_categorical_crossentropy requires data to be three dim

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 [46]:

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

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

In [47]:

```
#we are reshaping the dataset because the sparese_categorical_crossentropy requires data to be three dim

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 [48]:

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

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

In [49]:

```
import pandas as pd
pd.DataFrame(source_encoded_docs_train).to_csv("source_encoded_docs_train.csv")
pd.DataFrame(source_encoded_docs_test).to_csv("source_encoded_docs_test.csv")
pd.DataFrame(target_encoded_docs_train).to_csv("target_encoded_docs_train.csv")
pd.DataFrame(target_encoded_docs_test).to_csv("target_encoded_docs_test.csv")
```

In [50]:

```
#creating 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
```

In [51]:

```
embedding_matrix.shape
```

Out[51]:

```
(3699, 300)
```

Model1:

In [73]:

```
input=tf.keras.layers.Input(shape=(43,))
embed=tf.keras.layers.Embedding(source_vocab_size,300,weights=[embedding_matrix],input_length=source_padc
lstm1=tf.keras.layers.LSTM(100, return_sequences=True)(embed)
output=tf.keras.layers.TimeDistributed(tf.keras.layers.Dense(target_vocab_size, activation='softmax'))(ls
model=tf.keras.models.Model(inputs=input,outputs=output)
model.summary()
```

```
Model: "model_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_5 (InputLayer)         [(None, 43)]              0
_________________________________________________________________
embedding_4 (Embedding)      (None, 43, 300)           1109700
_________________________________________________________________
lstm_4 (LSTM)                (None, 43, 100)           160400
_________________________________________________________________
time_distributed_4 (TimeDist (None, 43, 3034)          306434
=================================================================
Total params: 1,576,534
Trainable params: 466,834
Non-trainable params: 1,109,700
_________________________________________________________________
```

In [74]:

```
# Compile model
model.compile(optimizer=tf.keras.optimizers.Adam(0.001),
              loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

