Developing Simple Chatbot using Python and Deep Learning

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

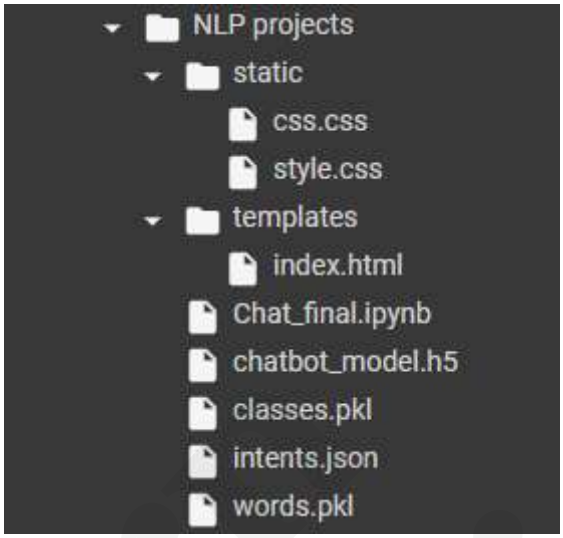
A Chatbot is an application that is used to manage an online chat conversation through text or text to speech format. Most of the chatbots are accessed online through various websites or assistances with a popup.

Examples:- E-commerce websites, health, news, etc.

Basic workflow for building chatbot:

1. Import and load the data file
2. Preprocess data
3. Create training and testing data
4. Build the model
5. Predict the response

Note:



I have provided the files in the links given below:
static:
https://drive.google.com/drive/folders/1o48FdC0MeAQoA_0xANrPpg-pXQ5aCwD6?usp=sharing
(https://drive.google.com/drive/folders/1o48FdC0MeAQoA_0xANrPpg-pXQ5aCwD6?usp=sharing)

templates:
https://drive.google.com/drive/folders/1bVIVMDEXRSFvOAICRkzlll_S3cqWJwPZ?usp=sharing
(https://drive.google.com/drive/folders/1bVIVMDEXRSFvOAICRkzlll_S3cqWJwPZ?usp=sharing)

intents.json:
https://drive.google.com/file/d/1z4L3Oba6biQgl6gnMc2SUVmzcMUkqcjb/view?usp=sharing
(https://drive.google.com/file/d/1z4L3Oba6biQgl6gnMc2SUVmzcMUkqcjb/view?usp=sharing)

You need to upload the files in the given format only. Rest other files will get create while executing the code.
If you are unable to upload the files these the proper format then the code might give error.

In [ ]:
```
# import the  libraries
import random
from tensorflow import keras
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.models import load_model
from tensorflow.keras.models import Sequential
import numpy as np
import pickle
import json
```

In [ ]:
```
#import nltk,WordNetLemmatizer
#download punkt and wordnet

import nltk
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
nltk.download("punkt")
nltk.download("wordnet")
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
```

Out[ ]:
```
True
```

Now create a folder in drive and add path here to make use of files and intents.

In [ ]:
```
cd /content/drive/MyDrive/Aiforeverything/Chatbot
```

```
/content/drive/MyDrive/Aiforeverything/Chatbot
```

In [ ]:
```
!ls
```

```
chatbot_model.h5  intents.json                          static    words.pkl
classes.pkl       learneverythingaiChatbot_final.ipynb  template
```

Now can download the intent file here:
https://drive.google.com/file/d/1z4L3Oba6biQgl6gnMc2SUVmzcMUkqcjb/view?usp=sharing
(https://drive.google.com/file/d/1z4L3Oba6biQgl6gnMc2SUVmzcMUkqcjb/view?usp=sharing)

In [ ]:
```
# init file
#create a lists of words, classes and documents.
#open the intent.json file and read that file
words = []
classes = []
documents = []
ignore_words = ["?", "!"]
data_file = open("/content/drive/MyDrive/Aiforeverything/Chatbot/intents.json"
).read()
intents = json.loads(data_file)
```

In [ ]:
```
#iterate over intents
#now iterate over the patterns
# take each word and tokenize it
for intent in intents["intents"]:
    for pattern in intent["patterns"]:
        w = nltk.word_tokenize(pattern)
        words.extend(w)
        documents.append((w, intent["tag"]))
        if intent["tag"] not in classes:
            classes.append(intent["tag"])
```

In [ ]:
```
# lemmatizer
#first the lower th words and then iterate throught words and check if it is n
ot present in the ignore words
words = [lemmatizer.lemmatize(w.lower()) for w in words if w not in ignore_wor
ds]
words = sorted(list(set(words)))
classes = sorted(list(set(classes)))
```

