```
import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.layers import Embedding, LSTM, Dense
from tensorflow.keras.models import Sequential
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.optimizers import Adam
import pickle
import numpy as np
import os


from google.colab import files
uploaded = files.upload()
```

Choose Files gan.txt

- **gan.txt**(text/plain) - 25 bytes, last modified: 11/9/2022 - 100% done

Saving gan.txt to gan (8).txt

```
file = open("gan (8).txt", "r", encoding = "utf8")

# store file in list
lines = []
for i in file:
    lines.append(i)

# Convert list to string
data = ""
for i in lines:
  data = ' '. join(lines)

#replace unnecessary stuff with space
data = data.replace('\n', '').replace('\r', '').replace('\ufeff', '').replace('“','').repl

#remove unnecessary spaces
data = data.split()
data = ' '.join(data)
data[:500]
```

```
'am ap haa amma appa hello'
```

```
len(data)
```

```
25
```

```
tokenizer = Tokenizer()
tokenizer.fit_on_texts([data])

# saving the tokenizer for predict function
```

```
pickle.dump(tokenizer, open('token.pkl', 'wb'))

sequence_data = tokenizer.texts_to_sequences([data])[0]
sequence_data[:15]
```

```
[1, 2, 3, 4, 5, 6]
```

```
len(sequence_data)
```

```
6
```

```
vocab_size = len(tokenizer.word_index) + 1
print(vocab_size)
```

```
7
```

```
sequences = []

for i in range(3, len(sequence_data)):
    words = sequence_data[i-3:i+1]
    sequences.append(words)

print("The Length of sequences are: ", len(sequences))
sequences = np.array(sequences)
sequences[:10]
```

```
The Length of sequences are:  3
array([[1, 2, 3, 4],
       [2, 3, 4, 5],
       [3, 4, 5, 6]])
```

```
X = []
y = []

for i in sequences:
    X.append(i[0:3])
    y.append(i[3])

X = np.array(X)
y = np.array(y)
```

```
print("Data: ", X[:10])
print("Response: ", y[:10])
```

```
Data:  [[1 2 3]
 [2 3 4]
 [3 4 5]]
Response:  [4 5 6]
```

```
y = to_categorical(y, num_classes=vocab_size)
y[:5]
```