In [75]:

```
model.fit(source_padded_docs_train,target_padded_docs_train,batch_size=1024,epochs=200,
          validation_data=(source_padded_docs_test,target_padded_docs_test))
```

```
Epoch 1/200
2/2 [==============================] - 2s 538ms/step - loss: 8.0101 - accuracy: 0.2927 - val_loss:
7.9882 - val_accuracy: 0.6791
Epoch 2/200
2/2 [==============================] - 0s 195ms/step - loss: 7.9829 - accuracy: 0.6694 - val_loss:
7.9436 - val_accuracy: 0.6837
Epoch 3/200
2/2 [==============================] - 0s 192ms/step - loss: 7.9329 - accuracy: 0.6736 - val_loss:
7.8498 - val_accuracy: 0.6837
Epoch 4/200
2/2 [==============================] - 0s 198ms/step - loss: 7.8248 - accuracy: 0.6739 - val_loss:
7.6467 - val_accuracy: 0.6837
Epoch 5/200
2/2 [==============================] - 0s 199ms/step - loss: 7.5876 - accuracy: 0.6739 - val_loss:
7.2696 - val_accuracy: 0.6837
Epoch 6/200
2/2 [==============================] - 0s 192ms/step - loss: 7.1973 - accuracy: 0.6738 - val_loss:
6.8527 - val_accuracy: 0.6837
Epoch 7/200
2/2 [==============================] - 0s 198ms/step - loss: 6.7971 - accuracy: 0.6738 - val_loss:
6.4778 - val_accuracy: 0.6837
Epoch 8/200
2/2 [==============================] - 0s 193ms/step - loss: 6.4322 - accuracy: 0.6738 - val_loss:
6.1163 - val_accuracy: 0.6837
Epoch 9/200
2/2 [==============================] - 0s 195ms/step - loss: 6.0793 - accuracy: 0.6737 - val_loss:
5.7553 - val_accuracy: 0.6837
Epoch 10/200
2/2 [==============================] - 0s 199ms/step - loss: 5.7251 - accuracy: 0.6737 - val_loss:
5.3881 - val_accuracy: 0.6837
Epoch 11/200
2/2 [==============================] - 0s 194ms/step - loss: 5.3655 - accuracy: 0.6737 - val_loss:
5.0132 - val_accuracy: 0.6837
Epoch 12/200
2/2 [==============================] - 0s 196ms/step - loss: 4.9990 - accuracy: 0.6737 - val_loss:
4.6350 - val_accuracy: 0.6837
Epoch 13/200
2/2 [==============================] - 0s 195ms/step - loss: 4.6313 - accuracy: 0.6737 - val_loss:
4.2619 - val_accuracy: 0.6837
Epoch 14/200
2/2 [==============================] - 0s 197ms/step - loss: 4.2699 - accuracy: 0.6737 - val_loss:
3.9042 - val_accuracy: 0.6837
Epoch 15/200
2/2 [==============================] - 0s 194ms/step - loss: 3.9267 - accuracy: 0.6737 - val_loss:
3.5719 - val_accuracy: 0.6837
Epoch 16/200
2/2 [==============================] - 0s 195ms/step - loss: 3.6112 - accuracy: 0.6737 - val_loss:
3.2737 - val_accuracy: 0.6837
Epoch 17/200
2/2 [==============================] - 0s 194ms/step - loss: 3.3295 - accuracy: 0.6737 - val_loss:
3.0183 - val_accuracy: 0.6837
Epoch 18/200
```

```
2/2 [==============================] - 0s 194ms/step - loss: 3.0919 - accuracy: 0.6737 - val_loss:
2.8150 - val_accuracy: 0.6837
Epoch 19/200
2/2 [==============================] - 0s 193ms/step - loss: 2.9066 - accuracy: 0.6737 - val_loss:
2.6693 - val_accuracy: 0.6837
Epoch 20/200
2/2 [==============================] - 0s 196ms/step - loss: 2.7791 - accuracy: 0.6737 - val_loss:
2.5787 - val_accuracy: 0.6837
Epoch 21/200
2/2 [==============================] - 0s 196ms/step - loss: 2.7035 - accuracy: 0.6737 - val_loss:
2.5324 - val_accuracy: 0.6837
Epoch 22/200
2/2 [==============================] - 0s 192ms/step - loss: 2.6682 - accuracy: 0.6737 - val_loss:
2.5152 - val_accuracy: 0.6837
Epoch 23/200
2/2 [==============================] - 0s 196ms/step - loss: 2.6574 - accuracy: 0.6737 - val_loss:
2.5130 - val_accuracy: 0.6837
Epoch 24/200
2/2 [==============================] - 0s 194ms/step - loss: 2.6585 - accuracy: 0.6737 - val_loss:
2.5153 - val_accuracy: 0.6837
Epoch 25/200
2/2 [==============================] - 0s 194ms/step - loss: 2.6616 - accuracy: 0.6737 - val_loss:
2.5153 - val_accuracy: 0.6837
Epoch 26/200
2/2 [==============================] - 0s 193ms/step - loss: 2.6609 - accuracy: 0.6737 - val_loss:
2.5083 - val_accuracy: 0.6837
Epoch 27/200
2/2 [==============================] - 0s 197ms/step - loss: 2.6524 - accuracy: 0.6737 - val_loss:
2.4880 - val_accuracy: 0.6837
Epoch 28/200
2/2 [==============================] - 0s 192ms/step - loss: 2.6330 - accuracy: 0.6737 - val_loss:
2.4732 - val_accuracy: 0.6837
Epoch 29/200
2/2 [==============================] - 0s 201ms/step - loss: 2.6203 - accuracy: 0.6737 - val_loss:
2.4701 - val_accuracy: 0.6837
Epoch 30/200
2/2 [==============================] - 0s 193ms/step - loss: 2.6151 - accuracy: 0.6737 - val_loss:
2.4576 - val_accuracy: 0.6837
Epoch 31/200
2/2 [==============================] - 0s 193ms/step - loss: 2.6016 - accuracy: 0.6737 - val_loss:
2.4411 - val_accuracy: 0.6837
Epoch 32/200
2/2 [==============================] - 0s 195ms/step - loss: 2.5870 - accuracy: 0.6737 - val_loss:
2.4277 - val_accuracy: 0.6837
Epoch 33/200
2/2 [==============================] - 0s 196ms/step - loss: 2.5777 - accuracy: 0.6737 - val_loss:
2.4183 - val_accuracy: 0.6837
Epoch 34/200
2/2 [==============================] - 0s 194ms/step - loss: 2.5710 - accuracy: 0.6737 - val_loss:
2.4095 - val_accuracy: 0.6837
Epoch 35/200
2/2 [==============================] - 0s 196ms/step - loss: 2.5628 - accuracy: 0.6737 - val_loss:
2.3998 - val_accuracy: 0.6837
Epoch 36/200
2/2 [==============================] - 0s 193ms/step - loss: 2.5527 - accuracy: 0.6737 - val_loss:
2.3913 - val_accuracy: 0.6837
Epoch 37/200
2/2 [==============================] - 0s 198ms/step - loss: 2.5440 - accuracy: 0.6737 - val_loss:
2.3896 - val_accuracy: 0.6837
Epoch 38/200
2/2 [==============================] - 0s 195ms/step - loss: 2.5410 - accuracy: 0.6737 - val_loss:
2.3883 - val_accuracy: 0.6837
Epoch 39/200
2/2 [==============================] - 0s 195ms/step - loss: 2.5376 - accuracy: 0.6737 - val_loss:
2.3790 - val_accuracy: 0.6837
Epoch 40/200
2/2 [==============================] - 0s 194ms/step - loss: 2.5277 - accuracy: 0.6737 - val_loss:
2.3654 - val_accuracy: 0.6837
Epoch 41/200
2/2 [==============================] - 0s 196ms/step - loss: 2.5155 - accuracy: 0.6737 - val_loss:
2.3525 - val_accuracy: 0.6837
Epoch 42/200
2/2 [==============================] - 0s 195ms/step - loss: 2.5042 - accuracy: 0.6737 - val_loss:
2.3428 - val_accuracy: 0.6837
Epoch 43/200
2/2 [==============================] - 0s 197ms/step - loss: 2.4952 - accuracy: 0.6737 - val_loss:
2.3367 - val_accuracy: 0.6837
Epoch 44/200
```

```
Epoch 44/200
2/2 [==============================] - 0s 194ms/step - loss: 2.4882 - accuracy: 0.6737 - val_loss:
2.3307 - val_accuracy: 0.6837
Epoch 45/200
2/2 [==============================] - 0s 199ms/step - loss: 2.4802 - accuracy: 0.6737 - val_loss:
2.3221 - val_accuracy: 0.6837
Epoch 46/200
2/2 [==============================] - 0s 198ms/step - loss: 2.4699 - accuracy: 0.6737 - val_loss:
2.3119 - val_accuracy: 0.6837
Epoch 47/200
2/2 [==============================] - 0s 195ms/step - loss: 2.4589 - accuracy: 0.6737 - val_loss:
2.3011 - val_accuracy: 0.6837
Epoch 48/200
2/2 [==============================] - 0s 195ms/step - loss: 2.4478 - accuracy: 0.6737 - val_loss:
2.2892 - val_accuracy: 0.6837
Epoch 49/200
2/2 [==============================] - 0s 198ms/step - loss: 2.4354 - accuracy: 0.6737 - val_loss:
2.2772 - val_accuracy: 0.6837
Epoch 50/200
2/2 [==============================] - 0s 198ms/step - loss: 2.4228 - accuracy: 0.6737 - val_loss:
2.2652 - val_accuracy: 0.6837
Epoch 51/200
2/2 [==============================] - 0s 196ms/step - loss: 2.4101 - accuracy: 0.6737 - val_loss:
2.2524 - val_accuracy: 0.6837
Epoch 52/200
2/2 [==============================] - 0s 198ms/step - loss: 2.3979 - accuracy: 0.6737 - val_loss:
2.2417 - val_accuracy: 0.6837
Epoch 53/200
2/2 [==============================] - 0s 195ms/step - loss: 2.3885 - accuracy: 0.6737 - val_loss:
2.2336 - val_accuracy: 0.6837
Epoch 54/200
2/2 [==============================] - 0s 197ms/step - loss: 2.3797 - accuracy: 0.6737 - val_loss:
2.2251 - val_accuracy: 0.6837
Epoch 55/200
2/2 [==============================] - 0s 193ms/step - loss: 2.3701 - accuracy: 0.6737 - val_loss:
2.2144 - val_accuracy: 0.6837
Epoch 56/200
2/2 [==============================] - 0s 204ms/step - loss: 2.3592 - accuracy: 0.6737 - val_loss:
2.2030 - val_accuracy: 0.6837
Epoch 57/200
2/2 [==============================] - 0s 202ms/step - loss: 2.3480 - accuracy: 0.6737 - val_loss:
2.1919 - val_accuracy: 0.6837
Epoch 58/200
2/2 [==============================] - 0s 198ms/step - loss: 2.3366 - accuracy: 0.6737 - val_loss:
2.1809 - val_accuracy: 0.6837
Epoch 59/200
2/2 [==============================] - 0s 195ms/step - loss: 2.3253 - accuracy: 0.6737 - val_loss:
2.1695 - val_accuracy: 0.6837
Epoch 60/200
2/2 [==============================] - 0s 197ms/step - loss: 2.3135 - accuracy: 0.6737 - val_loss:
2.1573 - val_accuracy: 0.6837
Epoch 61/200
2/2 [==============================] - 0s 195ms/step - loss: 2.3014 - accuracy: 0.6737 - val_loss:
2.1459 - val_accuracy: 0.6837
Epoch 62/200
2/2 [==============================] - 0s 196ms/step - loss: 2.2900 - accuracy: 0.6737 - val_loss:
2.1359 - val_accuracy: 0.6837
Epoch 63/200
2/2 [==============================] - 0s 195ms/step - loss: 2.2796 - accuracy: 0.6737 - val_loss:
2.1264 - val_accuracy: 0.6837
Epoch 64/200
2/2 [==============================] - 0s 203ms/step - loss: 2.2698 - accuracy: 0.6737 - val_loss:
2.1175 - val_accuracy: 0.6837
Epoch 65/200
2/2 [==============================] - 0s 200ms/step - loss: 2.2606 - accuracy: 0.6737 - val_loss:
2.1087 - val_accuracy: 0.6837
Epoch 66/200
2/2 [==============================] - 0s 193ms/step - loss: 2.2518 - accuracy: 0.6737 - val_loss:
2.1010 - val_accuracy: 0.6837
Epoch 67/200
2/2 [==============================] - 0s 199ms/step - loss: 2.2436 - accuracy: 0.6737 - val_loss:
2.0938 - val_accuracy: 0.6837
Epoch 68/200
2/2 [==============================] - 0s 196ms/step - loss: 2.2360 - accuracy: 0.6737 - val_loss:
2.0870 - val_accuracy: 0.6837
Epoch 69/200
2/2 [==============================] - 0s 200ms/step - loss: 2.2287 - accuracy: 0.6737 - val_loss:
```

```
2.0800 - val_accuracy: 0.6837
Epoch 70/200
2/2 [==============================] - 0s 198ms/step - loss: 2.2217 - accuracy: 0.6737 - val_loss:
2.0731 - val_accuracy: 0.6837
Epoch 71/200
2/2 [==============================] - 0s 197ms/step - loss: 2.2152 - accuracy: 0.6746 - val_loss:
2.0672 - val_accuracy: 0.6860
Epoch 72/200
2/2 [==============================] - 0s 195ms/step - loss: 2.2088 - accuracy: 0.6756 - val_loss:
2.0618 - val_accuracy: 0.6860
Epoch 73/200
2/2 [==============================] - 0s 196ms/step - loss: 2.2028 - accuracy: 0.6756 - val_loss:
2.0563 - val_accuracy: 0.6860
Epoch 74/200
2/2 [==============================] - 0s 201ms/step - loss: 2.1971 - accuracy: 0.6756 - val_loss:
2.0507 - val_accuracy: 0.6860
Epoch 75/200
2/2 [==============================] - 0s 197ms/step - loss: 2.1915 - accuracy: 0.6756 - val_loss:
2.0454 - val_accuracy: 0.6860
Epoch 76/200
2/2 [==============================] - 0s 197ms/step - loss: 2.1861 - accuracy: 0.6756 - val_loss:
2.0407 - val_accuracy: 0.6860
Epoch 77/200
2/2 [==============================] - 0s 194ms/step - loss: 2.1810 - accuracy: 0.6756 - val_loss:
2.0360 - val_accuracy: 0.6860
Epoch 78/200
2/2 [==============================] - 0s 194ms/step - loss: 2.1761 - accuracy: 0.6756 - val_loss:
2.0318 - val_accuracy: 0.6860
Epoch 79/200
2/2 [==============================] - 0s 197ms/step - loss: 2.1715 - accuracy: 0.6756 - val_loss:
2.0275 - val_accuracy: 0.6860
Epoch 80/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1671 - accuracy: 0.6756 - val_loss:
2.0234 - val_accuracy: 0.6860
Epoch 81/200
2/2 [==============================] - 0s 195ms/step - loss: 2.1629 - accuracy: 0.6756 - val_loss:
2.0198 - val_accuracy: 0.6860
Epoch 82/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1590 - accuracy: 0.6756 - val_loss:
2.0162 - val_accuracy: 0.6860
Epoch 83/200
2/2 [==============================] - 0s 197ms/step - loss: 2.1552 - accuracy: 0.6756 - val_loss:
2.0128 - val_accuracy: 0.6860
Epoch 84/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1516 - accuracy: 0.6756 - val_loss:
2.0094 - val_accuracy: 0.6860
Epoch 85/200
2/2 [==============================] - 0s 194ms/step - loss: 2.1482 - accuracy: 0.6756 - val_loss:
2.0061 - val_accuracy: 0.6860
Epoch 86/200
2/2 [==============================] - 0s 200ms/step - loss: 2.1448 - accuracy: 0.6756 - val_loss:
2.0029 - val_accuracy: 0.6860
Epoch 87/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1416 - accuracy: 0.6756 - val_loss:
2.0000 - val_accuracy: 0.6860
Epoch 88/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1385 - accuracy: 0.6756 - val_loss:
1.9972 - val_accuracy: 0.6860
Epoch 89/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1355 - accuracy: 0.6756 - val_loss:
1.9945 - val_accuracy: 0.6860
Epoch 90/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1326 - accuracy: 0.6756 - val_loss:
1.9917 - val_accuracy: 0.6860
Epoch 91/200
2/2 [==============================] - 0s 194ms/step - loss: 2.1297 - accuracy: 0.6756 - val_loss:
1.9890 - val_accuracy: 0.6860
Epoch 92/200
2/2 [==============================] - 0s 199ms/step - loss: 2.1270 - accuracy: 0.6756 - val_loss:
1.9864 - val_accuracy: 0.6860
Epoch 93/200
2/2 [==============================] - 0s 197ms/step - loss: 2.1243 - accuracy: 0.6756 - val_loss:
1.9838 - val_accuracy: 0.6860
Epoch 94/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1217 - accuracy: 0.6756 - val_loss:
1.9815 - val_accuracy: 0.6860
Epoch 95/200
```

```
2/2 [==============================] - 0s 196ms/step - loss: 2.1191 - accuracy: 0.6756 - val_loss:
1.9791 - val_accuracy: 0.6860
Epoch 96/200
2/2 [==============================] - 0s 197ms/step - loss: 2.1166 - accuracy: 0.6756 - val_loss:
1.9769 - val_accuracy: 0.6860
Epoch 97/200
2/2 [==============================] - 0s 197ms/step - loss: 2.1142 - accuracy: 0.6756 - val_loss:
1.9746 - val_accuracy: 0.6860
Epoch 98/200
2/2 [==============================] - 0s 197ms/step - loss: 2.1117 - accuracy: 0.6756 - val_loss:
1.9725 - val_accuracy: 0.6860
Epoch 99/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1094 - accuracy: 0.6756 - val_loss:
1.9704 - val_accuracy: 0.6860
Epoch 100/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1070 - accuracy: 0.6756 - val_loss:
1.9683 - val_accuracy: 0.6860
Epoch 101/200
