DS505: INTRODUCTION TO DEEP LEARNING

S02: PROJECT: Chatbot Using Machine Learning

In [1]:

```
import json
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Embedding, GlobalAveragePooling1D
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from sklearn.preprocessing import LabelEncoder
```

In [2]:

```
with open('intents.JSON') as file:
    data = json.load(file)

training_sentences = []
training_labels = []
labels = []
responses = []


for intent in data['intents']:
    for pattern in intent['patterns']:
        training_sentences.append(pattern)
        training_labels.append(intent['tag'])
    responses.append(intent['responses'])

    if intent['tag'] not in labels:
        labels.append(intent['tag'])

num_classes = len(labels)
```

In [3]:

```
lbl_encoder = LabelEncoder()
lbl_encoder.fit(training_labels)
training_labels = lbl_encoder.transform(training_labels)
```

In [4]:
```
vocab_size = 1000
embedding_dim = 16
max_len = 20
oov_token = "<OOV>"

tokenizer = Tokenizer(num_words=vocab_size, oov_token=oov_token)
tokenizer.fit_on_texts(training_sentences)
word_index = tokenizer.word_index
sequences = tokenizer.texts_to_sequences(training_sentences)
padded_sequences = pad_sequences(sequences, truncating='post', maxlen=max_len)
```

In [5]:
```
model = Sequential()
model.add(Embedding(vocab_size, embedding_dim, input_length=max_len))
model.add(GlobalAveragePooling1D())
model.add(Dense(16, activation='relu'))
model.add(Dense(16, activation='relu'))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss='sparse_categorical_crossentropy',
              optimizer='adam', metrics=['accuracy'])

model.summary()
epochs = 500
history = model.fit(padded_sequences, np.array(training_labels), epochs=epochs)
```

```
=================================================================
Total params: 16,697
Trainable params: 16,697
Non-trainable params: 0

_________________________________________________________________
Epoch 1/500
2/2 [==============================] - 1s 8ms/step - loss: 2.1993 - accuracy:
0.0750
Epoch 2/500
2/2 [==============================] - 0s 5ms/step - loss: 2.1975 - accuracy:
0.0250
Epoch 3/500
2/2 [==============================] - 0s 4ms/step - loss: 2.1960 - accuracy:
0.0750
Epoch 4/500

2/2 [==============================] - 0s 4ms/step - loss: 2.1950 - accuracy:
0.1250
Epoch 5/500
2/2 [==============================] - 0s 5ms/step - loss: 2.1942 - accuracy:
```

In [6]:

```
model.save("chat_model")

import pickle

with open('tokenizer.pickle', 'wb') as handle:
    pickle.dump(tokenizer, handle, protocol=pickle.HIGHEST_PROTOCOL)#saving the f

with open('label_encoder.pickle', 'wb') as ecn_file:
    pickle.dump(lbl_encoder, ecn_file, protocol=pickle.HIGHEST_PROTOCOL)#saving t
```

```
INFO:tensorflow:Assets written to: chat_model\assets
```

In [*]:

```
import json
import numpy as np
from tensorflow import keras
from sklearn.preprocessing import LabelEncoder

import colorama
colorama.init()
from colorama import Fore, Style, Back

import random
import pickle

with open("intents.json") as file:
    data = json.load(file)


def chat():
    model = keras.models.load_model('chat_model')# loading trained model

    with open('tokenizer.pickle', 'rb') as handle:
        tokenizer = pickle.load(handle)# loading tokenizer object


    with open('label_encoder.pickle', 'rb') as enc:
        lbl_encoder = pickle.load(enc)# loading label encoder object

    # parameters
    max_len = 20

    while True:
        print(Fore.LIGHTBLUE_EX + "User: " + Style.RESET_ALL, end="")
        inp = input()
        if inp.lower() == "quit":
            break

        result = model.predict(keras.preprocessing.sequence.pad_sequences(tokeniz
                                             truncating='post', maxlen=max_len))
        tag = lbl_encoder.inverse_transform([np.argmax(result)])

        for i in data['intents']:
            if i['tag'] == tag:
                print(Fore.GREEN + "ChatBot:" + Style.RESET_ALL , np.random.choic

print("Start messaging with the bot (type quit to stop)!" + Style.RESET_ALL)
chat()
```

```
Start messaging with the bot (type quit to stop)!
User: who created you
1/1 [==============================] - 0s 193ms/step
ChatBot: Alaissa, Neil and Smith are my team members
User: what is your name
1/1 [==============================] - 0s 49ms/step
ChatBot: I'm Chatbot!
User: i need to create a new account
1/1 [==============================] - 0s 43ms/step
ChatBot: You can just easily create a new account from our web site
```

```
User: have a complaint
1/1 [==============================] - 0s 49ms/step
ChatBot: Please mention your complaint, we will reach you and sorry for any i
nconvenience caused
User:
```

In [ ]: