```
!pip install tensorflow
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-packages (2.15.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (23.5.26)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.5.4)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.9.0)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (16.0.6)
Requirement already satisfied: ml-dtypes~=0.2.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.25.2)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from tensorflow) (23.2)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in /usr/local/lib/python
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from tensorflow) (67.7.2)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.4.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (4.10.0)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.36.0
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.62.0)
Requirement already satisfied: tensorboard<2.16,>=2.15 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.2)
Requirement already satisfied: tensorflow-estimator<2.16,>=2.15.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.0)
Requirement already satisfied: keras<2.16,>=2.15.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from astunparse>=1.6.0->tensorflow) (0.42.
Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->tensorflo
Requirement already satisfied: google-auth-oauthlib<2,>=0.5 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->te
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->tensorflow) (3.
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->tensorflow)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->tensorflow) (3.
Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensorboar
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensorboard
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.16,>=
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from google-auth-oauthlib<2,>=0.5->t
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboar
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboard<2.16,>=2.1
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboard<2.16
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboard<2.16
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from werkzeug>=1.0.1->tensorboard<2.16,>=2.
Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-auth<
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oa
```

```
import numpy as np
from keras.models import Sequential
from keras.layers import SimpleRNN,Dense,Embedding
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.utils import to_categorical
```

```
sentences=["I love learning","I love python","I hate school","Recurrent Neural Network are powerful"]
```

```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(sentences)
total_words=len(tokenizer.word_index)+1
print(total_words)
```

```
12
```

```
import pandas as pd
df=pd.read_csv("/content/earth.1.txt",sep=".")
```

```
# Creating input sequences and their corresponding next words
input_sequences = []
for sentence in df:
    tokenized_sentence = tokenizer.texts_to_sequences([sentence])[0]
    for i in range(1, len(tokenized_sentence)):
        n_gram_sequence = tokenized_sentence[:i+1]
        input_sequences.append(n_gram_sequence)
input_sequences
```

```
[]
```

```
# Padding sequences for consistent input size
max_sequence_length = max([len(seq) for seq in input_sequences])
input_sequences = pad_sequences(input_sequences,maxlen=max_sequence_length,padding='pre')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-36-bb1cbb90ca40> in <cell line: 2>()
      1 # Padding sequences for consistent input size
----> 2 max_sequence_length = max([len(seq) for seq in input_sequences])
      3 input_sequences = pad_sequences(input_sequences,maxlen=max_sequence_length,padding='pre')

ValueError: max() arg is an empty sequence
```

```
# Creating input and output data
X, y = input_sequences[:, :-1], input_sequences[:, -1]
y = to_categorical(y, num_classes=total_words)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-37-69a9654a827b> in <cell line: 2>()
      1 # Creating input and output data
----> 2 X, y = input_sequences[:, :-1], input_sequences[:, -1]
      3 y = to_categorical(y, num_classes=total_words)

TypeError: list indices must be integers or slices, not tuple
```

```
# Building a simple RNN model
model = Sequential()
model.add(Embedding(input_dim=total_words, output_dim=50, input_length=max_sequence_length-1))
model.add(SimpleRNN(100, return_sequences=True))
model.add(SimpleRNN(100))
model.add(Dense(total_words, activation='softmax'))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-38-84ee22014762> in <cell line: 3>()
      1 # Building a simple RNN model
      2 model = Sequential()
----> 3 model.add(Embedding(input_dim=total_words, output_dim=50, input_length=max_sequence_length-1))
      4 model.add(SimpleRNN(100, return_sequences=True))
      5 model.add(SimpleRNN(100))

NameError: name 'max_sequence_length' is not defined
```

```
# compiling the model
model.compile(optimizer="adam",loss="categorical_crossentropy",metrics=["accuracy"])
model.fit(X,y,epochs=50,verbose=2)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-39-758ae04d247c> in <cell line: 3>()
      1 # compiling the model
      2 model.compile(optimizer="adam",loss="categorical_crossentropy",metrics=["accuracy"])
----> 3 model.fit(X,y,epochs=50,verbose=2)

NameError: name 'X' is not defined
```

```
# Generating text using the trained model
seed_text = input("Enter the starting word: ")
next_words = int(input("Enter how many words to predict: "))

for _ in range(next_words):
    tokenized_seed = tokenizer.texts_to_sequences([seed_text])[0]
    tokenized_seed = pad_sequences([tokenized_seed], maxlen=max_sequence_length-1, padding='pre')
    predicted_word_index = np.argmax(model.predict(tokenized_seed), axis=-1)
    predicted_word = tokenizer.index_word[predicted_word_index[0]]
    seed_text += " " + predicted_word

print(seed_text)
```

```
Enter the starting word: 59
Enter how many words to predict: 49
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-40-cf37750cc438> in <cell line: 5>()
      5 for _ in range(next_words):
      6     tokenized_seed = tokenizer.texts_to_sequences([seed_text])[0]
----> 7     tokenized_seed = pad_sequences([tokenized_seed], maxlen=max_sequence_length-1, padding='pre')
      8     predicted_word_index = np.argmax(model.predict(tokenized_seed), axis=-1)
      9     predicted_word = tokenizer.index_word[predicted_word_index[0]]

NameError: name 'max_sequence_length' is not defined
```