2/2 [==============================] - 0s 197ms/step - loss: 2.1047 - accuracy: 0.6757 - val_loss:
1.9662 - val_accuracy: 0.6860
Epoch 102/200
2/2 [==============================] - 0s 201ms/step - loss: 2.1024 - accuracy: 0.6757 - val_loss:
1.9641 - val_accuracy: 0.6860
Epoch 103/200
2/2 [==============================] - 0s 196ms/step - loss: 2.1001 - accuracy: 0.6757 - val_loss:
1.9620 - val_accuracy: 0.6860
Epoch 104/200
2/2 [==============================] - 0s 195ms/step - loss: 2.0978 - accuracy: 0.6757 - val_loss:
1.9598 - val_accuracy: 0.6860
Epoch 105/200
2/2 [==============================] - 0s 196ms/step - loss: 2.0956 - accuracy: 0.6757 - val_loss:
1.9578 - val_accuracy: 0.6860
Epoch 106/200
2/2 [==============================] - 0s 196ms/step - loss: 2.0933 - accuracy: 0.6757 - val_loss:
1.9559 - val_accuracy: 0.6860
Epoch 107/200
2/2 [==============================] - 0s 194ms/step - loss: 2.0911 - accuracy: 0.6758 - val_loss:
1.9541 - val_accuracy: 0.6872
Epoch 108/200
2/2 [==============================] - 0s 198ms/step - loss: 2.0889 - accuracy: 0.6759 - val_loss:
1.9521 - val_accuracy: 0.6872
Epoch 109/200
2/2 [==============================] - 0s 195ms/step - loss: 2.0867 - accuracy: 0.6760 - val_loss:
1.9501 - val_accuracy: 0.6872
Epoch 110/200
2/2 [==============================] - 0s 197ms/step - loss: 2.0844 - accuracy: 0.6768 - val_loss:
1.9480 - val_accuracy: 0.6884
Epoch 111/200
2/2 [==============================] - 0s 200ms/step - loss: 2.0823 - accuracy: 0.6777 - val_loss:
1.9461 - val_accuracy: 0.6895
Epoch 112/200
2/2 [==============================] - 0s 196ms/step - loss: 2.0801 - accuracy: 0.6779 - val_loss:
1.9442 - val_accuracy: 0.6895
Epoch 113/200
2/2 [==============================] - 0s 195ms/step - loss: 2.0780 - accuracy: 0.6781 - val_loss:
1.9424 - val_accuracy: 0.6895
Epoch 114/200
2/2 [==============================] - 0s 196ms/step - loss: 2.0759 - accuracy: 0.6784 - val_loss:
1.9405 - val_accuracy: 0.6907
Epoch 115/200
2/2 [==============================] - 0s 195ms/step - loss: 2.0737 - accuracy: 0.6788 - val_loss:
1.9388 - val_accuracy: 0.6907
Epoch 116/200
2/2 [==============================] - 0s 199ms/step - loss: 2.0717 - accuracy: 0.6790 - val_loss:
1.9369 - val_accuracy: 0.6907
Epoch 117/200
2/2 [==============================] - 0s 197ms/step - loss: 2.0695 - accuracy: 0.6797 - val_loss:
1.9349 - val_accuracy: 0.6907
Epoch 118/200
2/2 [==============================] - 0s 201ms/step - loss: 2.0675 - accuracy: 0.6801 - val_loss:
1.9331 - val_accuracy: 0.6907
Epoch 119/200
2/2 [==============================] - 0s 199ms/step - loss: 2.0654 - accuracy: 0.6806 - val_loss:
1.9312 - val_accuracy: 0.6930
Epoch 120/200
2/2 [==============================] - 0s 197ms/step - loss: 2.0633 - accuracy: 0.6814 - val_loss:
1.9295 - val_accuracy: 0.6930
```

```
Epoch 121/200
2/2 [==============================] - 0s 197ms/step - loss: 2.0612 - accuracy: 0.6826 - val_loss: 1.9277 - val_accuracy: 0.6930
Epoch 122/200
2/2 [==============================] - 0s 198ms/step - loss: 2.0591 - accuracy: 0.6849 - val_loss: 1.9258 - val_accuracy: 0.6930
Epoch 123/200
2/2 [==============================] - 0s 197ms/step - loss: 2.0570 - accuracy: 0.6861 - val_loss: 1.9239 - val_accuracy: 0.6930
Epoch 124/200
2/2 [==============================] - 0s 198ms/step - loss: 2.0549 - accuracy: 0.6872 - val_loss: 1.9219 - val_accuracy: 0.6942
Epoch 125/200
2/2 [==============================] - 0s 197ms/step - loss: 2.0528 - accuracy: 0.6880 - val_loss: 1.9202 - val_accuracy: 0.6965
Epoch 126/200
2/2 [==============================] - 0s 208ms/step - loss: 2.0507 - accuracy: 0.6887 - val_loss: 1.9184 - val_accuracy: 0.6965
Epoch 127/200
2/2 [==============================] - 0s 196ms/step - loss: 2.0486 - accuracy: 0.6896 - val_loss: 1.9165 - val_accuracy: 0.6965
Epoch 128/200
2/2 [==============================] - 0s 195ms/step - loss: 2.0464 - accuracy: 0.6899 - val_loss: 1.9146 - val_accuracy: 0.6977
Epoch 129/200
2/2 [==============================] - 0s 196ms/step - loss: 2.0443 - accuracy: 0.6902 - val_loss: 1.9128 - val_accuracy: 0.6977
Epoch 130/200
2/2 [==============================] - 0s 201ms/step - loss: 2.0421 - accuracy: 0.6904 - val_loss: 1.9110 - val_accuracy: 0.6988
Epoch 131/200
2/2 [==============================] - 0s 202ms/step - loss: 2.0399 - accuracy: 0.6906 - val_loss: 1.9092 - val_accuracy: 0.7000
Epoch 132/200
2/2 [==============================] - 0s 195ms/step - loss: 2.0376 - accuracy: 0.6910 - val_loss: 1.9074 - val_accuracy: 0.7000
Epoch 133/200
2/2 [==============================] - 0s 196ms/step - loss: 2.0354 - accuracy: 0.6913 - val_loss: 1.9055 - val_accuracy: 0.7000
Epoch 134/200
2/2 [==============================] - 0s 196ms/step - loss: 2.0331 - accuracy: 0.6916 - val_loss: 1.9037 - val_accuracy: 0.7000
Epoch 135/200
2/2 [==============================] - 0s 199ms/step - loss: 2.0309 - accuracy: 0.6920 - val_loss: 1.9017 - val_accuracy: 0.7000
Epoch 136/200
2/2 [==============================] - 0s 194ms/step - loss: 2.0285 - accuracy: 0.6922 - val_loss: 1.8998 - val_accuracy: 0.7000
Epoch 137/200
2/2 [==============================] - 0s 198ms/step - loss: 2.0262 - accuracy: 0.6924 - val_loss: 1.8978 - val_accuracy: 0.7012
Epoch 138/200
2/2 [==============================] - 0s 195ms/step - loss: 2.0238 - accuracy: 0.6927 - val_loss: 1.8959 - val_accuracy: 0.7012
Epoch 139/200
2/2 [==============================] - 0s 194ms/step - loss: 2.0214 - accuracy: 0.6928 - val_loss: 1.8940 - val_accuracy: 0.7023
Epoch 140/200
2/2 [==============================] - 0s 199ms/step - loss: 2.0189 - accuracy: 0.6930 - val_loss: 1.8922 - val_accuracy: 0.7023
Epoch 141/200
2/2 [==============================] - 0s 195ms/step - loss: 2.0165 - accuracy: 0.6932 - val_loss: 1.8903 - val_accuracy: 0.7035
Epoch 142/200
2/2 [==============================] - 0s 196ms/step - loss: 2.0139 - accuracy: 0.6934 - val_loss: 1.8884 - val_accuracy: 0.7035
Epoch 143/200
2/2 [==============================] - 0s 197ms/step - loss: 2.0114 - accuracy: 0.6935 - val_loss: 1.8864 - val_accuracy: 0.7035
Epoch 144/200
2/2 [==============================] - 0s 197ms/step - loss: 2.0089 - accuracy: 0.6937 - val_loss: 1.8846 - val_accuracy: 0.7035
Epoch 145/200
2/2 [==============================] - 0s 198ms/step - loss: 2.0063 - accuracy: 0.6940 - val_loss: 1.8826 - val_accuracy: 0.7035
Epoch 146/200
2/2 [==============================] - 0s 200ms/step - loss: 2.0037 - accuracy: 0.6942 - val_loss:
```

```
1.8806 - val_accuracy: 0.7035
Epoch 147/200
2/2 [==============================] - 0s 201ms/step - loss: 2.0011 - accuracy: 0.6944 - val_loss: 
1.8787 - val_accuracy: 0.7035
Epoch 148/200
2/2 [==============================] - 0s 203ms/step - loss: 1.9984 - accuracy: 0.6947 - val_loss: 
1.8768 - val_accuracy: 0.7035
Epoch 149/200
2/2 [==============================] - 0s 198ms/step - loss: 1.9958 - accuracy: 0.6951 - val_loss: 
1.8749 - val_accuracy: 0.7035
Epoch 150/200
2/2 [==============================] - 0s 198ms/step - loss: 1.9931 - accuracy: 0.6952 - val_loss: 
1.8730 - val_accuracy: 0.7035
Epoch 151/200
2/2 [==============================] - 0s 195ms/step - loss: 1.9904 - accuracy: 0.6955 - val_loss: 
1.8712 - val_accuracy: 0.7035
Epoch 152/200
2/2 [==============================] - 0s 195ms/step - loss: 1.9877 - accuracy: 0.6958 - val_loss: 
1.8693 - val_accuracy: 0.7023
Epoch 153/200
2/2 [==============================] - 0s 194ms/step - loss: 1.9849 - accuracy: 0.6961 - val_loss: 
1.8674 - val_accuracy: 0.7023
Epoch 154/200
2/2 [==============================] - 0s 199ms/step - loss: 1.9822 - accuracy: 0.6964 - val_loss: 
1.8656 - val_accuracy: 0.7035
Epoch 155/200
2/2 [==============================] - 0s 196ms/step - loss: 1.9794 - accuracy: 0.6967 - val_loss: 
1.8636 - val_accuracy: 0.7047
Epoch 156/200
2/2 [==============================] - 0s 197ms/step - loss: 1.9765 - accuracy: 0.6969 - val_loss: 
1.8616 - val_accuracy: 0.7047
Epoch 157/200
2/2 [==============================] - 0s 197ms/step - loss: 1.9737 - accuracy: 0.6971 - val_loss: 
1.8597 - val_accuracy: 0.7047
Epoch 158/200
2/2 [==============================] - 0s 196ms/step - loss: 1.9708 - accuracy: 0.6973 - val_loss: 
1.8578 - val_accuracy: 0.7047
Epoch 159/200
2/2 [==============================] - 0s 195ms/step - loss: 1.9679 - accuracy: 0.6977 - val_loss: 
1.8559 - val_accuracy: 0.7047
Epoch 160/200
2/2 [==============================] - 0s 194ms/step - loss: 1.9650 - accuracy: 0.6979 - val_loss: 
1.8539 - val_accuracy: 0.7058
Epoch 161/200
2/2 [==============================] - 0s 197ms/step - loss: 1.9621 - accuracy: 0.6981 - val_loss: 
1.8518 - val_accuracy: 0.7058
Epoch 162/200
2/2 [==============================] - 0s 199ms/step - loss: 1.9591 - accuracy: 0.6981 - val_loss: 
1.8497 - val_accuracy: 0.7070
Epoch 163/200
2/2 [==============================] - 0s 196ms/step - loss: 1.9561 - accuracy: 0.6983 - val_loss: 
1.8477 - val_accuracy: 0.7093
Epoch 164/200
2/2 [==============================] - 0s 196ms/step - loss: 1.9532 - accuracy: 0.6991 - val_loss: 
1.8458 - val_accuracy: 0.7093
Epoch 165/200
2/2 [==============================] - 0s 195ms/step - loss: 1.9501 - accuracy: 0.6991 - val_loss: 
1.8439 - val_accuracy: 0.7105
Epoch 166/200
2/2 [==============================] - 0s 199ms/step - loss: 1.9471 - accuracy: 0.6992 - val_loss: 
1.8418 - val_accuracy: 0.7140
Epoch 167/200
2/2 [==============================] - 0s 199ms/step - loss: 1.9441 - accuracy: 0.6995 - val_loss: 
1.8397 - val_accuracy: 0.7151
Epoch 168/200
2/2 [==============================] - 0s 195ms/step - loss: 1.9411 - accuracy: 0.7000 - val_loss: 
1.8378 - val_accuracy: 0.7151
Epoch 169/200
2/2 [==============================] - 0s 194ms/step - loss: 1.9380 - accuracy: 0.7002 - val_loss: 
1.8357 - val_accuracy: 0.7151
Epoch 170/200
2/2 [==============================] - 0s 195ms/step - loss: 1.9349 - accuracy: 0.7002 - val_loss: 
1.8337 - val_accuracy: 0.7163
Epoch 171/200
2/2 [==============================] - 0s 196ms/step - loss: 1.9318 - accuracy: 0.7002 - val_loss: 
1.8316 - val_accuracy: 0.7163
Epoch 172/200
```

```
2/2 [==============================] - 0s 196ms/step - loss: 1.9286 - accuracy: 0.7006 - val_loss: 1.8295 - val_accuracy: 0.7174
Epoch 173/200
2/2 [==============================] - 0s 195ms/step - loss: 1.9256 - accuracy: 0.7010 - val_loss: 1.8275 - val_accuracy: 0.7174
Epoch 174/200
2/2 [==============================] - 0s 197ms/step - loss: 1.9223 - accuracy: 0.7009 - val_loss: 1.8256 - val_accuracy: 0.7174
Epoch 175/200
2/2 [==============================] - 0s 201ms/step - loss: 1.9192 - accuracy: 0.7010 - val_loss: 1.8233 - val_accuracy: 0.7174
Epoch 176/200
2/2 [==============================] - 0s 199ms/step - loss: 1.9159 - accuracy: 0.7012 - val_loss: 1.8212 - val_accuracy: 0.7174
Epoch 177/200
2/2 [==============================] - 0s 195ms/step - loss: 1.9127 - accuracy: 0.7014 - val_loss: 1.8191 - val_accuracy: 0.7174
Epoch 178/200
2/2 [==============================] - 0s 196ms/step - loss: 1.9095 - accuracy: 0.7015 - val_loss: 1.8171 - val_accuracy: 0.7174
Epoch 179/200
2/2 [==============================] - 0s 198ms/step - loss: 1.9063 - accuracy: 0.7017 - val_loss: 1.8148 - val_accuracy: 0.7186
Epoch 180/200
2/2 [==============================] - 0s 197ms/step - loss: 1.9030 - accuracy: 0.7027 - val_loss: 1.8127 - val_accuracy: 0.7186
Epoch 181/200
2/2 [==============================] - 0s 198ms/step - loss: 1.8997 - accuracy: 0.7028 - val_loss: 1.8106 - val_accuracy: 0.7186
Epoch 182/200
2/2 [==============================] - 0s 198ms/step - loss: 1.8963 - accuracy: 0.7030 - val_loss: 1.8084 - val_accuracy: 0.7174
Epoch 183/200
2/2 [==============================] - 0s 198ms/step - loss: 1.8930 - accuracy: 0.7032 - val_loss: 1.8065 - val_accuracy: 0.7174
Epoch 184/200
2/2 [==============================] - 0s 195ms/step - loss: 1.8897 - accuracy: 0.7034 - val_loss: 1.8044 - val_accuracy: 0.7174
Epoch 185/200
2/2 [==============================] - 0s 195ms/step - loss: 1.8863 - accuracy: 0.7036 - val_loss: 1.8022 - val_accuracy: 0.7174
Epoch 186/200
2/2 [==============================] - 0s 198ms/step - loss: 1.8829 - accuracy: 0.7038 - val_loss: 1.8002 - val_accuracy: 0.7174
Epoch 187/200
2/2 [==============================] - 0s 196ms/step - loss: 1.8795 - accuracy: 0.7040 - val_loss: 1.7982 - val_accuracy: 0.7186
Epoch 188/200
2/2 [==============================] - 0s 196ms/step - loss: 1.8762 - accuracy: 0.7044 - val_loss: 1.7960 - val_accuracy: 0.7186
Epoch 189/200
2/2 [==============================] - 0s 196ms/step - loss: 1.8727 - accuracy: 0.7050 - val_loss: 1.7938 - val_accuracy: 0.7198
Epoch 190/200
2/2 [==============================] - 0s 196ms/step - loss: 1.8693 - accuracy: 0.7054 - val_loss: 1.7915 - val_accuracy: 0.7198
Epoch 191/200
2/2 [==============================] - 0s 201ms/step - loss: 1.8658 - accuracy: 0.7058 - val_loss: 1.7892 - val_accuracy: 0.7198
Epoch 192/200
2/2 [==============================] - 0s 200ms/step - loss: 1.8623 - accuracy: 0.7061 - val_loss: 1.7872 - val_accuracy: 0.7209
Epoch 193/200
2/2 [==============================] - 0s 198ms/step - loss: 1.8588 - accuracy: 0.7066 - val_loss: 1.7849 - val_accuracy: 0.7221
Epoch 194/200
2/2 [==============================] - 0s 196ms/step - loss: 1.8553 - accuracy: 0.7071 - val_loss: 1.7825 - val_accuracy: 0.7244
Epoch 195/200
2/2 [==============================] - 0s 196ms/step - loss: 1.8518 - accuracy: 0.7075 - val_loss: 1.7805 - val_accuracy: 0.7244
Epoch 196/200
2/2 [==============================] - 0s 199ms/step - loss: 1.8483 - accuracy: 0.7078 - val_loss: 1.7781 - val_accuracy: 0.7244
Epoch 197/200
2/2 [==============================] - 0s 194ms/step - loss: 1.8446 - accuracy: 0.7081 - val_loss: 1.7756 - val_accuracy: 0.7244
```

```
Epoch 198/200
2/2 [==============================] - 0s 198ms/step - loss: 1.8411 - accuracy: 0.7085 - val_loss:
1.7739 - val_accuracy: 0.7256
Epoch 199/200
2/2 [==============================] - 0s 196ms/step - loss: 1.8375 - accuracy: 0.7091 - val_loss:
1.7712 - val_accuracy: 0.7279
Epoch 200/200
2/2 [==============================] - 0s 197ms/step - loss: 1.8339 - accuracy: 0.7097 - val_loss:
1.7690 - val_accuracy: 0.7279
```