In [ ]:
```
#print the len of documents
print(len(documents), "documents")
print(len(classes), "classes", classes)
print(len(words), "unique lemmatized words", words)
```

```
100 documents
55 classes ['AI', 'abbr', 'artificial', 'bend', 'body', 'bot1', 'breathe', 'b
usiness', 'chatbot', 'chatterbox', 'clone', 'comp', 'computer', 'control', 'c
ramped', 'date', 'death', 'do', 'events', 'fav', 'fight', 'goodbye', 'greetin
gs', 'hardware', 'hobby', 'idea', 'imortal', 'lang', 'laugh', 'lie', 'machin
e', 'malfunction', 'motormouth', 'move', 'name', 'name1', 'need', 'noanswer',
'os', 'program', 'programming', 'ratchet', 'robotics', 'robots', 'robotss',
'sapient', 'sense', 'sentiment', 'shoe', 'sound', 'stupid', 'thanks', 'usag
e', 'who', 'wt']
126 unique lemmatized words ["'m", "'s", ',', 'a', 'ai', 'all', 'allowed', 'a
m', 'an', 'are', 'artificial', 'awesome', 'be', 'being', 'bend', 'body', 'bo
t', 'breathe', 'business', 'bye', 'can', 'chat', 'chatterbox', 'clone', 'coff
ee', 'computer', 'control', 'cramped', 'data', 'date', 'die', 'do', 'entity',
'event', 'favorite', 'favour', 'fight', 'for', 'good', 'great', 'hardware',
'haroo', 'hello', 'help', 'helpful', 'helping', 'hey', 'hi', 'hobby', 'holl
a', 'hope', 'how', 'i', 'idea', 'immortal', 'in', 'is', 'it', 'jaw', 'kind',
'language', 'later', 'laugh', 'lie', 'like', 'linguistic', 'making', 'malfunc
tion', 'mate', 'me', 'motormouth', 'move', 'my', 'name', 'need', 'not', 'of',
'okay', 'on', 'operating', 'out', 'over', 'popping', 'product', 'program', 'p
rogramming', 'purpose', 'ratchet', 'robot', 'robotics', 'sapient', 'see', 'se
nse', 'sentient', 'shoe', 'should', 'size', 'sound', 'stupid', 'system', 'tak
e', 'thank', 'thanks', 'thankyou', 'that', 'the', 'there', 'to', 'true', 'typ
e', 'upcoming', 'use', 'walk', 'want', 'wassup', 'what', 'when', 'who', 'wil
l', 'work', 'wow', 'written', 'wtf', 'yaw', 'you', 'your']
```

For more about pickle refer :
https://docs.python.org/3/library/pickle.html (https://docs.python.org/3/library/pickle.html)

In [ ]:
```
#create pickle dump for words and open with words.pkl and write in wb mode
pickle.dump(words, open("words.pkl", "wb"))
#create pickle dump for classes and open with classes.pkl and write in wb mode
pickle.dump(classes, open("classes.pkl", "wb"))
```

In [ ]:
```
# training initializer
# initializing training data
training = []
output_empty = [0] * len(classes)
for doc in documents:
    bag = []
    pattern_words = doc[0]
    pattern_words = [lemmatizer.lemmatize(word.lower()) for word in pattern_wo
rds]
    for w in words:
        bag.append(1) if w in pattern_words else bag.append(0)
    output_row = list(output_empty)
    output_row[classes.index(doc[1])] = 1
    training.append([bag, output_row])
```

In [ ]:
```
# shuffle our features and turn into np.array
random.shuffle(training)
training = np.array(training)
# create train and test lists. X - patterns, Y - intents
train_x = list(training[:, 0])
train_y = list(training[:, 1])
print("Training data created")
```

```
Training data created

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: VisibleDeprec
ationWarning: Creating an ndarray from ragged nested sequences (which is a li
st-or-tuple of lists-or-tuples-or ndarrays with different lengths or shapes)
is deprecated. If you meant to do this, you must specify 'dtype=object' when
creating the ndarray.
  This is separate from the ipykernel package so we can avoid doing imports u
ntil
```

Now we train our model by creating a model that has 3 layers. The first layer is 128 neurons, the second layer is 64 neurons and the third output layer contains number of neurons equal to number of intents to predict output intent with softmax.

In [ ]:
```
# actual training
model = Sequential()
model.add(Dense(128, input_shape=(len(train_x[0]),), activation="relu"))
model.add(Dropout(0.5))
model.add(Dense(64, activation="relu"))
model.add(Dropout(0.5))
model.add(Dense(len(train_y[0]), activation="softmax"))
```

After adding all the hidden layers, now we check the summary of our model.

In [ ]:
```
#check the summary
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               16256

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 64)                8256

 dropout_1 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 55)                3575

=================================================================
Total params: 28,087
Trainable params: 28,087
Non-trainable params: 0
_________________________________________________________________
```

The model is compiled with a loss function of categorical_crossentropy and an optimizer of SGD.The metrics for this model are accuracy, which is calculated as the percentage of correct predictions in total number of predictions made.

In [ ]:
```
# Compile model. Stochastic gradient descent with Nesterov accelerated gradien
t gives good results for this model
sgd = SGD(learning_rate=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss="categorical_crossentropy", optimizer=sgd, metrics=["accura
cy"])
```