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
#Define image size and batch
IMG_SIZE=224
BATCH_SIZE=32
```

```
#creating training data
train_datagen=ImageDataGenerator(
    rescale=1./225,
    validation_split=0.2
)
```

```
#creating training data with above parameters
#folder=parameters.flow_from_directorty(path,target_size,batch_size,class_mode,subset)
train_generator=train_datagen.flow_from_directory('/content/drive/MyDrive/Brain_Tumor_Detection/Train',target_size=(IMG_SIZE,IMG_SIZE),batch
```

```
Found 2408 images belonging to 2 classes.
```

```
valid_generator=train_datagen.flow_from_directory('/content/drive/MyDrive/Brain_Tumor_Detection/Train',
                                                  target_size=(IMG_SIZE,IMG_SIZE),
                                                  batch_size=BATCH_SIZE,
                                                  class_mode='binary',
                                                  subset='validation'
)
```

```
Found 602 images belonging to 2 classes.
```

```
#Define the  model
import keras
from keras import layers
model=keras.Sequential([
    layers.Conv2D(32,(3,3),activation='relu',input_shape=(IMG_SIZE,IMG_SIZE,3)),
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(64,(3,3),activation="relu"),
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(128,(3,3),activation="relu"),
    layers.MaxPooling2D((2,2)),
    layers.Flatten(),
    layers.Dense(128,activation='relu'),
    layers.Dense(1,activation="sigmoid")
])
```

```
model.compile(optimizer="adam",loss="binary_crossentropy",metrics=['accuracy'])
```

```
model.fit(train_generator,validation_data=valid_generator ,epochs=5)
```

```
Epoch 1/5
76/76 [==============================] - 376s 5s/step - loss: 0.4749 - accuracy: 0.8032 - val_loss: 0.2526 - val_accuracy: 0.8953
Epoch 2/5
76/76 [==============================] - 312s 4s/step - loss: 0.1868 - accuracy: 0.9282 - val_loss: 0.1414 - val_accuracy: 0.9585
Epoch 3/5
76/76 [==============================] - 311s 4s/step - loss: 0.0850 - accuracy: 0.9722 - val_loss: 0.0637 - val_accuracy: 0.9850
Epoch 4/5
76/76 [==============================] - 314s 4s/step - loss: 0.0373 - accuracy: 0.9921 - val_loss: 0.0259 - val_accuracy: 0.9900
Epoch 5/5
76/76 [==============================] - 294s 4s/step - loss: 0.0340 - accuracy: 0.9904 - val_loss: 0.0340 - val_accuracy: 0.9917
```

```
<keras.src.callbacks.History at 0x7d47e0a54940>
```

```
model.save("Model.h5","label.txt")
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `m
  saving_api.save_model(
```