Out[75]:

```
<tensorflow.python.keras.callbacks.History at 0x7f1f17cd8350>
```

In [76]:

```
model.fit(source_padded_docs_train,target_padded_docs_train,batch_size=1024,epochs=100,
          validation_data=(source_padded_docs_test,target_padded_docs_test))
```

```
Epoch 1/100
2/2 [==============================] - 0s 214ms/step - loss: 1.8304 - accuracy: 0.7102 - val_loss:
1.7669 - val_accuracy: 0.7279
Epoch 2/100
2/2 [==============================] - 0s 189ms/step - loss: 1.8268 - accuracy: 0.7111 - val_loss:
1.7647 - val_accuracy: 0.7291
Epoch 3/100
2/2 [==============================] - 0s 192ms/step - loss: 1.8231 - accuracy: 0.7115 - val_loss:
1.7634 - val_accuracy: 0.7302
Epoch 4/100
2/2 [==============================] - 0s 193ms/step - loss: 1.8195 - accuracy: 0.7118 - val_loss:
1.7596 - val_accuracy: 0.7291
Epoch 5/100
2/2 [==============================] - 0s 196ms/step - loss: 1.8160 - accuracy: 0.7124 - val_loss:
1.7578 - val_accuracy: 0.7302
Epoch 6/100
2/2 [==============================] - 0s 195ms/step - loss: 1.8122 - accuracy: 0.7126 - val_loss:
1.7559 - val_accuracy: 0.7302
Epoch 7/100
2/2 [==============================] - 0s 193ms/step - loss: 1.8086 - accuracy: 0.7131 - val_loss:
1.7531 - val_accuracy: 0.7291
Epoch 8/100
2/2 [==============================] - 0s 191ms/step - loss: 1.8049 - accuracy: 0.7136 - val_loss:
1.7518 - val_accuracy: 0.7291
Epoch 9/100
2/2 [==============================] - 0s 194ms/step - loss: 1.8013 - accuracy: 0.7138 - val_loss:
1.7487 - val_accuracy: 0.7291
Epoch 10/100
2/2 [==============================] - 0s 197ms/step - loss: 1.7976 - accuracy: 0.7145 - val_loss:
1.7469 - val_accuracy: 0.7314
Epoch 11/100
2/2 [==============================] - 0s 192ms/step - loss: 1.7939 - accuracy: 0.7147 - val_loss:
1.7451 - val_accuracy: 0.7314
Epoch 12/100
2/2 [==============================] - 0s 194ms/step - loss: 1.7902 - accuracy: 0.7151 - val_loss:
1.7423 - val_accuracy: 0.7314
Epoch 13/100
2/2 [==============================] - 0s 194ms/step - loss: 1.7866 - accuracy: 0.7154 - val_loss:
1.7410 - val_accuracy: 0.7326
Epoch 14/100
2/2 [==============================] - 0s 194ms/step - loss: 1.7830 - accuracy: 0.7156 - val_loss:
1.7375 - val_accuracy: 0.7337
Epoch 15/100
2/2 [==============================] - 0s 193ms/step - loss: 1.7793 - accuracy: 0.7166 - val_loss:
1.7360 - val_accuracy: 0.7337
Epoch 16/100
2/2 [==============================] - 0s 198ms/step - loss: 1.7758 - accuracy: 0.7163 - val_loss:
1.7335 - val_accuracy: 0.7337
Epoch 17/100
2/2 [==============================] - 0s 193ms/step - loss: 1.7721 - accuracy: 0.7173 - val_loss:
1.7308 - val_accuracy: 0.7337
Epoch 18/100
2/2 [==============================] - 0s 192ms/step - loss: 1.7681 - accuracy: 0.7176 - val_loss:
1.7307 - val_accuracy: 0.7337
Epoch 19/100
2/2 [==============================] - 0s 192ms/step - loss: 1.7645 - accuracy: 0.7179 - val_loss:
1.7263 - val_accuracy: 0.7337
Epoch 20/100
2/2 [==============================] - 0s 194ms/step - loss: 1.7609 - accuracy: 0.7192 - val_loss:
1.7258 - val_accuracy: 0.7337
Epoch 21/100
2/2 [==============================] - 0s 197ms/step - loss: 1.7570 - accuracy: 0.7193 - val_loss:
```

```
1.7234 - val_accuracy: 0.7337
Epoch 22/100
2/2 [==============================] - 0s 195ms/step - loss: 1.7531 - accuracy: 0.7197 - val_loss:
1.7210 - val_accuracy: 0.7337
Epoch 23/100
2/2 [==============================] - 0s 194ms/step - loss: 1.7494 - accuracy: 0.7200 - val_loss:
1.7203 - val_accuracy: 0.7337
Epoch 24/100
2/2 [==============================] - 0s 197ms/step - loss: 1.7457 - accuracy: 0.7203 - val_loss:
1.7171 - val_accuracy: 0.7337
Epoch 25/100
2/2 [==============================] - 0s 196ms/step - loss: 1.7419 - accuracy: 0.7208 - val_loss:
1.7159 - val_accuracy: 0.7337
Epoch 26/100
2/2 [==============================] - 0s 196ms/step - loss: 1.7381 - accuracy: 0.7210 - val_loss:
1.7133 - val_accuracy: 0.7337
Epoch 27/100
2/2 [==============================] - 0s 196ms/step - loss: 1.7344 - accuracy: 0.7218 - val_loss:
1.7108 - val_accuracy: 0.7337
Epoch 28/100
2/2 [==============================] - 0s 199ms/step - loss: 1.7306 - accuracy: 0.7221 - val_loss:
1.7090 - val_accuracy: 0.7337
Epoch 29/100
2/2 [==============================] - 0s 204ms/step - loss: 1.7268 - accuracy: 0.7224 - val_loss:
1.7060 - val_accuracy: 0.7337
Epoch 30/100
2/2 [==============================] - 0s 198ms/step - loss: 1.7231 - accuracy: 0.7229 - val_loss:
1.7045 - val_accuracy: 0.7337
Epoch 31/100
2/2 [==============================] - 0s 196ms/step - loss: 1.7192 - accuracy: 0.7233 - val_loss:
1.7022 - val_accuracy: 0.7337
Epoch 32/100
2/2 [==============================] - 0s 198ms/step - loss: 1.7155 - accuracy: 0.7236 - val_loss:
1.6999 - val_accuracy: 0.7337
Epoch 33/100
2/2 [==============================] - 0s 195ms/step - loss: 1.7117 - accuracy: 0.7242 - val_loss:
1.6987 - val_accuracy: 0.7349
Epoch 34/100
2/2 [==============================] - 0s 195ms/step - loss: 1.7080 - accuracy: 0.7243 - val_loss:
1.6956 - val_accuracy: 0.7349
Epoch 35/100
2/2 [==============================] - 0s 199ms/step - loss: 1.7045 - accuracy: 0.7250 - val_loss:
1.6961 - val_accuracy: 0.7349
Epoch 36/100
2/2 [==============================] - 0s 198ms/step - loss: 1.7007 - accuracy: 0.7249 - val_loss:
1.6918 - val_accuracy: 0.7349
Epoch 37/100
2/2 [==============================] - 0s 199ms/step - loss: 1.6971 - accuracy: 0.7259 - val_loss:
1.6905 - val_accuracy: 0.7349
Epoch 38/100
2/2 [==============================] - 0s 198ms/step - loss: 1.6932 - accuracy: 0.7260 - val_loss:
1.6878 - val_accuracy: 0.7349
Epoch 39/100
2/2 [==============================] - 0s 196ms/step - loss: 1.6893 - accuracy: 0.7269 - val_loss:
1.6860 - val_accuracy: 0.7349
Epoch 40/100
2/2 [==============================] - 0s 195ms/step - loss: 1.6853 - accuracy: 0.7270 - val_loss:
1.6862 - val_accuracy: 0.7360
Epoch 41/100
2/2 [==============================] - 0s 198ms/step - loss: 1.6816 - accuracy: 0.7273 - val_loss:
1.6819 - val_accuracy: 0.7360
Epoch 42/100
2/2 [==============================] - 0s 197ms/step - loss: 1.6781 - accuracy: 0.7278 - val_loss:
1.6824 - val_accuracy: 0.7360
Epoch 43/100
2/2 [==============================] - 0s 199ms/step - loss: 1.6740 - accuracy: 0.7281 - val_loss:
1.6785 - val_accuracy: 0.7360
Epoch 44/100
2/2 [==============================] - 0s 198ms/step - loss: 1.6702 - accuracy: 0.7289 - val_loss:
1.6779 - val_accuracy: 0.7360
Epoch 45/100
2/2 [==============================] - 0s 196ms/step - loss: 1.6664 - accuracy: 0.7292 - val_loss:
1.6759 - val_accuracy: 0.7360
Epoch 46/100
2/2 [==============================] - 0s 197ms/step - loss: 1.6626 - accuracy: 0.7298 - val_loss:
1.6739 - val_accuracy: 0.7360
Epoch 47/100
```

```
2/2 [==============================] - 0s 196ms/step - loss: 1.6588 - accuracy: 0.7303 - val_loss:
1.6724 - val_accuracy: 0.7349
Epoch 48/100
2/2 [==============================] - 0s 196ms/step - loss: 1.6551 - accuracy: 0.7308 - val_loss:
1.6695 - val_accuracy: 0.7372
Epoch 49/100
2/2 [==============================] - 0s 199ms/step - loss: 1.6513 - accuracy: 0.7317 - val_loss:
1.6688 - val_accuracy: 0.7372
Epoch 50/100
2/2 [==============================] - 0s 198ms/step - loss: 1.6477 - accuracy: 0.7324 - val_loss:
1.6665 - val_accuracy: 0.7372
Epoch 51/100
2/2 [==============================] - 0s 199ms/step - loss: 1.6438 - accuracy: 0.7330 - val_loss:
1.6644 - val_accuracy: 0.7384
Epoch 52/100
2/2 [==============================] - 0s 197ms/step - loss: 1.6400 - accuracy: 0.7340 - val_loss:
1.6624 - val_accuracy: 0.7384
Epoch 53/100
2/2 [==============================] - 0s 196ms/step - loss: 1.6362 - accuracy: 0.7345 - val_loss:
1.6617 - val_accuracy: 0.7407
Epoch 54/100
2/2 [==============================] - 0s 194ms/step - loss: 1.6326 - accuracy: 0.7350 - val_loss:
1.6572 - val_accuracy: 0.7407
Epoch 55/100
2/2 [==============================] - 0s 205ms/step - loss: 1.6292 - accuracy: 0.7364 - val_loss:
1.6594 - val_accuracy: 0.7407
Epoch 56/100
2/2 [==============================] - 0s 195ms/step - loss: 1.6257 - accuracy: 0.7361 - val_loss:
1.6531 - val_accuracy: 0.7419
Epoch 57/100
2/2 [==============================] - 0s 199ms/step - loss: 1.6224 - accuracy: 0.7380 - val_loss:
1.6538 - val_accuracy: 0.7419
Epoch 58/100
2/2 [==============================] - 0s 206ms/step - loss: 1.6181 - accuracy: 0.7376 - val_loss:
1.6501 - val_accuracy: 0.7430
Epoch 59/100
2/2 [==============================] - 0s 199ms/step - loss: 1.6142 - accuracy: 0.7390 - val_loss:
1.6488 - val_accuracy: 0.7430
Epoch 60/100
2/2 [==============================] - 0s 197ms/step - loss: 1.6105 - accuracy: 0.7393 - val_loss:
1.6495 - val_accuracy: 0.7430
Epoch 61/100
2/2 [==============================] - 0s 196ms/step - loss: 1.6068 - accuracy: 0.7404 - val_loss:
1.6460 - val_accuracy: 0.7430
Epoch 62/100
2/2 [==============================] - 0s 197ms/step - loss: 1.6031 - accuracy: 0.7408 - val_loss:
1.6460 - val_accuracy: 0.7430
Epoch 63/100
2/2 [==============================] - 0s 199ms/step - loss: 1.5992 - accuracy: 0.7417 - val_loss:
1.6424 - val_accuracy: 0.7419
Epoch 64/100
2/2 [==============================] - 0s 199ms/step - loss: 1.5954 - accuracy: 0.7423 - val_loss:
1.6440 - val_accuracy: 0.7430
Epoch 65/100
2/2 [==============================] - 0s 202ms/step - loss: 1.5916 - accuracy: 0.7429 - val_loss:
1.6385 - val_accuracy: 0.7407
Epoch 66/100
2/2 [==============================] - 0s 202ms/step - loss: 1.5883 - accuracy: 0.7440 - val_loss:
1.6411 - val_accuracy: 0.7407
Epoch 67/100
2/2 [==============================] - 0s 206ms/step - loss: 1.5845 - accuracy: 0.7440 - val_loss:
1.6355 - val_accuracy: 0.7407
Epoch 68/100
2/2 [==============================] - 0s 202ms/step - loss: 1.5807 - accuracy: 0.7450 - val_loss:
1.6353 - val_accuracy: 0.7419
Epoch 69/100
2/2 [==============================] - 0s 203ms/step - loss: 1.5769 - accuracy: 0.7454 - val_loss:
1.6328 - val_accuracy: 0.7419
Epoch 70/100
2/2 [==============================] - 0s 201ms/step - loss: 1.5732 - accuracy: 0.7462 - val_loss:
1.6317 - val_accuracy: 0.7419
Epoch 71/100
2/2 [==============================] - 0s 206ms/step - loss: 1.5696 - accuracy: 0.7466 - val_loss:
1.6309 - val_accuracy: 0.7407
Epoch 72/100
2/2 [==============================] - 0s 198ms/step - loss: 1.5659 - accuracy: 0.7472 - val_loss:
1.6288 - val_accuracy: 0.7395
```

```
Epoch 73/100
2/2 [==============================] - 0s 196ms/step - loss: 1.5623 - accuracy: 0.7477 - val_loss:
1.6275 - val_accuracy: 0.7384
Epoch 74/100
2/2 [==============================] - 0s 205ms/step - loss: 1.5586 - accuracy: 0.7484 - val_loss:
1.6256 - val_accuracy: 0.7384
Epoch 75/100
2/2 [==============================] - 0s 196ms/step - loss: 1.5551 - accuracy: 0.7490 - val_loss:
1.6245 - val_accuracy: 0.7384
Epoch 76/100
2/2 [==============================] - 0s 196ms/step - loss: 1.5516 - accuracy: 0.7497 - val_loss:
1.6239 - val_accuracy: 0.7384
Epoch 77/100
2/2 [==============================] - 0s 197ms/step - loss: 1.5479 - accuracy: 0.7501 - val_loss:
1.6211 - val_accuracy: 0.7384
Epoch 78/100
2/2 [==============================] - 0s 198ms/step - loss: 1.5442 - accuracy: 0.7506 - val_loss:
1.6187 - val_accuracy: 0.7384
Epoch 79/100
2/2 [==============================] - 0s 198ms/step - loss: 1.5407 - accuracy: 0.7511 - val_loss:
1.6179 - val_accuracy: 0.7384
Epoch 80/100
2/2 [==============================] - 0s 198ms/step - loss: 1.5371 - accuracy: 0.7515 - val_loss:
1.6162 - val_accuracy: 0.7384
Epoch 81/100
2/2 [==============================] - 0s 198ms/step - loss: 1.5335 - accuracy: 0.7520 - val_loss:
1.6158 - val_accuracy: 0.7395
Epoch 82/100
2/2 [==============================] - 0s 198ms/step - loss: 1.5299 - accuracy: 0.7526 - val_loss:
1.6135 - val_accuracy: 0.7395
Epoch 83/100
2/2 [==============================] - 0s 199ms/step - loss: 1.5264 - accuracy: 0.7532 - val_loss:
1.6121 - val_accuracy: 0.7395
Epoch 84/100
2/2 [==============================] - 0s 197ms/step - loss: 1.5228 - accuracy: 0.7539 - val_loss:
1.6114 - val_accuracy: 0.7395
Epoch 85/100
2/2 [==============================] - 0s 197ms/step - loss: 1.5193 - accuracy: 0.7543 - val_loss:
1.6086 - val_accuracy: 0.7407
Epoch 86/100
2/2 [==============================] - 0s 194ms/step - loss: 1.5159 - accuracy: 0.7551 - val_loss:
1.6075 - val_accuracy: 0.7407
Epoch 87/100
2/2 [==============================] - 0s 198ms/step - loss: 1.5123 - accuracy: 0.7559 - val_loss:
1.6083 - val_accuracy: 0.7395
Epoch 88/100
2/2 [==============================] - 0s 196ms/step - loss: 1.5087 - accuracy: 0.7561 - val_loss:
1.6036 - val_accuracy: 0.7419
Epoch 89/100
2/2 [==============================] - 0s 198ms/step - loss: 1.5054 - accuracy: 0.7569 - val_loss:
1.6051 - val_accuracy: 0.7419
Epoch 90/100
2/2 [==============================] - 0s 197ms/step - loss: 1.5019 - accuracy: 0.7574 - val_loss:
1.6015 - val_accuracy: 0.7430
Epoch 91/100
2/2 [==============================] - 0s 199ms/step - loss: 1.4982 - accuracy: 0.7580 - val_loss:
1.6025 - val_accuracy: 0.7419
Epoch 92/100
2/2 [==============================] - 0s 197ms/step - loss: 1.4949 - accuracy: 0.7586 - val_loss:
1.6001 - val_accuracy: 0.7442
Epoch 93/100
2/2 [==============================] - 0s 198ms/step - loss: 1.4918 - accuracy: 0.7589 - val_loss:
1.5958 - val_accuracy: 0.7442
Epoch 94/100
2/2 [==============================] - 0s 197ms/step - loss: 1.4886 - accuracy: 0.7600 - val_loss:
1.6021 - val_accuracy: 0.7430
Epoch 95/100
2/2 [==============================] - 0s 199ms/step - loss: 1.4853 - accuracy: 0.7599 - val_loss:
1.5931 - val_accuracy: 0.7442
Epoch 96/100
2/2 [==============================] - 0s 196ms/step - loss: 1.4816 - accuracy: 0.7610 - val_loss:
1.5987 - val_accuracy: 0.7430
Epoch 97/100
2/2 [==============================] - 0s 200ms/step - loss: 1.4782 - accuracy: 0.7607 - val_loss:
1.5903 - val_accuracy: 0.7442
Epoch 98/100
2/2 [==============================] - 0s 195ms/step - loss: 1.4748 - accuracy: 0.7620 - val_loss:
```

```
1.5954 - val_accuracy: 0.7430
Epoch 99/100
2/2 [==============================] - 0s 195ms/step - loss: 1.4712 - accuracy: 0.7622 - val_loss:
1.5893 - val_accuracy: 0.7465
Epoch 100/100
2/2 [==============================] - 0s 198ms/step - loss: 1.4677 - accuracy: 0.7629 - val_loss:
1.5939 - val_accuracy: 0.7442
```