In [ ]:
```
# fitting our model
hist = model.fit(np.array(train_x), np.array(train_y), epochs=150, batch_size=
5, verbose=1)
#save our model in .h5 extension
model.save("chatbot_model.h5", hist)
print("model created")
```

```
Epoch 1/150
20/20 [==============================] - 1s 2ms/step - loss: 4.0051 - accurac
y: 0.0000e+00
Epoch 2/150
20/20 [==============================] - 0s 2ms/step - loss: 3.9277 - accurac
y: 0.1500
Epoch 3/150
20/20 [==============================] - 0s 2ms/step - loss: 3.8842 - accurac
y: 0.0900
Epoch 4/150
20/20 [==============================] - 0s 2ms/step - loss: 3.8100 - accurac
y: 0.1300
Epoch 5/150
20/20 [==============================] - 0s 2ms/step - loss: 3.6929 - accurac
y: 0.1800
Epoch 6/150
20/20 [==============================] - 0s 2ms/step - loss: 3.6401 - accurac
y: 0.1900
Epoch 7/150
20/20 [==============================] - 0s 2ms/step - loss: 3.5439 - accurac
y: 0.1600
Epoch 8/150
20/20 [==============================] - 0s 2ms/step - loss: 3.4328 - accurac
y: 0.1800
Epoch 9/150
20/20 [==============================] - 0s 2ms/step - loss: 3.3568 - accurac
y: 0.2100
Epoch 10/150
20/20 [==============================] - 0s 2ms/step - loss: 3.2868 - accurac
y: 0.2000
Epoch 11/150
20/20 [==============================] - 0s 2ms/step - loss: 3.1920 - accurac
y: 0.2000
Epoch 12/150
20/20 [==============================] - 0s 2ms/step - loss: 3.0510 - accurac
y: 0.3000
Epoch 13/150
20/20 [==============================] - 0s 2ms/step - loss: 2.9517 - accurac
y: 0.2400
Epoch 14/150
20/20 [==============================] - 0s 2ms/step - loss: 2.8362 - accurac
y: 0.3000
Epoch 15/150
20/20 [==============================] - 0s 2ms/step - loss: 2.6747 - accurac
y: 0.3600
Epoch 16/150
20/20 [==============================] - 0s 2ms/step - loss: 2.6794 - accurac
y: 0.3200
Epoch 17/150
20/20 [==============================] - 0s 2ms/step - loss: 2.6036 - accurac
y: 0.3800
Epoch 18/150
20/20 [==============================] - 0s 2ms/step - loss: 2.4172 - accurac
y: 0.3900
Epoch 19/150
20/20 [==============================] - 0s 2ms/step - loss: 2.3111 - accurac
y: 0.4000
```

```
Epoch 20/150
20/20 [==============================] - 0s 2ms/step - loss: 2.3431 - accurac
y: 0.4000
Epoch 21/150
20/20 [==============================] - 0s 2ms/step - loss: 2.2905 - accurac
y: 0.4200
Epoch 22/150
20/20 [==============================] - 0s 2ms/step - loss: 2.0401 - accurac
y: 0.4200
Epoch 23/150
20/20 [==============================] - 0s 2ms/step - loss: 1.9793 - accurac
y: 0.4900
Epoch 24/150
20/20 [==============================] - 0s 2ms/step - loss: 2.0058 - accurac
y: 0.4700
Epoch 25/150
20/20 [==============================] - 0s 2ms/step - loss: 1.8917 - accurac
y: 0.4500
Epoch 26/150
20/20 [==============================] - 0s 2ms/step - loss: 1.9696 - accurac
y: 0.5100
Epoch 27/150
20/20 [==============================] - 0s 2ms/step - loss: 1.9474 - accurac
y: 0.4900
Epoch 28/150
20/20 [==============================] - 0s 2ms/step - loss: 1.8201 - accurac
y: 0.4900
Epoch 29/150
20/20 [==============================] - 0s 2ms/step - loss: 1.6377 - accurac
y: 0.5600
Epoch 30/150
20/20 [==============================] - 0s 2ms/step - loss: 1.5176 - accurac
y: 0.5900
Epoch 31/150
20/20 [==============================] - 0s 2ms/step - loss: 1.6624 - accurac
y: 0.4800
Epoch 32/150
20/20 [==============================] - 0s 2ms/step - loss: 1.4870 - accurac
y: 0.5900
Epoch 33/150
20/20 [==============================] - 0s 2ms/step - loss: 1.4806 - accurac
y: 0.5900
Epoch 34/150
20/20 [==============================] - 0s 2ms/step - loss: 1.4590 - accurac
y: 0.5700
Epoch 35/150
20/20 [==============================] - 0s 2ms/step - loss: 1.4641 - accurac
y: 0.5400
Epoch 36/150
20/20 [==============================] - 0s 2ms/step - loss: 1.3130 - accurac
y: 0.6200
Epoch 37/150
20/20 [==============================] - 0s 2ms/step - loss: 1.1834 - accurac
y: 0.7000
Epoch 38/150
20/20 [==============================] - 0s 2ms/step - loss: 1.2436 - accurac
y: 0.7000
```

```
Epoch 39/150
20/20 [==============================] - 0s 2ms/step - loss: 1.2485 - accurac
y: 0.6000
Epoch 40/150
20/20 [==============================] - 0s 2ms/step - loss: 1.0835 - accurac
y: 0.6500
Epoch 41/150
20/20 [==============================] - 0s 2ms/step - loss: 1.3568 - accurac
y: 0.5900
Epoch 42/150
20/20 [==============================] - 0s 2ms/step - loss: 1.2257 - accurac
y: 0.5900
Epoch 43/150
20/20 [==============================] - 0s 2ms/step - loss: 1.1359 - accurac
y: 0.7200
Epoch 44/150
20/20 [==============================] - 0s 2ms/step - loss: 0.9841 - accurac
y: 0.7500
Epoch 45/150
20/20 [==============================] - 0s 2ms/step - loss: 1.0716 - accurac
y: 0.7400
Epoch 46/150
20/20 [==============================] - 0s 2ms/step - loss: 1.0819 - accurac
y: 0.6200
Epoch 47/150
20/20 [==============================] - 0s 2ms/step - loss: 1.0871 - accurac
y: 0.6800
Epoch 48/150
20/20 [==============================] - 0s 2ms/step - loss: 0.9932 - accurac
y: 0.6900
Epoch 49/150
20/20 [==============================] - 0s 2ms/step - loss: 0.9510 - accurac
y: 0.7200
Epoch 50/150
20/20 [==============================] - 0s 2ms/step - loss: 1.0060 - accurac
y: 0.6800
Epoch 51/150
20/20 [==============================] - 0s 2ms/step - loss: 0.9544 - accurac
y: 0.7000
Epoch 52/150
20/20 [==============================] - 0s 2ms/step - loss: 0.8536 - accurac
y: 0.7600
Epoch 53/150
20/20 [==============================] - 0s 2ms/step - loss: 0.8757 - accurac
y: 0.7600
Epoch 54/150
20/20 [==============================] - 0s 2ms/step - loss: 0.9331 - accurac
y: 0.7400
Epoch 55/150
20/20 [==============================] - 0s 2ms/step - loss: 0.9640 - accurac
y: 0.7100
Epoch 56/150
20/20 [==============================] - 0s 2ms/step - loss: 0.8242 - accurac
y: 0.7800
Epoch 57/150
20/20 [==============================] - 0s 2ms/step - loss: 0.8803 - accurac
y: 0.7500
```

```
Epoch 58/150
20/20 [==============================] - 0s 2ms/step - loss: 0.9541 - accurac
y: 0.7100
Epoch 59/150
20/20 [==============================] - 0s 2ms/step - loss: 0.8372 - accurac
y: 0.7500
Epoch 60/150
20/20 [==============================] - 0s 2ms/step - loss: 0.7995 - accurac
y: 0.7700
Epoch 61/150
20/20 [==============================] - 0s 2ms/step - loss: 0.7792 - accurac
y: 0.7300
Epoch 62/150
20/20 [==============================] - 0s 2ms/step - loss: 0.7861 - accurac
y: 0.7600
Epoch 63/150
20/20 [==============================] - 0s 2ms/step - loss: 0.8143 - accurac
y: 0.7300
Epoch 64/150
20/20 [==============================] - 0s 2ms/step - loss: 0.7510 - accurac
y: 0.7600
Epoch 65/150
20/20 [==============================] - 0s 2ms/step - loss: 0.7483 - accurac
y: 0.7300
Epoch 66/150
20/20 [==============================] - 0s 2ms/step - loss: 0.8964 - accurac
y: 0.6900
Epoch 67/150
20/20 [==============================] - 0s 2ms/step - loss: 0.7620 - accurac
y: 0.7500
Epoch 68/150
20/20 [==============================] - 0s 2ms/step - loss: 0.7617 - accurac
y: 0.7600
Epoch 69/150
20/20 [==============================] - 0s 2ms/step - loss: 0.7310 - accurac
y: 0.7900
Epoch 70/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5538 - accurac
