```
import string
import re
import numpy as np
from numpy import array, argmax, random, take
import pandas as pd



from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM, Embedding, RepeatVector
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.callbacks import ModelCheckpoint
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import load_model
from tensorflow.keras import optimizers


import matplotlib.pyplot as plt
pd.set_option('display.max_colwidth',200)


data_path='fra.txt'
with open(data_path, 'r', encoding='utf-8') as f:
  lines = f.read()

lines
```

```
'Go.\tVa !\tCC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #1158250 (Wittydev)\nGo.\tMarche.\tCC-BY 2.0 (France) Attributi
on: tatoeba.org #2877272 (CM) & #8090732 (Micsmithel)\nGo.\tEn route !\tCC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #82
67435 (felix63)\nGo.\tBouge !\tCC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #9022935 (Micsmithel)\nHi.\tSalut !\tCC-BY
2.0 (France) Attribution: tatoeba.org #538123 (CM) & #509819 (Aiji)\nHi.\tSalut.\tCC-BY 2.0 (France) Attribution: tatoeba.org #538123
(CM) & #4320462 (gillux)\nRun!\tCours\u202f!\tCC-BY 2.0 (France) Attribution: tatoeba.org #906328 (papabear) & #906331 (sacredceltic)\n
Run!\tCourez\u202f!\tCC-BY 2.0 (France) Attribution: tatoeba.org #906328 (papabear) & #906332 (sacredceltic)\nRun!\tPrenez vos jambes à
vos cous !\tCC-BY 2.0 (France) Attribution: tatoeba.org #906328 (papabear) & #2077449 (sacredceltic)\nRun!\tFile !\tCC-BY 2.0 (France)
```

```
def to_lines(text):
  sents=text.strip().split('\n')
  sents=[i.split('\t') for i in sents ]
  return sents


fra_eng=to_lines(lines)
fra_eng[:5]
```

```
[['Go.',
  'Va !',
  'CC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #1158250 (Wittydev)'],
 ['Go.',
  'Marche.',
  'CC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #8090732 (Micsmithel)'],
 ['Go.',
  'En route !',
  'CC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #8267435 (felix63)'],
 ['Go.',
  'Bouge !',
  'CC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #9022935 (Micsmithel)'],
 ['Hi.',
  'Salut !',
  'CC-BY 2.0 (France) Attribution: tatoeba.org #538123 (CM) & #509819 (Aiji)']]
```

```
fra_eng=array(fra_eng)
fra_eng[:5]
```

```
array([['Go.', 'Va !',
        'CC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #1158250 (Wittydev)'],
       ['Go.', 'Marche.',
        'CC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #8090732 (Micsmithel)'],
       ['Go.', 'En route !',
        'CC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #8267435 (felix63)'],
       ['Go.', 'Bouge !',
        'CC-BY 2.0 (France) Attribution: tatoeba.org #2877272 (CM) & #9022935 (Micsmithel)'],
       ['Hi.', 'Salut !',
        'CC-BY 2.0 (France) Attribution: tatoeba.org #538123 (CM) & #509819 (Aiji)']],
      dtype='<U101')
```

```
fra_eng.shape
```

```
(50875, 3)
```

```
fra_eng=fra_eng[:100000,:]


fra_eng=fra_eng[:,[0,1]]
fra_eng[:5]
```

```
array([['Go.', 'Va !'],
       ['Go.', 'Marche.'],
       ['Go.', 'En route !'],
       ['Go.', 'Bouge !'],
       ['Hi.', 'Salut !']], dtype='<U101')
```

```
fra_eng[:,0]=[s.translate(str.maketrans('','',string.punctuation)) for s in fra_eng[:,0]]
fra_eng[:,1]=[s.translate(str.maketrans('','',string.punctuation)) for s in fra_eng[:,1]]
fra_eng[:5]
```

```
array([['Go', 'Va '],
       ['Go', 'Marche'],
       ['Go', 'En route '],
       ['Go', 'Bouge '],
       ['Hi', 'Salut ']], dtype='<U101')
```

```
for i in range(len(fra_eng)):
  fra_eng[i,0]=fra_eng[i,0].lower()
  fra_eng[i,1]=fra_eng[i,1].lower()

fra_eng
```

```
array([['go', 'va '],
       ['go', 'marche'],
       ['go', 'en route '],
       ...,
       ['its not a classroom', 'ce nest pas une salle de classe'],
       ['its not a good time', 'ça tombe assez mal'],
       ['its not about money', 'il ne sagit pas dargent']], dtype='<U101')