Out[76]:

```
<tensorflow.python.keras.callbacks.History at 0x7f1f17610b10>
```

In [77]:

```
model.fit(source_padded_docs_train,target_padded_docs_train,batch_size=1024,epochs=100,
          validation_data=(source_padded_docs_test,target_padded_docs_test))
```

```
Epoch 1/100
2/2 [==============================] - 0s 214ms/step - loss: 1.4645 - accuracy: 0.7633 - val_loss:
1.5882 - val_accuracy: 0.7477
Epoch 2/100
2/2 [==============================] - 0s 196ms/step - loss: 1.4612 - accuracy: 0.7640 - val_loss:
1.5913 - val_accuracy: 0.7465
Epoch 3/100
2/2 [==============================] - 0s 192ms/step - loss: 1.4581 - accuracy: 0.7641 - val_loss:
1.5854 - val_accuracy: 0.7477
Epoch 4/100
2/2 [==============================] - 0s 195ms/step - loss: 1.4548 - accuracy: 0.7647 - val_loss:
1.5852 - val_accuracy: 0.7477
Epoch 5/100
2/2 [==============================] - 0s 196ms/step - loss: 1.4512 - accuracy: 0.7654 - val_loss:
1.5841 - val_accuracy: 0.7465
Epoch 6/100
2/2 [==============================] - 0s 194ms/step - loss: 1.4478 - accuracy: 0.7656 - val_loss:
1.5831 - val_accuracy: 0.7465
Epoch 7/100
2/2 [==============================] - 0s 192ms/step - loss: 1.4444 - accuracy: 0.7663 - val_loss:
1.5823 - val_accuracy: 0.7465
Epoch 8/100
2/2 [==============================] - 0s 192ms/step - loss: 1.4411 - accuracy: 0.7666 - val_loss:
1.5821 - val_accuracy: 0.7465
Epoch 9/100
2/2 [==============================] - 0s 193ms/step - loss: 1.4379 - accuracy: 0.7671 - val_loss:
1.5812 - val_accuracy: 0.7465
Epoch 10/100
2/2 [==============================] - 0s 195ms/step - loss: 1.4345 - accuracy: 0.7676 - val_loss:
1.5793 - val_accuracy: 0.7465
Epoch 11/100
2/2 [==============================] - 0s 196ms/step - loss: 1.4313 - accuracy: 0.7683 - val_loss:
1.5771 - val_accuracy: 0.7477
Epoch 12/100
2/2 [==============================] - 0s 193ms/step - loss: 1.4281 - accuracy: 0.7688 - val_loss:
1.5773 - val_accuracy: 0.7453
Epoch 13/100
2/2 [==============================] - 0s 194ms/step - loss: 1.4249 - accuracy: 0.7694 - val_loss:
1.5760 - val_accuracy: 0.7477
Epoch 14/100
2/2 [==============================] - 0s 194ms/step - loss: 1.4217 - accuracy: 0.7700 - val_loss:
1.5746 - val_accuracy: 0.7465
Epoch 15/100
2/2 [==============================] - 0s 194ms/step - loss: 1.4185 - accuracy: 0.7707 - val_loss:
1.5751 - val_accuracy: 0.7453
Epoch 16/100
2/2 [==============================] - 0s 195ms/step - loss: 1.4153 - accuracy: 0.7711 - val_loss:
1.5717 - val_accuracy: 0.7465
Epoch 17/100
2/2 [==============================] - 0s 193ms/step - loss: 1.4121 - accuracy: 0.7716 - val_loss:
1.5747 - val_accuracy: 0.7453
Epoch 18/100
2/2 [==============================] - 0s 192ms/step - loss: 1.4092 - accuracy: 0.7720 - val_loss:
1.5715 - val_accuracy: 0.7465
Epoch 19/100
2/2 [==============================] - 0s 219ms/step - loss: 1.4059 - accuracy: 0.7726 - val_loss:
1.5690 - val_accuracy: 0.7488
Epoch 20/100
2/2 [==============================] - 0s 193ms/step - loss: 1.4028 - accuracy: 0.7732 - val_loss:
1.5713 - val_accuracy: 0.7477
Epoch 21/100
2/2 [==============================] - 0s 194ms/step - loss: 1.3999 - accuracy: 0.7735 - val_loss:
1.5669 - val_accuracy: 0.7500
Epoch 22/100
```

```
Epoch 22/100
2/2 [==============================] - 0s 193ms/step - loss: 1.3966 - accuracy: 0.7739 - val_loss:
1.5680 - val_accuracy: 0.7500
Epoch 23/100
2/2 [==============================] - 0s 194ms/step - loss: 1.3934 - accuracy: 0.7744 - val_loss:
1.5681 - val_accuracy: 0.7488
Epoch 24/100
2/2 [==============================] - 0s 195ms/step - loss: 1.3904 - accuracy: 0.7746 - val_loss:
1.5638 - val_accuracy: 0.7523
Epoch 25/100
2/2 [==============================] - 0s 195ms/step - loss: 1.3874 - accuracy: 0.7754 - val_loss:
1.5682 - val_accuracy: 0.7512
Epoch 26/100
2/2 [==============================] - 0s 195ms/step - loss: 1.3856 - accuracy: 0.7758 - val_loss:
1.5636 - val_accuracy: 0.7523
Epoch 27/100
2/2 [==============================] - 0s 193ms/step - loss: 1.3823 - accuracy: 0.7760 - val_loss:
1.5555 - val_accuracy: 0.7547
Epoch 28/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3804 - accuracy: 0.7773 - val_loss:
1.5689 - val_accuracy: 0.7523
Epoch 29/100
2/2 [==============================] - 0s 195ms/step - loss: 1.3777 - accuracy: 0.7769 - val_loss:
1.5538 - val_accuracy: 0.7535
Epoch 30/100
2/2 [==============================] - 0s 198ms/step - loss: 1.3735 - accuracy: 0.7780 - val_loss:
1.5633 - val_accuracy: 0.7523
Epoch 31/100
2/2 [==============================] - 0s 193ms/step - loss: 1.3703 - accuracy: 0.7778 - val_loss:
1.5545 - val_accuracy: 0.7535
Epoch 32/100
2/2 [==============================] - 0s 195ms/step - loss: 1.3671 - accuracy: 0.7786 - val_loss:
1.5610 - val_accuracy: 0.7523
Epoch 33/100
2/2 [==============================] - 0s 194ms/step - loss: 1.3644 - accuracy: 0.7784 - val_loss:
1.5537 - val_accuracy: 0.7535
Epoch 34/100
2/2 [==============================] - 0s 197ms/step - loss: 1.3621 - accuracy: 0.7795 - val_loss:
1.5620 - val_accuracy: 0.7523
Epoch 35/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3588 - accuracy: 0.7790 - val_loss:
1.5528 - val_accuracy: 0.7535
Epoch 36/100
2/2 [==============================] - 0s 194ms/step - loss: 1.3558 - accuracy: 0.7802 - val_loss:
1.5567 - val_accuracy: 0.7523
Epoch 37/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3526 - accuracy: 0.7800 - val_loss:
1.5533 - val_accuracy: 0.7547
Epoch 38/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3495 - accuracy: 0.7809 - val_loss:
1.5525 - val_accuracy: 0.7558
Epoch 39/100
2/2 [==============================] - 0s 194ms/step - loss: 1.3469 - accuracy: 0.7807 - val_loss:
1.5530 - val_accuracy: 0.7547
Epoch 40/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3438 - accuracy: 0.7815 - val_loss:
1.5514 - val_accuracy: 0.7547
Epoch 41/100
2/2 [==============================] - 0s 200ms/step - loss: 1.3409 - accuracy: 0.7816 - val_loss:
1.5516 - val_accuracy: 0.7547
Epoch 42/100
2/2 [==============================] - 0s 194ms/step - loss: 1.3378 - accuracy: 0.7824 - val_loss:
1.5507 - val_accuracy: 0.7558
Epoch 43/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3350 - accuracy: 0.7825 - val_loss:
1.5532 - val_accuracy: 0.7558
Epoch 44/100
2/2 [==============================] - 0s 195ms/step - loss: 1.3323 - accuracy: 0.7828 - val_loss:
1.5499 - val_accuracy: 0.7558
Epoch 45/100
2/2 [==============================] - 0s 195ms/step - loss: 1.3295 - accuracy: 0.7831 - val_loss:
1.5513 - val_accuracy: 0.7558
Epoch 46/100
2/2 [==============================] - 0s 197ms/step - loss: 1.3266 - accuracy: 0.7836 - val_loss:
1.5490 - val_accuracy: 0.7570
Epoch 47/100
2/2 [==============================] - 0s 194ms/step - loss: 1.3239 - accuracy: 0.7841 - val_loss:
1.5481 - val_accuracy: 0.7570
```

```
1.5481 - val_accuracy: 0.7570
Epoch 48/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3212 - accuracy: 0.7845 - val_loss:
1.5483 - val_accuracy: 0.7593
Epoch 49/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3184 - accuracy: 0.7847 - val_loss:
1.5452 - val_accuracy: 0.7581
Epoch 50/100
2/2 [==============================] - 0s 194ms/step - loss: 1.3156 - accuracy: 0.7852 - val_loss:
1.5474 - val_accuracy: 0.7593
Epoch 51/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3130 - accuracy: 0.7855 - val_loss:
1.5430 - val_accuracy: 0.7605
Epoch 52/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3105 - accuracy: 0.7861 - val_loss:
1.5497 - val_accuracy: 0.7605
Epoch 53/100
2/2 [==============================] - 0s 202ms/step - loss: 1.3079 - accuracy: 0.7860 - val_loss:
1.5420 - val_accuracy: 0.7605
Epoch 54/100
2/2 [==============================] - 0s 196ms/step - loss: 1.3050 - accuracy: 0.7868 - val_loss:
1.5478 - val_accuracy: 0.7605
Epoch 55/100
2/2 [==============================] - 0s 194ms/step - loss: 1.3026 - accuracy: 0.7867 - val_loss:
1.5391 - val_accuracy: 0.7616
Epoch 56/100
2/2 [==============================] - 0s 200ms/step - loss: 1.3001 - accuracy: 0.7872 - val_loss:
1.5439 - val_accuracy: 0.7605
Epoch 57/100
2/2 [==============================] - 0s 196ms/step - loss: 1.2972 - accuracy: 0.7876 - val_loss:
1.5407 - val_accuracy: 0.7616
Epoch 58/100
2/2 [==============================] - 0s 197ms/step - loss: 1.2946 - accuracy: 0.7877 - val_loss:
1.5423 - val_accuracy: 0.7616
Epoch 59/100
2/2 [==============================] - 0s 197ms/step - loss: 1.2922 - accuracy: 0.7882 - val_loss:
1.5411 - val_accuracy: 0.7616
Epoch 60/100
2/2 [==============================] - 0s 200ms/step - loss: 1.2893 - accuracy: 0.7883 - val_loss:
1.5399 - val_accuracy: 0.7616
Epoch 61/100
2/2 [==============================] - 0s 198ms/step - loss: 1.2866 - accuracy: 0.7888 - val_loss:
1.5407 - val_accuracy: 0.7616
Epoch 62/100
2/2 [==============================] - 0s 201ms/step - loss: 1.2839 - accuracy: 0.7889 - val_loss:
1.5394 - val_accuracy: 0.7628
Epoch 63/100
2/2 [==============================] - 0s 192ms/step - loss: 1.2813 - accuracy: 0.7894 - val_loss:
1.5409 - val_accuracy: 0.7605
Epoch 64/100
2/2 [==============================] - 0s 195ms/step - loss: 1.2789 - accuracy: 0.7894 - val_loss:
1.5365 - val_accuracy: 0.7593
Epoch 65/100
2/2 [==============================] - 0s 195ms/step - loss: 1.2765 - accuracy: 0.7898 - val_loss:
1.5431 - val_accuracy: 0.7605
Epoch 66/100
2/2 [==============================] - 0s 193ms/step - loss: 1.2744 - accuracy: 0.7899 - val_loss:
1.5366 - val_accuracy: 0.7593
Epoch 67/100
2/2 [==============================] - 0s 196ms/step - loss: 1.2716 - accuracy: 0.7901 - val_loss:
1.5424 - val_accuracy: 0.7605
Epoch 68/100
2/2 [==============================] - 0s 197ms/step - loss: 1.2701 - accuracy: 0.7907 - val_loss:
1.5333 - val_accuracy: 0.7605
Epoch 69/100
2/2 [==============================] - 0s 195ms/step - loss: 1.2675 - accuracy: 0.7908 - val_loss:
1.5381 - val_accuracy: 0.7605
Epoch 70/100
2/2 [==============================] - 0s 199ms/step - loss: 1.2644 - accuracy: 0.7912 - val_loss:
1.5326 - val_accuracy: 0.7616
Epoch 71/100
2/2 [==============================] - 0s 196ms/step - loss: 1.2620 - accuracy: 0.7911 - val_loss:
1.5353 - val_accuracy: 0.7616
Epoch 72/100
2/2 [==============================] - 0s 203ms/step - loss: 1.2594 - accuracy: 0.7918 - val_loss:
1.5310 - val_accuracy: 0.7605
Epoch 73/100
2/2 [==============================] - 0s 199ms/step - loss: 1.2569 - accuracy: 0.7916 - val_loss:
```

```
2/2 [==============================] - 0s 199ms/step - loss: 1.2569 - accuracy: 0.7916 - val_loss:
1.5333 - val_accuracy: 0.7605
Epoch 74/100
2/2 [==============================] - 0s 197ms/step - loss: 1.2541 - accuracy: 0.7919 - val_loss:
1.5337 - val_accuracy: 0.7605
Epoch 75/100
2/2 [==============================] - 0s 198ms/step - loss: 1.2515 - accuracy: 0.7922 - val_loss:
1.5362 - val_accuracy: 0.7605
Epoch 76/100
2/2 [==============================] - 0s 196ms/step - loss: 1.2490 - accuracy: 0.7924 - val_loss:
1.5334 - val_accuracy: 0.7605
Epoch 77/100
2/2 [==============================] - 0s 195ms/step - loss: 1.2466 - accuracy: 0.7925 - val_loss:
1.5351 - val_accuracy: 0.7593
Epoch 78/100
2/2 [==============================] - 0s 196ms/step - loss: 1.2441 - accuracy: 0.7928 - val_loss:
1.5319 - val_accuracy: 0.7605
Epoch 79/100
2/2 [==============================] - 0s 196ms/step - loss: 1.2419 - accuracy: 0.7931 - val_loss:
1.5345 - val_accuracy: 0.7605
Epoch 80/100
2/2 [==============================] - 0s 198ms/step - loss: 1.2394 - accuracy: 0.7934 - val_loss:
1.5320 - val_accuracy: 0.7616
Epoch 81/100
2/2 [==============================] - 0s 197ms/step - loss: 1.2370 - accuracy: 0.7935 - val_loss:
1.5325 - val_accuracy: 0.7605
Epoch 82/100
2/2 [==============================] - 0s 196ms/step - loss: 1.2347 - accuracy: 0.7938 - val_loss:
1.5321 - val_accuracy: 0.7616
Epoch 83/100
2/2 [==============================] - 0s 202ms/step - loss: 1.2324 - accuracy: 0.7941 - val_loss:
1.5300 - val_accuracy: 0.7628
Epoch 84/100
2/2 [==============================] - 0s 200ms/step - loss: 1.2300 - accuracy: 0.7945 - val_loss:
1.5331 - val_accuracy: 0.7605
Epoch 85/100
2/2 [==============================] - 0s 195ms/step - loss: 1.2277 - accuracy: 0.7944 - val_loss:
1.5278 - val_accuracy: 0.7640
Epoch 86/100
2/2 [==============================] - 0s 195ms/step - loss: 1.2254 - accuracy: 0.7947 - val_loss:
1.5322 - val_accuracy: 0.7628
Epoch 87/100
2/2 [==============================] - 0s 195ms/step - loss: 1.2233 - accuracy: 0.7949 - val_loss:
1.5290 - val_accuracy: 0.7651
Epoch 88/100
2/2 [==============================] - 0s 196ms/step - loss: 1.2212 - accuracy: 0.7950 - val_loss:
1.5264 - val_accuracy: 0.7674
Epoch 89/100
2/2 [==============================] - 0s 198ms/step - loss: 1.2189 - accuracy: 0.7954 - val_loss:
1.5342 - val_accuracy: 0.7651
Epoch 90/100
2/2 [==============================] - 0s 206ms/step - loss: 1.2170 - accuracy: 0.7954 - val_loss:
1.5258 - val_accuracy: 0.7663
Epoch 91/100
2/2 [==============================] - 0s 200ms/step - loss: 1.2141 - accuracy: 0.7958 - val_loss:
1.5293 - val_accuracy: 0.7651
Epoch 92/100
2/2 [==============================] - 0s 199ms/step - loss: 1.2119 - accuracy: 0.7962 - val_loss:
1.5256 - val_accuracy: 0.7651
Epoch 93/100
2/2 [==============================] - 0s 199ms/step - loss: 1.2095 - accuracy: 0.7961 - val_loss:
1.5277 - val_accuracy: 0.7651
Epoch 94/100
2/2 [==============================] - 0s 197ms/step - loss: 1.2074 - accuracy: 0.7964 - val_loss:
1.5279 - val_accuracy: 0.7663
Epoch 95/100
2/2 [==============================] - 0s 196ms/step - loss: 1.2049 - accuracy: 0.7965 - val_loss:
1.5257 - val_accuracy: 0.7663
Epoch 96/100
2/2 [==============================] - 0s 197ms/step - loss: 1.2026 - accuracy: 0.7969 - val_loss:
1.5290 - val_accuracy: 0.7651
Epoch 97/100
2/2 [==============================] - 0s 198ms/step - loss: 1.2004 - accuracy: 0.7970 - val_loss:
1.5253 - val_accuracy: 0.7663
Epoch 98/100
2/2 [==============================] - 0s 197ms/step - loss: 1.1981 - accuracy: 0.7972 - val_loss:
1.5282 - val_accuracy: 0.7663
Epoch 99/100
```

```
Epoch 99/100
2/2 [==============================] - 0s 198ms/step - loss: 1.1960 - accuracy: 0.7975 - val_loss: 1.5256 - val_accuracy: 0.7663
Epoch 100/100
2/2 [==============================] - 0s 197ms/step - loss: 1.1937 - accuracy: 0.7977 - val_loss: 1.5260 - val_accuracy: 0.7663
```