y: 0.8700
Epoch 71/150
20/20 [==============================] - 0s 2ms/step - loss: 0.8277 - accurac
y: 0.7600
Epoch 72/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6875 - accurac
y: 0.7800
Epoch 73/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6158 - accurac
y: 0.8300
Epoch 74/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6768 - accurac
y: 0.7600
Epoch 75/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5802 - accurac
y: 0.7900
Epoch 76/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6707 - accurac
y: 0.7600
```

```
Epoch 77/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6446 - accurac
y: 0.7600
Epoch 78/150
20/20 [==============================] - 0s 2ms/step - loss: 0.7106 - accurac
y: 0.7900
Epoch 79/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6451 - accurac
y: 0.8600
Epoch 80/150
20/20 [==============================] - 0s 2ms/step - loss: 0.7029 - accurac
y: 0.7900
Epoch 81/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6375 - accurac
y: 0.8200
Epoch 82/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6342 - accurac
y: 0.8100
Epoch 83/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6544 - accurac
y: 0.8000
Epoch 84/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5175 - accurac
y: 0.8500
Epoch 85/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5193 - accurac
y: 0.8600
Epoch 86/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5111 - accurac
y: 0.8500
Epoch 87/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5737 - accurac
y: 0.7500
Epoch 88/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3569 - accurac
y: 0.9100
Epoch 89/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4505 - accurac
y: 0.8600
Epoch 90/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4288 - accurac
y: 0.8700
Epoch 91/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6705 - accurac
y: 0.7800
Epoch 92/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4492 - accurac
y: 0.8700
Epoch 93/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5179 - accurac
y: 0.8500
Epoch 94/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4968 - accurac
y: 0.8400
Epoch 95/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4226 - accurac
y: 0.9000
```

```
Epoch 96/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3947 - accurac
y: 0.8700
Epoch 97/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4258 - accurac
y: 0.8900
Epoch 98/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4511 - accurac
y: 0.8900
Epoch 99/150
20/20 [==============================] - 0s 2ms/step - loss: 0.6160 - accurac
y: 0.7900
Epoch 100/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5214 - accurac
y: 0.8400
Epoch 101/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3619 - accurac
y: 0.9000
Epoch 102/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4255 - accurac
y: 0.8600
Epoch 103/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3729 - accurac
y: 0.8600
Epoch 104/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4060 - accurac
y: 0.9000
Epoch 105/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4681 - accurac
y: 0.8200
Epoch 106/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4977 - accurac
y: 0.8300
Epoch 107/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3916 - accurac
y: 0.8600
Epoch 108/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3483 - accurac
y: 0.9100
Epoch 109/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4308 - accurac
y: 0.8700
Epoch 110/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5133 - accurac
y: 0.8300
Epoch 111/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3799 - accurac
y: 0.8800
Epoch 112/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4897 - accurac
y: 0.8700
Epoch 113/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5784 - accurac
y: 0.8100
Epoch 114/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4576 - accurac
y: 0.8800
```

```
Epoch 115/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3842 - accurac
y: 0.8900
Epoch 116/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3625 - accurac
y: 0.9300
Epoch 117/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3325 - accurac
y: 0.8900
Epoch 118/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4305 - accurac
y: 0.8700
Epoch 119/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3107 - accurac
y: 0.9100
Epoch 120/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3763 - accurac
y: 0.9100
Epoch 121/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5867 - accurac
y: 0.8000
Epoch 122/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4454 - accurac
y: 0.8400
Epoch 123/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3922 - accurac
y: 0.9000
Epoch 124/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3482 - accurac
y: 0.8900
Epoch 125/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4522 - accurac
y: 0.8700
Epoch 126/150
20/20 [==============================] - 0s 2ms/step - loss: 0.2787 - accurac
y: 0.9200
Epoch 127/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3511 - accurac
y: 0.8800
Epoch 128/150
20/20 [==============================] - 0s 2ms/step - loss: 0.2655 - accurac
y: 0.9400
Epoch 129/150
20/20 [==============================] - 0s 2ms/step - loss: 0.5169 - accurac
y: 0.8000
Epoch 130/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3403 - accurac
y: 0.8800
Epoch 131/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4841 - accurac
y: 0.8100
Epoch 132/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4215 - accurac
y: 0.8500
Epoch 133/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4529 - accurac
y: 0.8600
```

```
Epoch 134/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4336 - accurac
y: 0.8700
Epoch 135/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3121 - accurac
y: 0.8800
Epoch 136/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4814 - accurac
y: 0.8700
Epoch 137/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3479 - accurac
y: 0.8900
Epoch 138/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3569 - accurac
y: 0.9200
Epoch 139/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3132 - accurac
y: 0.8900
Epoch 140/150
20/20 [==============================] - 0s 2ms/step - loss: 0.2816 - accurac
y: 0.9100
Epoch 141/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4233 - accurac
y: 0.8500
Epoch 142/150
20/20 [==============================] - 0s 2ms/step - loss: 0.2628 - accurac
y: 0.9200
Epoch 143/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3927 - accurac
y: 0.8700
Epoch 144/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3017 - accurac
y: 0.9300
Epoch 145/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3381 - accurac
y: 0.8600
Epoch 146/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4954 - accurac
y: 0.8500
Epoch 147/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3350 - accurac
y: 0.8900
Epoch 148/150
20/20 [==============================] - 0s 2ms/step - loss: 0.2867 - accurac
y: 0.9000
Epoch 149/150
20/20 [==============================] - 0s 2ms/step - loss: 0.4424 - accurac
y: 0.8300
Epoch 150/150
20/20 [==============================] - 0s 2ms/step - loss: 0.3180 - accurac
y: 0.8700
model created
```