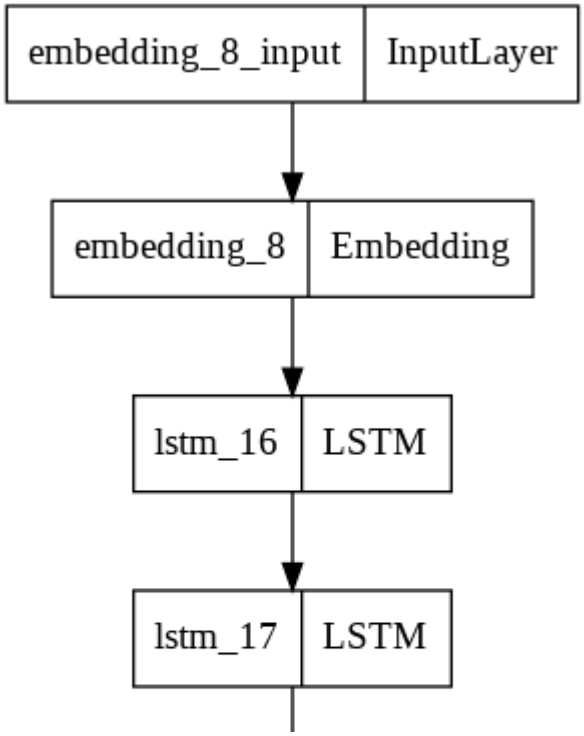
```
array([[0., 0., 0., 0., 1., 0., 0.],
       [0., 0., 0., 0., 0., 1., 0.],
       [0., 0., 0., 0., 0., 0., 1.]], dtype=float32)
```

```
model = Sequential()
model.add(Embedding(vocab_size, 10, input_length=3))
model.add(LSTM(1000, return_sequences=True))
model.add(LSTM(1000))
model.add(Dense(1000, activation="relu"))
model.add(Dense(vocab_size, activation="softmax"))
```

```
model.summary()
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 embedding_8 (Embedding)     (None, 3, 10)             70        

 lstm_16 (LSTM)              (None, 3, 1000)           4044000   

 lstm_17 (LSTM)              (None, 1000)              8004000   

 dense_16 (Dense)            (None, 1000)              1001000   

 dense_17 (Dense)            (None, 7)                 7007      

=================================================================
Total params: 13,056,077
Trainable params: 13,056,077
Non-trainable params: 0
_________________________________________________________________
```

```
from tensorflow import keras
from keras.utils.vis_utils import plot_model

keras.utils.plot_model(model, to_file='plot.png', show_layer_names=True)
```




```
from tensorflow.keras.callbacks import ModelCheckpoint

checkpoint = ModelCheckpoint("next_words.h5", monitor='loss', verbose=1, save_best_only=Tr
model.compile(loss="categorical_crossentropy", optimizer=Adam(learning_rate=0.001))
model.fit(X, y, epochs=70, batch_size=64, callbacks=[checkpoint])
```

```
Epoch 38: loss improved from 0.00574 to 0.00543, saving model to next_words.h5
1/1 [==============================] - 1s 638ms/step - loss: 0.0054
Epoch 39/70
1/1 [==============================] - ETA: 0s - loss: 0.0060
Epoch 39: loss did not improve from 0.00543
1/1 [==============================] - 0s 23ms/step - loss: 0.0060
Epoch 40/70
1/1 [==============================] - ETA: 0s - loss: 0.0052
Epoch 40: loss improved from 0.00543 to 0.00520, saving model to next_words.h5
1/1 [==============================] - 1s 724ms/step - loss: 0.0052
Epoch 41/70
1/1 [==============================] - ETA: 0s - loss: 0.0034
Epoch 41: loss improved from 0.00520 to 0.00338, saving model to next_words.h5
1/1 [==============================] - 0s 480ms/step - loss: 0.0034
Epoch 42/70
1/1 [==============================] - ETA: 0s - loss: 0.0018
Epoch 42: loss improved from 0.00338 to 0.00181, saving model to next_words.h5
1/1 [==============================] - 1s 599ms/step - loss: 0.0018
Epoch 43/70
1/1 [==============================] - ETA: 0s - loss: 9.1444e-04
Epoch 43: loss improved from 0.00181 to 0.00091, saving model to next_words.h5
1/1 [==============================] - 1s 939ms/step - loss: 9.1444e-04
Epoch 44/70
1/1 [==============================] - ETA: 0s - loss: 4.9063e-04
Epoch 44: loss improved from 0.00091 to 0.00049, saving model to next_words.h5
1/1 [==============================] - 0s 383ms/step - loss: 4.9063e-04
Epoch 45/70
1/1 [==============================] - ETA: 0s - loss: 3.0345e-04
Epoch 45: loss improved from 0.00049 to 0.00030, saving model to next_words.h5
```

```
Epoch 45: loss improved from 0.00049 to 0.00030, saving model to next_words.h5
1/1 [==============================] - 1s 538ms/step - loss: 3.0345e-04
Epoch 46/70
1/1 [==============================] - ETA: 0s - loss: 2.2857e-04
Epoch 46: loss improved from 0.00030 to 0.00023, saving model to next_words.h5
1/1 [==============================] - 1s 622ms/step - loss: 2.2857e-04
Epoch 47/70
1/1 [==============================] - ETA: 0s - loss: 2.0419e-04
Epoch 47: loss improved from 0.00023 to 0.00020, saving model to next_words.h5
1/1 [==============================] - 1s 653ms/step - loss: 2.0419e-04
Epoch 48/70
1/1 [==============================] - ETA: 0s - loss: 2.0276e-04
Epoch 48: loss improved from 0.00020 to 0.00020, saving model to next_words.h5
1/1 [==============================] - 1s 704ms/step - loss: 2.0276e-04
Epoch 49/70
1/1 [==============================] - ETA: 0s - loss: 2.1071e-04
Epoch 49: loss did not improve from 0.00020
1/1 [==============================] - 0s 35ms/step - loss: 2.1071e-04
Epoch 50/70
1/1 [==============================] - ETA: 0s - loss: 2.1826e-04
Epoch 50: loss did not improve from 0.00020
1/1 [==============================] - 0s 20ms/step - loss: 2.1826e-04
Epoch 51/70
1/1 [==============================] - ETA: 0s - loss: 2.2057e-04
Epoch 51: loss did not improve from 0.00020
1/1 [==============================] - 0s 28ms/step - loss: 2.2057e-04
Epoch 52/70
1/1 [==============================] - ETA: 0s - loss: 2.1461e-04
Epoch 52: loss did not improve from 0.00020
1/1 [==============================] - 0s 21ms/step - loss: 2.1461e-04
Epoch 53/70
```

```
from tensorflow.keras.models import load_model
import numpy as np
import pickle

# Load the model and tokenizer
model = load_model('next_words.h5')
tokenizer = pickle.load(open('token.pkl', 'rb'))

def Predict_Next_Words(model, tokenizer, text):

  sequence = tokenizer.texts_to_sequences([text])
  sequence = np.array(sequence)
  preds = np.argmax(model.predict(sequence))
  predicted_word = ""

  for key, value in tokenizer.word_index.items():
      if value == preds:
          predicted_word = key
          break

  print(predicted_word)
  return predicted_word
```

```
while(True):
  text = input("Enter your line: ")

  if text == "0":
      print("Execution completed.....")
      break

  else:
      try:
          text = text.split(" ")
          text = text[-1:]
          print(text)

          Predict_Next_Words(model, tokenizer, text)

      except Exception as e:
        print("Error occurred: ",e)
        continue
```

```
Enter your line: am
['am']
1/1 [==============================] - 0s 18ms/step
amma
Enter your line: ap
['ap']
1/1 [==============================] - 0s 16ms/step
appa
Enter your line: haa
['haa']
1/1 [==============================] - 0s 20ms/step
hello
Enter your line: 0
Execution completed.....
```

Colab paid products - Cancel contracts here

✓ 19s completed at 11:30 AM