```

```
def tokenization(lines):
  tokenizer=Tokenizer()
  tokenizer.fit_on_texts(lines)
  return tokenizer

eng_tokenizer=tokenization(fra_eng[:,0])
eng_vocab_size=len(eng_tokenizer.word_index)+1

eng_length=8
print('English vocabulary size:' , eng_vocab_size)
```

```
English vocabulary size: 6028
```

```
fra_tokenizer=tokenization(fra_eng[:,1])
fra_vocab_size=len(fra_tokenizer.word_index)+1
fra_length=8
print('French Vocabulary Size: ', fra_vocab_size)
```

```
French Vocabulary Size:  13842
```

```
def encode_sequences(tokenizer,length, lines):
  seq=tokenizer.texts_to_sequences(lines)
  seq=pad_sequences(seq,maxlen=length,padding='post')
  return seq


from sklearn.model_selection import train_test_split
train, test= train_test_split(fra_eng,test_size=0.2,random_state=12)


trainX=encode_sequences(fra_tokenizer,fra_length,train[:,1])
trainY= encode_sequences(eng_tokenizer,eng_length,train[:,0])

testX=encode_sequences(fra_tokenizer,fra_length,test[:,1])
testY=encode_sequences(eng_tokenizer, eng_length, test[:,0])


def define_model(in_vocab,out_vocab, in_timesteps,out_timesteps, units):
  model=Sequential()
  model.add(Embedding(in_vocab, units, input_length=in_timesteps, mask_zero=True))
```

```
  model.add(LSTM(units))
  model.add(RepeatVector(out_timesteps))
  model.add(LSTM(units,return_sequences=True))
  model.add(Dense(out_vocab, activation='softmax'))
  return model


model=define_model(fra_vocab_size, eng_vocab_size, fra_length, eng_length, 512)
rms=optimizers.RMSprop(learning_rate=0.001)
model.compile(optimizer=rms,loss='sparse_categorical_crossentropy')
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/layers/core/embedding.py:90: UserWarning: Argument `input_length` is deprecated. Just
  warnings.warn(
```

```
history=model.fit(trainX, trainY.reshape(trainY.shape[0], trainY.shape[1],1), epochs=30, batch_size=512, validation_split=0.2)
```

```
Epoch 1/30
64/64 ━━━━━━━━━━━━━━━━━━━━ 963s 15s/step - loss: 5.6927 - val_loss: 3.0776
Epoch 2/30
64/64 ━━━━━━━━━━━━━━━━━━━━ 948s 14s/step - loss: 2.9570 - val_loss: 2.7746
Epoch 3/30
26/64 ━━━━━━━━━━━━━━━━━━━━ 8:35 14s/step - loss: 2.7630
```

```
preds_probs = model.predict(testX)
preds = np.argmax(preds_probs, axis=-1)
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 4s 12ms/step
```

```
preds
```

```
array([[17,  2, 11, ...,  0,  0,  0],
       [ 1, 15,  4, ...,  0,  0,  0],
       [14,  5,  4, ...,  0,  0,  0],
       ...,
       [14, 15,  4, ...,  0,  0,  0],
       [39, 89,  9, ...,  0,  0,  0],
       [13,  2,  2, ...,  0,  0,  0]])
```

```
def get_word(n,tokenizer):
  for word,index in tokenizer.word_index.items():
    if index==n:
      return word
  return None


preds_text=[]
for i in preds:
  temp=[]
  for j in range(len(i)):
    t=get_word(i[j],eng_tokenizer)
    if j>0:
      if(t==get_word(i[j-1], eng_tokenizer)) or (t==None):
        temp.append('')
      else:
        temp.append(t)
    else:
      if(t==None):
        temp.append('')
      else:
        temp.append(t)
  preds_text.append(' '.join(temp))


pred_df=pd.DataFrame({'actual': test[:,0], 'predicted': preds_text})


pred_df.sample(20)
```

	actual	predicted
1214	i didnt go	i cant to
5649	thats mine	its is
7385	i am sure	im you
6864	it looked fresh	it is
5108	were here alone	were
9186	what was tom eating	how was me
6711	i felt scared	i cant to
9046	do you have any beer	i to
9332	i really missed you	you
3474	i bike to work	i cant to
3967	this is my horse	its is
633	its just wrong	its is
8692	my sunburn hurts	i cant to
4332	i miss you a lot	you
9909	youre hopeless	youre are
2721	toms a bright kid	tom is a
7933	get out of my bed	how you me
1489	are you giving up	lets it
1057	just do the job	you me
7202	i buried it	i want to

```
import matplotlib.pyplot as plt


plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.show()
```