Now our model is trained, So we have to deploy our model with an interface. So for flask implementation we need IDLEs for implementation. But in this project we are going to implement flask with the help of Google Colab Notebook.

So lets start with Deployment

First we will install flask_ngrok for flask implementation in google colab.

In [ ]:
```
#installing flask_ngrok with pip
!pip install flask_ngrok
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Collecting flask_ngrok
  Downloading flask_ngrok-0.0.25-py3-none-any.whl (3.1 kB)
Requirement already satisfied: Flask>=0.8 in /usr/local/lib/python3.7/dist-pa
ckages (from flask_ngrok) (1.1.4)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-pack
ages (from flask_ngrok) (2.23.0)
Requirement already satisfied: Jinja2<3.0,>=2.10.1 in /usr/local/lib/python3.
7/dist-packages (from Flask>=0.8->flask_ngrok) (2.11.3)
Requirement already satisfied: Werkzeug<2.0,>=0.15 in /usr/local/lib/python3.
7/dist-packages (from Flask>=0.8->flask_ngrok) (1.0.1)
Requirement already satisfied: itsdangerous<2.0,>=0.24 in /usr/local/lib/pyth
on3.7/dist-packages (from Flask>=0.8->flask_ngrok) (1.1.0)
Requirement already satisfied: click<8.0,>=5.1 in /usr/local/lib/python3.7/di
st-packages (from Flask>=0.8->flask_ngrok) (7.1.2)
Requirement already satisfied: MarkupSafe>=0.23 in /usr/local/lib/python3.7/d
ist-packages (from Jinja2<3.0,>=2.10.1->Flask>=0.8->flask_ngrok) (2.0.1)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.
7/dist-packages (from requests->flask_ngrok) (2022.6.15)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/
dist-packages (from requests->flask_ngrok) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-
packages (from requests->flask_ngrok) (2.10)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /us
r/local/lib/python3.7/dist-packages (from requests->flask_ngrok) (1.24.3)
Installing collected packages: flask-ngrok
Successfully installed flask-ngrok-0.0.25
```

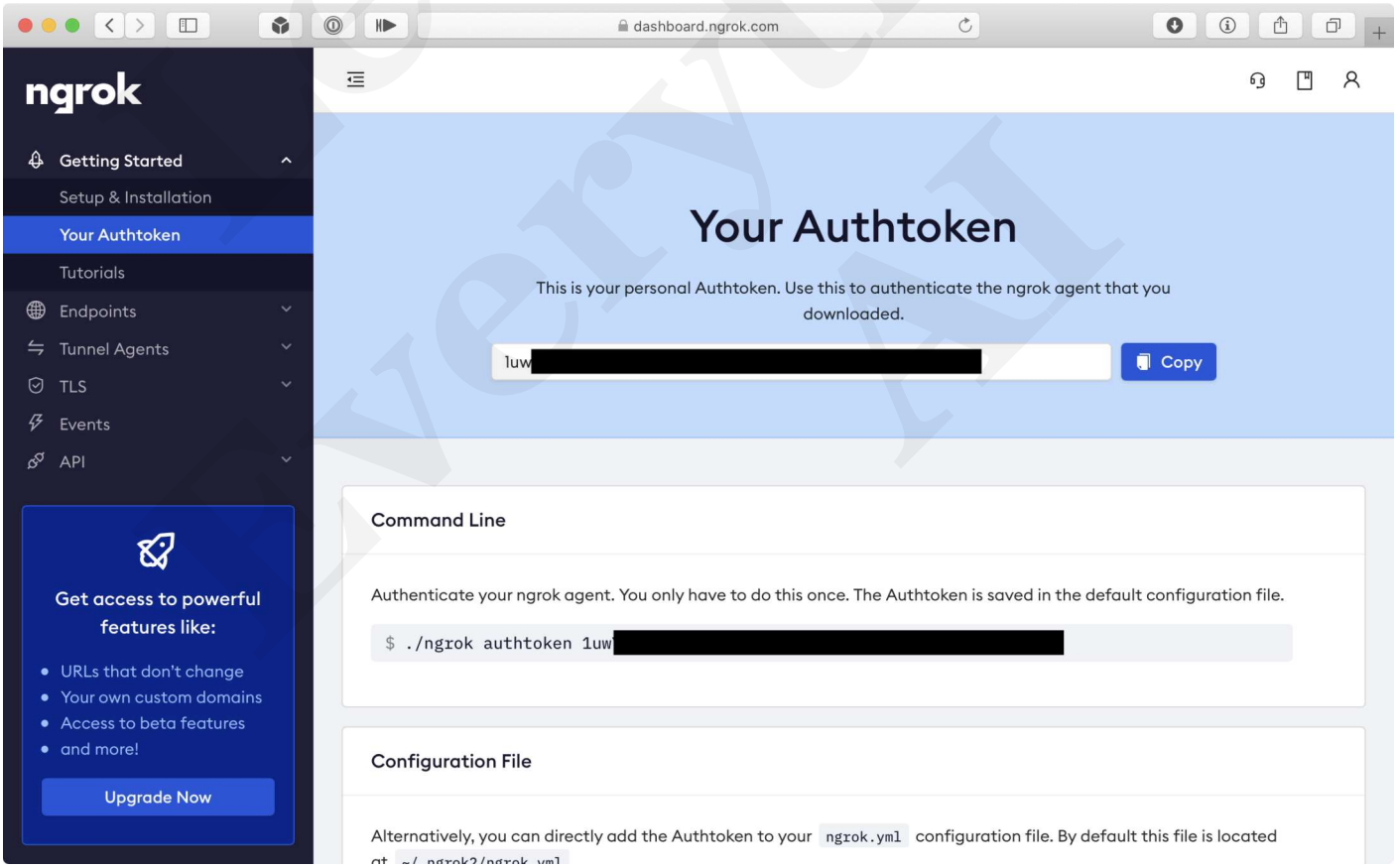
Now we will install and download the ngrok with these commands for installing.

