

**SUMMER INTERNSHIP & TRAINING COURSE**

**June – July 2019**

**Task Report**

**-----------------------------------------------------------------------------------------------------------------------------------------------------**

**Title of the Task :– Chatbot in Python Group No:- 21**

**----------------------------------------------------------------------------------------------------------------------------------------**

**Group Members**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Name** | **GR. No.** | **Div-Roll no.** | **Domain** |
| 1) | Shray Tyagi | 11810615 | E – 57 | Prog |
| 2) | Prashant Kumar | 11810969 | E – 46 | Prog |
|  |  |  |  |  |

**----------------------------------------------------------------------------------------------------------------------------------------**

The Project made by the students in the list above is assessed by us.

Prof. H. M. Khare Prof. Dr. G. N. Kotwal

(Guide & Examiner) (Guide & Examiner)

**TRF Summer internship and training course 2019 Task Report**

# **INDEX**

Group Number: 21

Project Title: Chatbot in Python

**Member 1** **Member 2**

Name: Shray Tyagi Name: Prashant Kumar

Div. & Roll Number: E-57 Div. & Roll Number: E-46

GR Number:11810615 Gr Number: 11810969

E-mail ID: [shray.tyagi18@vit.edu](mailto:shray.tyagi18@vit.edu) E-mail ID: [prashant.yadav18@vit.edu](mailto:prashant.yadav18@vit.edu)

# **INTRODUCTION:**

## **Chatbot–**

* Interacting with the machine via natural language is one of the requirements for general artificial intelligence. This field of AI is called dialogue systems, spoken dialogue systems, or chatbots. The machine needs to provide you with an informative answer, maintain the context of the dialogue, and be indistinguishable from the human (ideally).
* In practice, the last requirement is not reachable yet, but luckily, humans are ready to talk with robots if they are helpful, sometimes funny, and interesting interlocutors.

# **WORKING METHODOLOGY:**

**TASK 3:**

### Algorithm:

* First, we have imported two libraries:

(a)Chatterbot: - This we have used to generate the responses for the input message is given.

(b)Chatterbot.trainers :- This library is used to train our model.

* Then a chatbot is created with a suitable name of our choice.
* The bot is then trained using Chatterbot Corpus Trainer, we have used that type of trainer as it uses large dataset for the training of our model.
* Then file path is given to the chatbot which contains the general conversation between two persons to train and then traversing the file to train our model.
* The model gets saves automatically and now the code that go through our model and gives the output message.
* We then named the bot and specified the trainer.
* Then the main logic of operation is written where the input is taken from the trainer model and responses is given accordingly.

### **WORKING:**

OVERVIEW OF THE CONCEPT USED

**NLP**

* Natural Language Processing, or NLP for short, is broadly defined as the automatic manipulation of natural language, like speech and text, by software.
* Syntax and semantic analysis are two main techniques used with natural language processing.
* Syntax is the arrangement of words in a sentence to make grammatical sense.
* NLP uses syntax to assess meaning from a language based on grammatical rules.
* Syntax techniques used include parsing (grammatical analysis for a sentence), word segmentation (which divides a large piece of text to units), sentence breaking (which places sentence boundaries in large texts), morphological segmentation (which divides words into groups) and stemming (which divides words with inflection in them to root forms).
* Semantics involves the use and meaning behind words. NLP applies algorithms to understand the meaning and structure of sentences.
* Current approaches to NLP are based on deep learning, a type of AI that examines and uses patterns in data to improve a program's understanding.
* Deep learning models require massive amounts of labeled data to train on and identify relevant correlations, and assembling this kind of [big data](https://searchdatamanagement.techtarget.com/definition/big-data) set is one of the main hurdles to NLP currently.
* NLP can be used to interpret free text and make it analyzable. There is a tremendous amount of information stored in free text files, like patients' medical records, for example.
* Before [deep learning](https://searchenterpriseai.techtarget.com/definition/deep-learning-deep-neural-network)-based NLP models, this information was inaccessible to computer-assisted analysis and could not be analyzed in any systematic way. But NLP allows analysts to sift through massive troves of free text to find relevant information in the files.

**Word2Vec**

It is a method to construct such an embedding. It can be obtained using two methods (both involving Neural Networks): Skip Gram and Common Bag of Words (CBOW).

**CBOW Model:**

This method takes the context of each word as the input and tries to predict the word corresponding to the context. Consider our example: Have a great day.

Let the input to the Neural Network be the word, great. Notice that here we are trying to predict a target word (day) using a single context input word great. More specifically, we use the one hot encoding of the input word and measure the output error compared to one hot encoding of the target word (day). In the process of predicting the target word, we learn the vector representation of the target word.

The input or the context word is a one hot encoded vector of size V. The hidden layer contains N neurons and the output is again a V length vector with the elements being the softmax values.

Let’s get the terms in the picture right:  
- Wvn is the weight matrix that maps the input x to the hidden layer (V\*N dimensional matrix)-W`nv is the weight matrix that maps the hidden layer outputs to the final output layer (N\*V dimensional matrix)

The hidden layer neurons just copy the weighted sum of inputs to the next layer. There is no activation like sigmoid, tanh or ReLU. The only non-linearity is the softmax calculations in the output layer.

But, the above model used a single context word to predict the target. We can use multiple context words to do the same.

**🡪WORKING: -**

* Libraries Used: -

## **Chatterbot:**

Chatterbot is a Python library that makes it easy to generate automated responses to a user’s input. Chatterbot uses a selection of machine learning algorithms to produce different types of responses. This makes it easy for developers to create chat bots and automate conversations with users. The ideas and concepts behind Chatterbot see the process flow diagram.

## **Chatterbot.trainers:**

Chatterbot comes with training classes built in, or you can create your own if needed. To use a training class you call train() on an instance that has been initialized with your chat bot.It allows a chat bot to be trained for a conversation.

For example, if you were to run bot of the following training calls, then the resulting chatterbot would respond to both statements of “Hi there!” and “Greetings!” by saying “Hello”.

* bot= ChatBot('Bot') this created the chatbot with a name “Bot”. Your bot is created but at this point your bot has no knowledge, for that you have to train it on some data.
* Also, by default the Chatterbot library will create a SQLite database to build up statements of the chats Chatterbot comes with built in adapter classes that allow it to connect to different types of databases. In this tutorial, we will be using the SQLStorageAdapter which allows the chat bot to connect to SQL databases. By default, this adapter will create a [SQLite](https://www.sqlite.org/) database.
* Now its time to train the model through the specific type of trainer that is ChatterbotCorpusTrainer(<bot\_name>) in which the parameter is the bot we have to train.
* We can train our model through a list of questions and responses line by line but that’s very tedious job.
* It is difficult to train the bot on every single statement. So, we will use ChatterBotCorpusTrainer to train our bot on the large dataset or we can download the dataset of our own language and train our bot in your language
* Then we provide our model with a file that contains general conversation so as to train it.
* And now using a “for loop” we have read that whole file. i.e.

for file in os.listdir(corpus\_path) .

os.listdir() searches for the file by the file path in the OS.

trainer.train() trains the model and the file path is the parameter given to it.

* This model gets save automatically; we don’t need to save this model explicitly.
* Now another code file is written so as to link the model and in which the input is provided and the output responses is given after the code runs through the model created by us to train our model.
* Here we first imported the libraries used earlier and moreover we have used:

**OS: -** The OS module in python provides functions for interacting with the operating system. OS, comes under Python’s standard utility modules. This module provides a portable way of using operating system dependent functionality. The \*os\* and \*os.path\* modules include many functions to interact with the file system.

* We then specified our model and trainer name again in this code to link it with this code.
* Now we given conditional statement so as to provide a specific response if it encounters a given word.
* If it encounters ‘bye’ then the bot should give the same reply, this we have achieved using message.split() method, this function splits the string into subarrays. By this it searches for the given word in the string which is “bye” in our case.