Out[77]:

```
<tensorflow.python.keras.callbacks.History at 0x7f1f175ed5d0>
```

In [78]:

```python
#https://machinelearningmastery.com/beam-search-decoder-natural-language-processing/
#Beam Search
from math import log
from numpy import array
from numpy import argmax
import numpy as np
def beam_search_decoder(data, k):
 sequences = [[list(), 0.0]]
 # walk over each step in sequence
 #print(sequences)
 for row in data:
  all_candidates = list()
  # expand each current candidate
  for i in range(len(sequences)):
   seq, score = sequences[i]
   for j in range(len(row)):
    candidate = [seq + [j], score - np.log(row[j])]
    all_candidates.append(candidate)
  # order all candidates by score
  ordered = sorted(all_candidates, key=lambda tup:tup[1])
  sequences = ordered[:k]
 return sequences
```

In [79]:

```python
#prediction
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 x])
  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 for beam==3 : ")
  x=model.predict(source_padded_docs_test[i:i+1])

  res=beam_search_decoder(x[0],3)

  y1=prediction(res[0][0])
  y1=y1.split(' ')
  y_lst1=[]
  for i in y1:
    if i=='<PAD>':
      continue
    else:
      y_lst1.append(i)
  print(' '.join(y_lst1))


  y2=prediction(res[1][0])
  y2=y2.split(' ')
  y_lst2=[]
  for i in y2:
    if i=='<PAD>':
      continue
    else:
      y_lst2.append(i)
  print(' '.join(y_lst2))
```

```
    y3=prediction(res[2][0])
    y3=y3.split(' ')
    y_lst3=[]
    for i in y3:
      if i=='<PAD>':
        continue
      else:
        y_lst3.append(i)
    print(' '.join(y_lst3))
    print('>'*180)
```

```
Input text:
Help me collect e clothes, goin to rain....
Actual Output:
Help me collect the clothes, going to rain.
Predicted Output for beam==3 :
help me to the and going to to
help me to the going to to
help me to the go going to to
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Mimi40 u now working or studying?
Actual Output:
Mimi40, are you now working or studying?
Predicted Output for beam==3 :
i'm you now working or studying
i'll you now working or studying
yes you now working or studying
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
1215 lar... What if i dont have a photo leh? Will they kill me?
Actual Output:
12:15. What if I don't have a photo? Will they kill me?
Predicted Output for beam==3 :
6 then what if i don't have a a will
6 ah what if i don't have a a will
6 i what if i don't have a a will
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
juz to make frnd wif u mah.if u wan u msg me at 99876452.
Actual Output:
Just to make friend with you. If you want, message me at 99876452.
Predicted Output for beam==3 :
just to make with you you if you want you message me at to
just to make with you you if you want you message me at
just to make with you you if you want you you me at to
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Hey i hear postings out online... Go check !
Actual Output:
Hey, I heard postings are out online. Go and check!
Predicted Output for beam==3 :
hey i i out to go out
hey i i out to go to
hey i i out for go out
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Gd morning,how is lifè today?Gd?Taken ur breakfast?
Actual Output:
Good morning, how is life today? Good? Taken your breakfast?
Predicted Output for beam==3 :
good morning how is today good your lunch
good morning how is today good quite your lunch
good morning how is today good your dinner
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
No lah... But borburn coke is one o e more popular drinks lor... So is li en dancing?
Actual Output:
No. But Borburn coke is one of the more popular drinks. So is Li En dancing?
Predicted Output for beam==3 :
no no but is one the more
no i but is one the more
```

```
no i but is one the more
no haha but is one the more
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Ok lor... I buy dinner for them now oredi...
Actual Output:
Ok. I buy dinner for them now already.
Predicted Output for beam==3 :
ok i i buy dinner for now already
ok ok i buy dinner for now already
ok i i buy dinner for for now already
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
K.reen u change ur number isit?hw cme neber sms 2 mi...so sad..:(
Actual Output:
Ok. Reen, you change your number, is it? How come you didn't SMS to me? So sad.
Predicted Output for beam==3 :
ok reen you you your number is how sms to sad
ok reen you you your number how sms to sad
ok reen you you your number is how sms for sad
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
SEN.ANYBODY THERE.
Actual Output:
Anybody there?
Predicted Output for beam==3 :
hi anyone there
i anyone there
no anyone there
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Haha yiyun? oh yes change no nvr tell me nt my fault arh starhub got contract min 3 months tink they mgh
t nt tk ü leh
Actual Output:
Haha, Yiyun? Oh yes. You change number but never tell me. Not my fault. Starhub got contract minimum 3 m
onths. Think they might not take you.
Predicted Output for beam==3 :
haha haha oh yes change no never say me you my i got for for i they i you you you
haha haha oh yes change no never say me i my i got for for i they i you you you
haha haha oh yes change no never say me not my i got for for i they i you you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Thanx u darlin!im cool thanx. A few bday drinks 2 nite. 2morrow off! Take care c u soon.
Actual Output:
Thank you darling! I am cool, thanks. A few birthday drinks tonight. Tomorrow off! Take care, see you so
on.
Predicted Output for beam==3 :
thanks you darling i'm thanks a a birthday night for night tomorrow off take care to you soon
thanks you darling i'm thanks a a birthday good for night tomorrow off take care to you soon
thanks you darling i'm thanks a a birthday birthday for night tomorrow off take care to you soon
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Ok ok just thought u want a lift. I may go down earlier too. Will call u... Need to find a good tailor i
n far east.
Actual Output:
Ok, just thought you want a lift. I may go down earlier too. Will call you. Need to find a good tailor i
n Far East.
Predicted Output for beam==3 :
ok ok just i you want a help i may go down later too will call you you to find a good in the
ok ok just i you want a a i may go down later too will call you you to find a good in the
ok ok just i you want a help i may go down before too will call you you to find a good in the
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Where are you and mother and yun
Actual Output:
Where are you and Mother and Yun?
Predicted Output for beam==3 :
where are you you yun
where are you you and yun
where are you you
```