In [ ]:
```
!curl -s https://ngrok-agent.s3.amazonaws.com/ngrok.asc | sudo tee /etc/apt/trusted.gpg.d/ngrok.asc >/dev/null && echo "deb https://ngrok-agent.s3.amazonaws.com buster main" | sudo tee /etc/apt/sources.list.d/ngrok.list && sudo apt update && sudo apt install ngrok
```

```
deb https://ngrok-agent.s3.amazonaws.com buster main
Get:1 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:2 https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/ InRelease
[3,626 B]
Hit:3 http://archive.ubuntu.com/ubuntu bionic InRelease
Get:4 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:5 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:6 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic InRelease
[15.9 kB]
Ign:7 https://developer.download.nvidia.com/compute/machine-learning/repos/ub
untu1804/x86_64  InRelease
Get:8 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86
_64  InRelease [1,581 B]
Get:9 https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/ Packages [9
0.7 kB]
Hit:10 https://developer.download.nvidia.com/compute/machine-learning/repos/u
buntu1804/x86_64  Release
Get:11 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packa
ges [1,533 kB]
Get:12 https://ngrok-agent.s3.amazonaws.com buster InRelease [20.3 kB]
Hit:13 http://ppa.launchpad.net/cran/libgit2/ubuntu bionic InRelease
Get:14 http://security.ubuntu.com/ubuntu bionic-security/restricted amd64 Pac
kages [1,100 kB]
Get:15 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages
[2,937 kB]
Get:16 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu bionic InRelease [15.9
kB]
Get:17 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic InRelease
[21.3 kB]
Get:18 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages
[3,369 kB]
Get:19 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Packa
ges [1,141 kB]
Get:20 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Package
s [2,310 kB]
Get:21 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x8
6_64  Packages [903 kB]
Get:23 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic/main Sourc
es [2,089 kB]
Get:24 https://ngrok-agent.s3.amazonaws.com buster/main amd64 Packages [1,262
B]
Get:25 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic/main amd64
Packages [1,070 kB]
Get:26 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu bionic/main amd64 Packa
ges [45.3 kB]
Get:27 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic/main amd64
Packages [47.7 kB]
Fetched 17.0 MB in 4s (4,837 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
38 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libnvidia-common-460
```

```
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  ngrok
0 upgraded, 1 newly installed, 0 to remove and 38 not upgraded.
Need to get 5,464 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 https://ngrok-agent.s3.amazonaws.com buster/main amd64 ngrok amd64 3.0.
6 [5,464 kB]
Fetched 5,464 kB in 1s (5,105 kB/s)
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based fro
ntend cannot be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 76,
<> line 1.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (This frontend requires a controlling tty.)
debconf: falling back to frontend: Teletype
dpkg-preconfigure: unable to re-open stdin:
Selecting previously unselected package ngrok.
(Reading database ... 155680 files and directories currently installed.)
Preparing to unpack .../archives/ngrok_3.0.6_amd64.deb ...
Unpacking ngrok (3.0.6) ...
Setting up ngrok (3.0.6) ...
```

After importing the libraries, We need to add ngrok token here.



In [ ]:
```
#!ngrok add token here
#!ngrok authtoken 23mMEZeu7mTNdBDg1w3DctEhXpl_5MeopxnhcRvxuaxJGLEk2
!ngrok authtoken 2DLhfpaJspqieAYJCdEEIJIKSft_4PuqHs532NU9A323LHzWD
```

```
Authtoken saved to configuration file: /root/.config/ngrok/ngrok.yml
```

In [ ]:
```
# libraries
#import flask
from flask import *
#import run_with_ngrok from flask_ngrok
from flask_ngrok import run_with_ngrok
#import radom
import random
#import numpy
import numpy as np
#import pickle
import pickle
#import json
import json
#import load_model from tensorflow
from tensorflow.keras.models import load_model
#import nltk
import nltk
#import WordNetLemmatizer from nltk.stem
from nltk.stem import WordNetLemmatizer
#create a object of WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
```

In [ ]:
```
!pip install pyopenssl
```

```
Exception in thread _colab_inspector_thread:
Traceback (most recent call last):
  File "/usr/lib/python3.7/threading.py", line 926, in _bootstrap_inner
    self.run()
  File "/usr/lib/python3.7/threading.py", line 870, in run
    self._target(*self._args, **self._kwargs)
  File "/usr/local/lib/python3.7/dist-packages/google/colab/_debugpy.py", lin
e 64, in inspector_thread
    _variable_inspector.run(shell, time)
  File "/usr/local/lib/python3.7/dist-packages/google/colab/_variable_inspect
or.py", line 27, in run
    globals().clear()
TypeError: 'module' object is not callable


Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Collecting pyopenssl
  Downloading pyOpenSSL-22.0.0-py2.py3-none-any.whl (55 kB)
     |████████████████████████████████| 55 kB 2.7 MB/s
Collecting cryptography>=35.0
  Downloading cryptography-37.0.4-cp36-abi3-manylinux_2_24_x86_64.whl (4.1 M
B)
     |████████████████████████████████| 4.1 MB 18.2 MB/s
Requirement already satisfied: cffi>=1.12 in /usr/local/lib/python3.7/dist-pa
ckages (from cryptography>=35.0->pyopenssl) (1.15.1)
Requirement already satisfied: pycparser in /usr/local/lib/python3.7/dist-pac
kages (from cffi>=1.12->cryptography>=35.0->pyopenssl) (2.21)
Installing collected packages: cryptography, pyopenssl
Successfully installed cryptography-37.0.4 pyopenssl-22.0.0
```