And in another cases the bot should give suitable responses that is written in ‘else block’.

**INSIGHTS:**

Problems Faced:

• Obtaining the data for training the bot. Training an accurate bot requires huge data (5 GB file).

• Finding the right libraries.

• Understanding the working of Word2Vec.

Measures Taken:

• Took limited data for training without compromising with the accuracy.

* Don't train the model every time you run the code. Use the trained model for chatting.

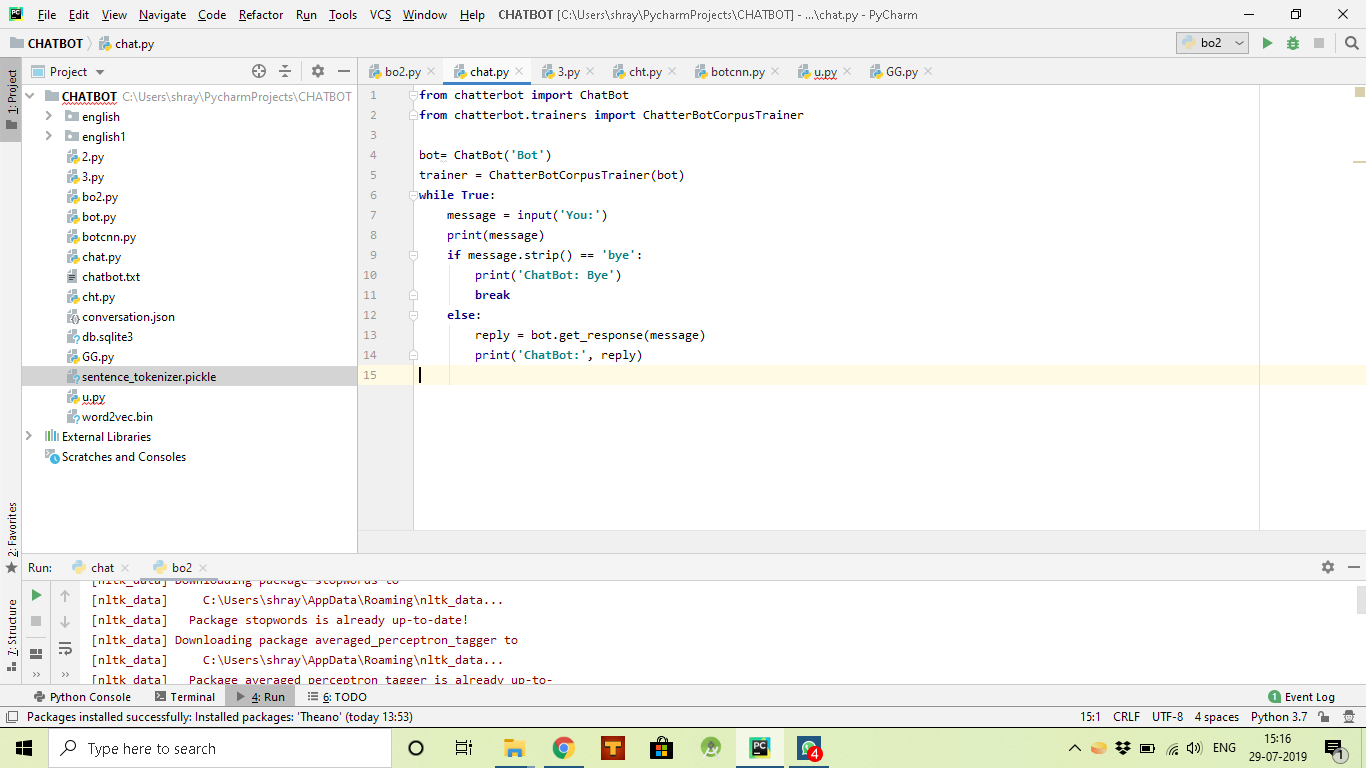
**RESULTS:**

## **INPUT CODE: -**

🡪MODEL

## 

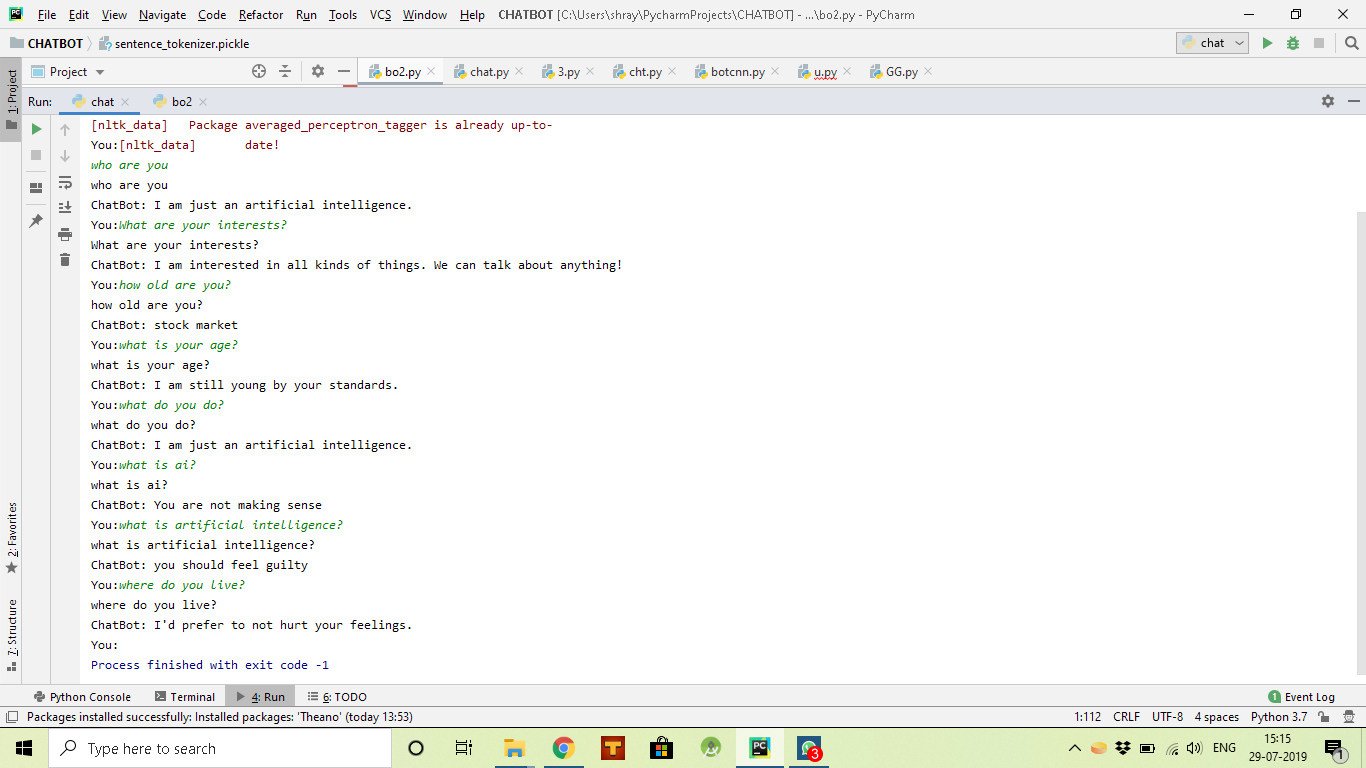
🡪CODE THAT USES THE MODEL TO PREDICT THE OUTPUT



**TRAINING OF OUR MODEL: -**

# 

**OUTPUT: -**



**REFERENCES:**

* [www.google.com](http://www.google.com)
* [www.stackoverflow.com](http://www.stackoverflow.com)
* [www.youtube.com](http://www.youtube.com)
* [www.github.com](http://www.github.com)
* https://chatterbot.readthedocs.io/en/stable/tutorial.html