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Okie... Den u'll reach ard wat time....
Actual Output:
Ok. Then around what time will you reach?
Predicted Output for beam==3 :
ok then reach around what time
okay then reach around what time
ok then then reach around what time
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Hey mel owes you money right? $5 remind me to pay you...
Actual Output:
Hey, Mel, owes you money right? $5, remind me to pay you.
Predicted Output for beam==3 :
hey you you right for you me to to you
hey you you right for you me to pay you
hey you you right for me to to you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
No probl... Maybe next time when u r free =5
Actual Output:
No problem. Maybe next time when you are free.
Predicted Output for beam==3 :
no maybe next time when you are free for
no maybe next time when you are free 2
no maybe next time when are are free for
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
How r u? Im slackg at home...Hows work so far?
Actual Output:
How are you? I'm slacking at home. How's your work so far?
Predicted Output for beam==3 :
how are you i'm at home how's work so far
how are you i at home how's work so far
how are you i'm at home work so far
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
No more stairs liao? Its bad for your knees so stoppin is quite good. Wat homework r u rushing? 1pm flig
ht? Ok... Mayb i go snatch josssticks... U know?
Actual Output:
No more stairs? It's bad for your knees, so stopping is quite good. What homework are you rushing? 1PM f
light? OK. Maybe I should go to snatch joss sticks. Do you know?
Predicted Output for beam==3 :
no more already it's bad for your so is quite good what are you ok maybe i go you you
no more already it bad for your so is quite good what are you ok maybe i go you you
no more already it's bad for your so is quite good what are you ok you i go you you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
yes, i noe,same here. but exams comin..hav to spend more time studyin...less time to meet up le :( u jus
started ah?thn go and study loh,i dun wanna disturb c:
Actual Output:
Yes, I know, same here. But exams are coming, have to spend more time studying, less time to meet up.
You just started? Then go and study, I don't want to disturb you.
Predicted Output for beam==3 :
yes i i same here but exams coming have to time more time studying more time to meet up the you just got
i i go go to to i don't want want see to
yes i i same here but exams coming have to time more time studying more time to meet up the you just got
i i go and to to i don't want want see to
yes i i same here but exams coming have to time more time studying more time to meet up the you just got
i i go to to to i don't want want see to
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

In [80]:

```python
import nltk.translate.bleu_score as bleu
bleu_score1=[]
bleu_score2=[]
bleu_score3=[]
#computing bleu_scores for 20 test points where beam=3
```

```
for i in range(20):
  b=list(y_test[i:i+1])
  x=model.predict(source_padded_docs_test[i:i+1])

  res=beam_search_decoder(x[0],3)

  y1=prediction(res[0][0])
  y1=y1.split(' ')
  y_lst1=[]
  for i in y1:
    if i=='<PAD>':
      continue
    else:
      y_lst1.append(i)
  bleu_score1.append(bleu.sentence_bleu([b[0].split(),],y_lst1))

  y2=prediction(res[1][0])
  y2=y2.split(' ')
  y_lst2=[]
  for i in y2:
    if i=='<PAD>':
      continue
    else:
      y_lst2.append(i)
  bleu_score2.append(bleu.sentence_bleu([b[0].split(),],y_lst2))

  y3=prediction(res[2][0])
  y3=y3.split(' ')
  y_lst3=[]
  for i in y3:
    if i=='<PAD>':
      continue
    else:
      y_lst3.append(i)
  bleu_score3.append(bleu.sentence_bleu([b[0].split(),],y_lst3))

print("The Average Bleu Score1 is: ",sum(bleu_score1)/20)
print('>'*180)
print("The Average Bleu Score2 is: ",sum(bleu_score2)/20)
print('>'*180)
print("The Average Bleu Score3 is: ",sum(bleu_score3)/20)
print('>'*180)
```

```
/usr/local/lib/python3.7/dist-packages/nltk/translate/bleu_score.py:490: UserWarning:
Corpus/Sentence contains 0 counts of 3-gram overlaps.
BLEU scores might be undesirable; use SmoothingFunction().
  warnings.warn(_msg)
/usr/local/lib/python3.7/dist-packages/nltk/translate/bleu_score.py:490: UserWarning:
Corpus/Sentence contains 0 counts of 4-gram overlaps.
BLEU scores might be undesirable; use SmoothingFunction().
  warnings.warn(_msg)
/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)
The Average Bleu Score1 is:  0.34713550129602855
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
The Average Bleu Score2 is:  0.3496921316896176
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
The Average Bleu Score3 is:  0.3447599549786778
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

Model2:

In [81]:

```
input=tf.keras.layers.Input(shape=(43,))
embed=tf.keras.layers.Embedding(source_vocab_size,300,weights=[embedding_matrix],input_length=source_padc
lstm1=tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(128, return_sequences=True))(embed)
output=tf.keras.layers.TimeDistributed(tf.keras.layers.Dense(target_vocab_size, activation='softmax'))(ls
model=tf.keras.models.Model(inputs=input,outputs=output)
model.summary()
```

```
Model: "model_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_6 (InputLayer)         [(None, 43)]              0         
_________________________________________________________________
embedding_5 (Embedding)      (None, 43, 300)           1109700   
_________________________________________________________________
bidirectional (Bidirectional (None, 43, 256)           439296    
_________________________________________________________________
time_distributed_5 (TimeDist (None, 43, 3034)          779738    
=================================================================
Total params: 2,328,734
Trainable params: 1,219,034
Non-trainable params: 1,109,700
_________________________________________________________________
```

In [82]:

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

In [83]:

```python
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 [==============================] - 3s 817ms/step - loss: 7.9019 - accuracy: 0.3301 - val_loss: 6.1246 - val_accuracy: 0.6895
Epoch 2/50
2/2 [==============================] - 0s 240ms/step - loss: 5.4224 - accuracy: 0.6821 - val_loss: 2.8513 - val_accuracy: 0.6837
Epoch 3/50
2/2 [==============================] - 0s 256ms/step - loss: 3.0681 - accuracy: 0.6737 - val_loss: 3.0906 - val_accuracy: 0.6837
Epoch 4/50
2/2 [==============================] - 0s 238ms/step - loss: 3.2525 - accuracy: 0.6737 - val_loss: 2.6719 - val_accuracy: 0.6837
Epoch 5/50
2/2 [==============================] - 0s 243ms/step - loss: 2.7828 - accuracy: 0.6752 - val_loss: 2.3982 - val_accuracy: 0.6907
Epoch 6/50
2/2 [==============================] - 0s 250ms/step - loss: 2.6079 - accuracy: 0.6804 - val_loss: 2.3878 - val_accuracy: 0.6942
Epoch 7/50
2/2 [==============================] - 0s 240ms/step - loss: 2.5251 - accuracy: 0.6820 - val_loss: 2.2288 - val_accuracy: 0.6930
Epoch 8/50
2/2 [==============================] - 0s 249ms/step - loss: 2.3942 - accuracy: 0.6797 - val_loss: 2.1736 - val_accuracy: 0.6907
Epoch 9/50
2/2 [==============================] - 0s 240ms/step - loss: 2.3324 - accuracy: 0.6793 - val_loss: 2.0823 - val_accuracy: 0.6930
Epoch 10/50
2/2 [==============================] - 0s 242ms/step - loss: 2.2250 - accuracy: 0.6826 - val_loss: 2.0031 - val_accuracy: 0.7012
Epoch 11/50
2/2 [==============================] - 0s 246ms/step - loss: 2.1373 - accuracy: 0.6878 - val_loss: 1.9494 - val_accuracy: 0.7058
Epoch 12/50
2/2 [==============================] - 0s 243ms/step - loss: 2.0773 - accuracy: 0.6901 - val_loss: 1.9092 - val_accuracy: 0.7081
Epoch 13/50
2/2 [==============================] - 0s 241ms/step - loss: 2.0361 - accuracy: 0.6921 - val_loss: 1.8817 - val_accuracy: 0.7058
Epoch 14/50
2/2 [==============================] - 0s 250ms/step - loss: 2.0039 - accuracy: 0.6936 - val_loss: 1.8604 - val_accuracy: 0.7093
Epoch 15/50
2/2 [==============================] - 0s 245ms/step - loss: 1.9748 - accuracy: 0.6966 - val_loss: 1.8433 - val_accuracy: 0.7128
Epoch 16/50
2/2 [==============================] - 0s 244ms/step - loss: 1.9505 - accuracy: 0.6981 - val_loss: 1.8256 - val_accuracy: 0.7128
Epoch 17/50
2/2 [==============================] - 0s 243ms/step - loss: 1.9290 - accuracy: 0.6996 - val_loss: 1.8128 - val_accuracy: 0.7140
Epoch 18/50
```

```
2/2 [==============================] - 0s 243ms/step - loss: 1.9091 - accuracy: 0.6999 - val_loss:
1.7979 - val_accuracy: 0.7174
Epoch 19/50
2/2 [==============================] - 0s 242ms/step - loss: 1.8866 - accuracy: 0.7009 - val_loss:
1.7839 - val_accuracy: 0.7221
Epoch 20/50
2/2 [==============================] - 0s 243ms/step - loss: 1.8649 - accuracy: 0.7034 - val_loss:
1.7697 - val_accuracy: 0.7221
Epoch 21/50
2/2 [==============================] - 0s 243ms/step - loss: 1.8410 - accuracy: 0.7068 - val_loss:
1.7543 - val_accuracy: 0.7256
Epoch 22/50
2/2 [==============================] - 0s 242ms/step - loss: 1.8155 - accuracy: 0.7092 - val_loss:
1.7408 - val_accuracy: 0.7279
Epoch 23/50
2/2 [==============================] - 0s 244ms/step - loss: 1.7897 - accuracy: 0.7122 - val_loss:
1.7264 - val_accuracy: 0.7302
Epoch 24/50
2/2 [==============================] - 0s 243ms/step - loss: 1.7613 - accuracy: 0.7150 - val_loss:
1.7097 - val_accuracy: 0.7349
Epoch 25/50
2/2 [==============================] - 0s 241ms/step - loss: 1.7312 - accuracy: 0.7183 - val_loss:
1.6928 - val_accuracy: 0.7372
Epoch 26/50
2/2 [==============================] - 0s 246ms/step - loss: 1.6988 - accuracy: 0.7208 - val_loss:
1.6743 - val_accuracy: 0.7384
Epoch 27/50
2/2 [==============================] - 0s 243ms/step - loss: 1.6647 - accuracy: 0.7243 - val_loss:
1.6568 - val_accuracy: 0.7407
Epoch 28/50
2/2 [==============================] - 0s 246ms/step - loss: 1.6301 - accuracy: 0.7289 - val_loss:
1.6349 - val_accuracy: 0.7419
Epoch 29/50
2/2 [==============================] - 0s 246ms/step - loss: 1.5940 - accuracy: 0.7338 - val_loss:
1.6166 - val_accuracy: 0.7442
Epoch 30/50
2/2 [==============================] - 0s 244ms/step - loss: 1.5580 - accuracy: 0.7392 - val_loss:
1.5949 - val_accuracy: 0.7453
Epoch 31/50
2/2 [==============================] - 0s 245ms/step - loss: 1.5211 - accuracy: 0.7436 - val_loss:
1.5796 - val_accuracy: 0.7500
Epoch 32/50
2/2 [==============================] - 0s 240ms/step - loss: 1.4830 - accuracy: 0.7483 - val_loss:
1.5642 - val_accuracy: 0.7547
Epoch 33/50
2/2 [==============================] - 0s 245ms/step - loss: 1.4452 - accuracy: 0.7545 - val_loss:
1.5524 - val_accuracy: 0.7535
Epoch 34/50
2/2 [==============================] - 0s 244ms/step - loss: 1.4081 - accuracy: 0.7614 - val_loss:
1.5361 - val_accuracy: 0.7593
Epoch 35/50
2/2 [==============================] - 0s 244ms/step - loss: 1.3718 - accuracy: 0.7669 - val_loss:
1.5273 - val_accuracy: 0.7605
Epoch 36/50
2/2 [==============================] - 0s 247ms/step - loss: 1.3348 - accuracy: 0.7720 - val_loss:
1.5197 - val_accuracy: 0.7605
Epoch 37/50
2/2 [==============================] - 0s 244ms/step - loss: 1.2995 - accuracy: 0.7774 - val_loss:
1.5049 - val_accuracy: 0.7605
Epoch 38/50
2/2 [==============================] - 0s 247ms/step - loss: 1.2646 - accuracy: 0.7822 - val_loss:
1.4991 - val_accuracy: 0.7616
Epoch 39/50
2/2 [==============================] - 0s 242ms/step - loss: 1.2311 - accuracy: 0.7866 - val_loss:
1.4901 - val_accuracy: 0.7628
Epoch 40/50
2/2 [==============================] - 0s 251ms/step - loss: 1.1984 - accuracy: 0.7908 - val_loss:
1.4832 - val_accuracy: 0.7616
Epoch 41/50
2/2 [==============================] - 0s 241ms/step - loss: 1.1667 - accuracy: 0.7957 - val_loss:
1.4771 - val_accuracy: 0.7628
Epoch 42/50
2/2 [==============================] - 0s 250ms/step - loss: 1.1370 - accuracy: 0.8004 - val_loss:
1.4728 - val_accuracy: 0.7651
Epoch 43/50
2/2 [==============================] - 0s 240ms/step - loss: 1.1084 - accuracy: 0.8042 - val_loss:
1.4689 - val_accuracy: 0.7674
Epoch 44/50
```