In [ ]:

```
#create a flask app
app = Flask(__name__)
#run_with_ngrok(app)
run_with_ngrok(app)

# chat initialization
#load model chatbot_model.h5
model = load_model("chatbot_model.h5")
#open and reading the intents.json
intents = json.loads(open("/content/drive/MyDrive/Aiforeverything/Chatbot/intents.json").read())
#pickle load for words and open with words.pkl and rb mode
words = pickle.load(open("words.pkl", "rb"))
#pickle load for classes and open with classes.pkl and rb mode
classes = pickle.load(open("classes.pkl", "rb"))

#app.route("/")
@app.route("/")
#def a home function to render index page
def home():
    #return render template index.html
    return render_template("/content/drive/MyDrive/Aiforeverything/Chatbot/template/index.html")

#@app.route("/get",methods=["POST"])
@app.route("/get", methods=["POST"])
#define a chatbot response
def chatbot_response():
    #requesting a msg
    #msg = request.form["msg"]
    msg = request.form["msg"]
    #if user give his/her name so to store that name in a variable
    #if msg.startswith('my name is'):
    if msg.startswith('my name is'):
        #now name =msg[11:]
        name = msg[11:]
        #use predict class and pass the msg and model parameter[#ints = predict_class(msg, model)]
        ints = predict_class(msg, model)
        #use getResponse and pass the ints and intents parameters[res1 = getResponse(ints, intents)]
        res1 = getResponse(ints, intents)
        #replace with name[res =res1.replace("{n}",name)]
        res =res1.replace("{n}",name)
    #if user type hi my name is so length will be increased so +3
    elif msg.startswith('hi my name is'):
        #now name =msg[14:]
        name = msg[14:]
        #use predict class and pass the msg and model parameter[#ints = predict_class(msg, model)]
        ints = predict_class(msg, model)
        #use getResponse and pass the ints and intents parameters[res1 = getResponse(ints, intents)]
        res1 = getResponse(ints, intents)
        #replace with name[res =res1.replace("{n}",name)]
        res =res1.replace("{n}",name)
```

```
    #if user does not type his/her name
    #esle
    else:
        #use predict class and pass the msg and model parameter[#ints = predic
t_class(msg, model)]
        ints = predict_class(msg, model)
        #use getResponse and pass the ints and intents parameters[res1 = getRe
sponse(ints, intents)]
        res = getResponse(ints, intents)
    #return the res
    return res


# chat functionalities
#define a function to clean or preprocess the sentences
def clean_up_sentence(sentence):
    #tokenizing the sentences
    sentence_words = nltk.word_tokenize(sentence)
    #lemmatizing the sentences
    sentence_words = [lemmatizer.lemmatize(word.lower()) for word in sentence_
words]
    #return preprocessed words or sentences
    return sentence_words


# return bag of words array: 0 or 1 for each word in the bag that exists in th
e sentence
#defining a function with sentences and words as parameter
def bow(sentence, words, show_details=True):
    # tokenize the pattern
    sentence_words = clean_up_sentence(sentence)
    # bag of words - matrix of N words, vocabulary matrix[bag = [0] * len(word
s)]
    bag = [0] * len(words)
    #iterate for sentences[for s in sentence_words:]
    for s in sentence_words:
        #now enumerate over words
        for i, w in enumerate(words):
          #if w=s assign 1 if current word is in the vocabulary position
            if w == s:
                # assign 1 if current word is in the vocabulary position
                bag[i] = 1
                #if show_details
                if show_details:
                    #print found baf i w
                    print("found in bag: %s" % w)
    #return numpy array with bag
    return np.array(bag)

#define function of predict class with sentence and model as parameter
def predict_class(sentence, model):
    # filter out predictions below a threshold
    #creating bow of sentence, words [p = bow(sentence, words, show_details=Fa
lse)]
    p = bow(sentence, words, show_details=False)
    #we will predict with numpy array taking 0th postion[res = model.predict(n
p.array([p]))[0]]
```

```
    res = model.predict(np.array([p]))[0]
    #intiailize the threshold value=0.25
    ERROR_THRESHOLD = 0.25
    #results = [[i, r] for i, r in enumerate(res) if r > ERROR_THRESHOLD]
    results = [[i, r] for i, r in enumerate(res) if r > ERROR_THRESHOLD]
    # sort by strength of probability
    results.sort(key=lambda x: x[1], reverse=True)
    #creating a list
    return_list = []
    #iterating in results
    for r in results:
        #appending intent and probabiltiy[ return_list.append({"intent": class
es[r[0]], "probability": str(r[1])})]
        return_list.append({"intent": classes[r[0]], "probability": str(r[1
])})
    #return list
    return return_list


#defing a getResponse with ints and intent_json as parameter
def getResponse(ints, intents_json):
    #tag=ints[0][intent]
    tag = ints[0]["intent"]
    #storing all intents in list of intents[list_of_intents = intents_json["in
tents"]]
    list_of_intents = intents_json["intents"]
    #iterate over lists of intents
    for i in list_of_intents:
        # if tag == tag then print respones
        if i["tag"] == tag:
            result = random.choice(i["responses"])
            break
    #return result
    return result

#if __name__ == "__main__":
if __name__ == "__main__":
    #run the app
    app.run()
    app.debug = True
```

```
 * Serving Flask app "__main__" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deplo
yment.
   Use a production WSGI server instead.
 * Debug mode: off

INFO:werkzeug: * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

 * Running on http://81be-35-199-55-191.ngrok.io
 * Traffic stats available on http://127.0.0.1:4040
```