```
Epoch 44/50
2/2 [==============================] - 0s 244ms/step - loss: 1.0801 - accuracy: 0.8087 - val_loss:
1.4642 - val_accuracy: 0.7686
Epoch 45/50
2/2 [==============================] - 0s 243ms/step - loss: 1.0532 - accuracy: 0.8121 - val_loss:
1.4625 - val_accuracy: 0.7663
Epoch 46/50
2/2 [==============================] - 0s 244ms/step - loss: 1.0278 - accuracy: 0.8161 - val_loss:
1.4579 - val_accuracy: 0.7663
Epoch 47/50
2/2 [==============================] - 0s 245ms/step - loss: 1.0037 - accuracy: 0.8193 - val_loss:
1.4560 - val_accuracy: 0.7686
Epoch 48/50
2/2 [==============================] - 0s 248ms/step - loss: 0.9803 - accuracy: 0.8220 - val_loss:
1.4503 - val_accuracy: 0.7674
Epoch 49/50
2/2 [==============================] - 0s 247ms/step - loss: 0.9587 - accuracy: 0.8246 - val_loss:
1.4520 - val_accuracy: 0.7721
Epoch 50/50
2/2 [==============================] - 0s 242ms/step - loss: 0.9372 - accuracy: 0.8266 - val_loss:
1.4531 - val_accuracy: 0.7709
```

Out[83]:

```
<tensorflow.python.keras.callbacks.History at 0x7f1f19dafa90>
```

In [84]:

```
#https://machinelearningmastery.com/beam-search-decoder-natural-language-processing/
#Beam Score
from math import log
from numpy import array
from numpy import argmax
import numpy as np
def beam_search_decoder(data, k):
 sequences = [[list(), 0.0]]
 # walk over each step in sequence
 #print(sequences)
 for row in data:
  all_candidates = list()
  # expand each current candidate
  for i in range(len(sequences)):
   seq, score = sequences[i]
   for j in range(len(row)):
    candidate = [seq + [j], score - np.log(row[j])]
    all_candidates.append(candidate)
  # order all candidates by score
  ordered = sorted(all_candidates, key=lambda tup:tup[1])
  sequences = ordered[:k]
 return sequences
```

In [85]:

```
#prediction
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 x])
  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 for beam==3 : ")
  x=model.predict(source_padded_docs_test[i:i+1])

  res=beam_search_decoder(x[0],3)

  y1=prediction(res[0][0])
  y1=y1.split(' ')
  y_lst1=[]
  for i in y1:
    if i=='<PAD>':
      continue
```

```
    else:
      y_lst1.append(i)
  print(' '.join(y_lst1))


  y2=prediction(res[1][0])
  y2=y2.split(' ')
  y_lst2=[]
  for i in y2:
    if i=='<PAD>':
      continue
    else:
      y_lst2.append(i)
  print(' '.join(y_lst2))

  y3=prediction(res[2][0])
  y3=y3.split(' ')
  y_lst3=[]
  for i in y3:
    if i=='<PAD>':
      continue
    else:
      y_lst3.append(i)
  print(' '.join(y_lst3))
  print('>'*180)
```

```
Input text:
Help me collect e clothes, goin to rain....
Actual Output:
Help me collect the clothes, going to rain.
Predicted Output for beam==3 :
help me to the and going to
help me to the and going to rain
help me to the to going to
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Mimi40 u now working or studying?
Actual Output:
Mimi40, are you now working or studying?
Predicted Output for beam==3 :
i'm you now working or studying
yes you now working or studying
i'm are now working or studying
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
1215 lar... What if i dont have a photo leh? Will they kill me?
Actual Output:
12:15. What if I don't have a photo? Will they kill me?
Predicted Output for beam==3 :
yes ah what if i don't have a a don't will they you me
yes ah what if i don't have a a don't will they go me
yes ah what if i don't have a a don't will they don't me
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
juz to make frnd wif u mah.if u wan u msg me at 99876452.
Actual Output:
Just to make friend with you. If you want, message me at 99876452.
Predicted Output for beam==3 :
just to make with you you if you want you message me at
just to make with you you if you want you you me at
just to make with you you if you you you message me at
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Hey i hear postings out online... Go check !
Actual Output:
Hey, I heard postings are out online. Go and check!
Predicted Output for beam==3 :
hey i call going to go out
hey i call out to go out
hey i you going to go out
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Gd morning how is lifè today?Gd?Taken ur breakfast?
```

```
Gd morning,how is life today?Gd?Taken ur breakfast?
Actual Output:
Good morning, how is life today? Good? Taken your breakfast?
Predicted Output for beam==3 :
good morning how is today good is your your
good morning how is today good is your right
good morning how is today good is your
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
No lah... But borburn coke is one o e more popular drinks lor... So is li en dancing?
Actual Output:
No. But Borburn coke is one of the more popular drinks. So is Li En dancing?
Predicted Output for beam==3 :
no no but i'm a is one and the more more a then so is
no no but i'm a is one and the more more coffee then so is
no no but i'm a is one the the more more a then so is
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Ok lor... I buy dinner for them now oredi...
Actual Output:
Ok. I buy dinner for them now already.
Predicted Output for beam==3 :
ok i i buy dinner for out now already
ok i i buy dinner for out now
ok ok i buy dinner for out now already
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
K.reen u change ur number isit?hw cme neber sms 2 mi...so sad..:(
Actual Output:
Ok. Reen, you change your number, is it? How come you didn't SMS to me? So sad.
Predicted Output for beam==3 :
ok reen you change your number is how sms to to so sad
ok reen you change your number is how sms to message so sad
ok reen you change your number is how sms to me so sad
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
SEN.ANYBODY THERE.
Actual Output:
Anybody there?
Predicted Output for beam==3 :
sen anyone there
lea anyone there
hi anyone there
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Haha yiyun? oh yes change no nvr tell me nt my fault arh starhub got contract min 3 months tink they mgh
t nt tk ü leh
Actual Output:
Haha, Yiyun? Oh yes. You change number but never tell me. Not my fault. Starhub got contract minimum 3 m
onths. Think they might not take you.
Predicted Output for beam==3 :
haha i'll oh yes change no never i me am my already got an for to year don't they not you
haha i'll oh yes change no never i me am my already got an for to year don't they not you you
haha i'll oh yes change no never i me am my already got an for to year think they not you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Thanx u darlin!im cool thanx. A few bday drinks 2 nite. 2morrow off! Take care c u soon.
Actual Output:
Thank you darling! I am cool, thanks. A few birthday drinks tonight. Tomorrow off! Take care, see you so
on.
Predicted Output for beam==3 :
thanks you darling i'm nice thanks a few birthday lunch to night tomorrow off take care see you soon
thanks you darling i'm nice thanks a few birthday coffee to night tomorrow off take care see you soon
thanks you darling i'm cool thanks a few birthday lunch to night tomorrow off take care see you soon
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Ok ok just thought u want a lift. I may go down earlier too. Will call u... Need to find a good tailor i
n far east.
Actual Output:
Ok, just thought you want a lift. I may go down earlier too. Will call you. Need to find a good tailor i
n Far East.
```

```
n Far East.
Predicted Output for beam==3 :
ok ok just i you want a and i may go down later too will call you need to find a good in far east
ok ok just i you want a and i may go down later too will call you need to find a a in far east
ok ok just i you want a help i may go down later too will call you need to find a good in far east
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Where are you and mother and yun
Actual Output:
Where are you and Mother and Yun?
Predicted Output for beam==3 :
where are you you
where are you you mom
where are you are
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Okie... Den u'll reach ard wat time....
Actual Output:
Ok. Then around what time will you reach?
Predicted Output for beam==3 :
ok then reach around what time
ok then reach the what time
ok then then reach around what time
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
Hey mel owes you money right? $5 remind me to pay you...
Actual Output:
Hey, Mel, owes you money right? $5, remind me to pay you.
Predicted Output for beam==3 :
hey you money right to remind me to pay you
hey you money right for remind me to pay you
hey you money right 8 remind me to pay you
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
No probl... Maybe next time when u r free =5
Actual Output:
No problem. Maybe next time when you are free.
Predicted Output for beam==3 :
no maybe next time when you are free
no maybe next time when you are free you
no maybe next time when you are free for
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
How r u? Im slackg at home...Hows work so far?
Actual Output:
How are you? I'm slacking at home. How's your work so far?
Predicted Output for beam==3 :
how are you i'm at home how's work so far
how are you am at home how's work so far
how are you i at home how's work so far
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
No more stairs liao? Its bad for your knees so stoppin is quite good. Wat homework r u rushing? 1pm flig
ht? Ok... Mayb i go snatch josssticks... U know?
Actual Output:
No more stairs? It's bad for your knees, so stopping is quite good. What homework are you rushing? 1PM f
light? OK. Maybe I should go to snatch joss sticks. Do you know?
Predicted Output for beam==3 :
no more already it's bad for your so is quite good what are you going ok maybe i go you know
no more already it's bad for your so is quite good what are you going ok then i go you know
no more already it's bad for your so is quite good what are you coming ok maybe i go you know
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Input text:
yes, i noe,same here. but exams comin..hav to spend more time studyin...less time to meet up le :( u jus
started ah?thn go and study loh,i dun wanna disturb c:
Actual Output:
Yes, I know, same here. But exams are coming, have to spend more time studying, less time to meet up.
You just started? Then go and study, I don't want to disturb you.
Predicted Output for beam==3 :
yes i know you here but exams going i to spend more time studying more time to meet up to you just came
```

```
i then go and to go i don't want further see
yes i know you here but exams going i to spend more time studying more time to meet up to you just came
i then go and to go i don't want want see
yes i know are here but exams going i to spend more time studying more time to meet up to you just came
i then go and to go i don't want further see
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

In [86]:

```
import nltk.translate.bleu_score as bleu
bleu_score1=[]
bleu_score2=[]
bleu_score3=[]
#computing bleu_scores for 20 test points where beam=3

for i in range(20):
  b=list(y_test[i:i+1])
  x=model.predict(source_padded_docs_test[i:i+1])

  res=beam_search_decoder(x[0],3)

  y1=prediction(res[0][0])
  y1=y1.split(' ')
  y_lst1=[]
  for i in y1:
    if i=='<PAD>':
      continue
    else:
      y_lst1.append(i)
  bleu_score1.append(bleu.sentence_bleu([b[0].split(),],y_lst1))

  y2=prediction(res[1][0])
  y2=y2.split(' ')
  y_lst2=[]
  for i in y2:
    if i=='<PAD>':
      continue
    else:
      y_lst2.append(i)
  bleu_score2.append(bleu.sentence_bleu([b[0].split(),],y_lst2))

  y3=prediction(res[2][0])
  y3=y3.split(' ')
  y_lst3=[]
  for i in y3:
    if i=='<PAD>':
      continue
    else:
      y_lst3.append(i)
  bleu_score3.append(bleu.sentence_bleu([b[0].split(),],y_lst3))

print("The Average Bleu Score1 is: ",sum(bleu_score1)/20)
print('>'*180)
print("The Average Bleu Score2 is: ",sum(bleu_score2)/20)
print('>'*180)
print("The Average Bleu Score3 is: ",sum(bleu_score3)/20)
print('>'*180)
```

```
/usr/local/lib/python3.7/dist-packages/nltk/translate/bleu_score.py:490: UserWarning:
Corpus/Sentence contains 0 counts of 3-gram overlaps.
BLEU scores might be undesirable; use SmoothingFunction().
  warnings.warn(_msg)
/usr/local/lib/python3.7/dist-packages/nltk/translate/bleu_score.py:490: UserWarning:
Corpus/Sentence contains 0 counts of 4-gram overlaps.
BLEU scores might be undesirable; use SmoothingFunction().
  warnings.warn(_msg)
/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)
The Average Bleu Score1 is:  0.3248761532555189
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
The Average Bleu Score2 is:  0.3524630607027792
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
The Average Bleu Score3 is:  0.32389016697371803
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

In [ ]: