A Mini Project Report on

CHATBOT USING NLP

Submitted in partial fulfillment of the requirements for the degree of

Fourth Year of Engineering in **Information Technology**

Submitted by

Ashish Jaiswal

Akshay Kalapgar

Mohit Kamble

Guided by

Prof. Nilesh Rathod

MCT
MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI

(Permanently Affiliated to University of Mumbai)
Juhu Versova Link Road, Andheri (West), Mumbai-53

DEPARTMENT OF INFORMATION TECHNOLOGY

UNIVERSITY OF MUMBAI
2020

MCT
MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
(Permanently Affiliated to University of Mumbai)
Juhu Versova Link Road, Andheri (West), Mumbai-53

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

Date:__________

This is to certify that, the project work embodied in this report entitled, "***Chatbot Using NLP***" submitted by "***Ashish Jaiswal*** bearing Roll No. 729", "***Akshay Kalapgar*** bearing Roll No.731", "***Mohit Kamble*** bearing Roll No. 732" for the award of Fourth year of Engineering (***B.E.***)degree in the subject of ***Artificial Intelligence***, is a work carried out by them under my guidance and supervision within the institute. The work described in this project report is carried out by the concerned students and has not been submitted for the award of any other degree of the University of Mumbai.

Further, it is to certify that the students were regular during the academic year 2020- 2021 and have worked under the guidance of concerned faculty until the submission of this project work at ***MCT's Rajiv Gandhi Institute of Technology, Mumbai***.

Mr. Nilesh Rathod
Project Guide

Dr. Sunil B. Wankhade
Head of Department

Dr. Sanjay Bokade
Principal

CERTIFICATE OF APPROVAL

This mini project report entitled

CHATBOT USING NLP

Submitted by:

ASHISH JAISWAL	**729**
AKSHAY KALAPGAR	**731**
MOHIT KAMBLE	**732**

In partial fulfillment of the requirements of the degree **of Fourth year of Engineering** in **Information Technology** is approved.

Internal Examiner

SEAL OF INSTITUTE

____

External Examiner

____

Date:

Place: Mumbai

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that i have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

ROLL NO.	NAME	SIGNATURE
729	Ashish Jaiswal	
731	Akshay Kalapgar	
732	Mohit Kamble	

Date:

Place: Mumbai

ACKNOWLEDGEMENT

With all reverence, we take the opportunity to express our deep sense of gratitude and wholehearted indebtedness to our respected guide, **Mr. Nilesh Rathod**, Department of Information Technology, Rajiv Gandhi Institute of Technology, Mumbai. From the day of conception of this project his active involvement and motivating guidance on day-to-day basis has made it possible for us to complete this challenging work in time.

We would like to express a deep sense of gratitude to our respected **Head of the Department**, **Dr**. **Sunil B. Wankhade** who went all the way out to help us in all genuine cases during the course of doing this project. We wish to express our sincere thanks to **Dr. Sanjay Bokade**, **Principal**, Rajiv Gandhi Institute of Technology, Mumbai and would to like to acknowledge specifically for giving guidance, encouragement and inspiration throughout the academics.

We would like to thank all the staff of Information Technology Department who continuously supported and motivated during our work. Also, we would like to thank our colleagues for their continuous support and motivation during the project work. Finally, we would like to express our gratitude to our family for their eternal belief in us. We would not be where we are today without their support and encouragement.

ASHISH JAISWAL

AKSHAY KALAPGAR

MOHIT KAMBLE

Date:

Place:

Abstract

A chatbot is Artificial Intelligence (AI) software that can simulate a conversation (or a chat) with a user in natural language through messaging applications, websites, and mobile apps or through the telephone.

It is often described as one of the most advanced and promising expressions of interaction between humans and machines. However, from a technological point of view, a chatbot only represents the natural evolution of a Question Answering system leveraging Natural Language Processing (NLP). Formulating responses to questions in natural language is one of the most typical Examples of Natural Language Processing applied in various enterprises' end-use applications.

Chatbot applications streamline interactions between people and services, enhancing customer experience. At the same time, they offer companies new opportunities to improve the customers engagement process and operational efficiency by reducing the typical cost of customer service.

To be successful, a chatbot solution should be able to effectively perform both of these tasks. Human support plays a key role here: Regardless of the kind of approach and the platform, human intervention is crucial in configuring, training and optimizing the chatbot system.

Contents

Table of Contents

Chapters	Title of chapters	Page No.
Chapter 1	Introduction	1
Chapter 2	Aims and Objectives	2
Chapter 3	Literature Surveyed	3
Chapter 4	Existing System	4
Chapter 5	Problem Statement	5
Chapter 6	Scope	6
Chapter 7	Proposed System	7
Chapter 8	Methodology	8
Chapter 9	Analysis	10
Chapter 10	Details of Hardware and Software	11
Chapter 11	Working	12
Chapter 12	References	21

Table of Figures

Fig No.	List of Figures	Page No.
1	Process flow Diagram	8

CHAPTER 1
INTRODUCTION

The development of Artificial Intelligence applications is challenging because computers traditionally require humans to speak to them in a programming language that is precise, unambiguous and highly structured or, perhaps through a limited number of clearly-stated voice commands.

Natural language processing (NLP) is a branch of artificial intelligence, and machine linguistics that enables computers to derive meaning from human or natural language input. It is used to analyze text, allowing machines to understand human's language. NLP considers the hierarchical structure of human language in which several words make a phrase; several phrases make a sentence and, ultimately, sentences convey ideas. However, the ambiguity of language in which humans speak is what makes natural language processing a difficult problem for computers to undertake.

The computer performs Natural Language Understanding (NLG) to overcome this obstacle. It is the process of disassembling and parsing input because of the occurrence of unknown and unexpected components in the input and the need to determine the appropriate syntactic and semantic schemes to apply to it. A Chatbot is a computer program which conducts conversation with human using auditory or textual methods. Chatbots are based on two basic principles Natural Language Processing and Pattern matching. We aim towards creating a conversational Chatbot with the help of NLP as well as pattern matching.

CHAPTER 2

AIMS AND OBJECTIVES

Aims:

We aim towards creating a conversational Chatbot with the help of NLP as well as pattern matching.

Objectives:

- The main objectives of the project were to develop an algorithm that will be used to identify answers related to user submitted questions.
- To develop a database were all the related data will be stored and to develop a web interface.
- To provide user friendly interface
- To reduce user efforts

CHAPTER 3
LITERATURE SURVEY

Paper 1: "Emassnuela Ha Simulates an Historical Figure"

Authors: Dr Raju Shanmugam, Soumya Ranjan Jena.

Publisher: IEEE Conference Publications, July 2013.

Observations:

- There may be applications that are incorporating man-like appearance and intending to imitate human, but in most of the scenarios the information of the conversational bot is stored in a db created by someone who has prolonged knowledge in that field.
- However, few experts may have investigated the idea of creating a Chat Bot using an artificial character and personality beginning from web-pages or plain-text of a certain person.
- The paper elaborates the idea of pointing out the important facts in texts explaining the life of a ancient figure for creating an agent that is used in school scenarios.

Paper 2: "Teaching Introductory Artificial Intelligence Using A simple Agent Framework"

Authors: Maja Pantic, Reinier Zwitserloot, and Robbert Jan Grootjans

Publisher: IEEE Transactions On Education, Vol. 48, No. 3, August 2005.

Observations:

- The paper explain a way of teaching artificial intelligence (AI) using a genuine, naive agent frameworks only for this course.
- Though many agent frameworks have been proposed in the literature, none of the available structures was easy or simple to be used by to be graduates of CSE.
- The main objective of using such a study was to keep busy the students into which they found very interesting.
- A constructive approach and a traditional approach was used so that students learn very effectively.

CHAPTER 4

EXISTING SYSTEM

There are many existing chatbots available in every area. Almost all the popular and active websites have chatbots. The chatbot is run manually by the database. It is difficult to run by the man power because at a time many customers ask questions so man power is difficult to answers the questions to each one in a short time, so therefore chatbot comes in study. The user has to manually ask the questions and thar chatbot database provides the answer so it is quite time consuming.

Drawbacks of Existing System

- It is difficult to run by man power
- System is run manually by the database
- Difficult to respond every user within short time

CHAPTER 5

PROBLEM STATEMENT

When one has to make choice out of an extensive alternative, one may face decision paralysis in which in unable to make decision and even opt out of the decision. Artificial intelligence chatbot is a technology that makes interactions between man and machines using natural language possible. From literature, we found out that in general, chatbot are functions like a typical search engine. Although chatbot just produced only one output instead of multiple outputs/results, the basic process flow is the same where each time an input is entered, the new search will be done. Nothing related to previous output.

This research is focused on enabling chatbot to become a search engine that can process the next search with the relation to the previous search output. In chatbot context, this functionality will enhance the capability of chatbot’s input processing.

CHAPTER 6

SCOPE

The future work include training the chatbot with more varied data; increasing the scope of the chatbot by adding a speech recognition feature so that users can speak to get responses.

The scopes of the application are:

- The system was partially successful in adding empathy since scope of these queries is vast and the system requires more rigorous data to handle all the questions which are out of script.

- To improve the current functionalities of Chatbot, in the future, the scope of the chatbot can be increased by inserting data for all the departments, training the bot with varied data, testing it on live website, and based on that feedback inserting more training data to the bot.

CHAPTER 7

PROPOSED SYSTEM

As we know the use of automated machines is increasing day by day i.e., the machine-human interaction is also increasing day by day. The case might as well come true where one day humans would be replaced by automated bots for performing specific functioning of the system. Where, a central system will be a chatbot which will receive the input from the user which will be in the form of an audio/ voice. The input can be received using a mic and then this voice input will be converted to the corresponding textual input using preexisting python libraries. Then this textual input will be provided to the system which consists of the chatterbot which will perform Natural Language Processing on the provided input and search within its database for the query.

Two cases will be generated:

(1) When the input entered is an identifiable query, to which already an answer is known to the system. This corresponding answer will be directly provided to the user in this case.

(2) When the input entered is not recognized, this is where the intelligence of the bot comes into play and the system will understand the input and answer it using its intelligence. This is termed as “Artificial Intelligence”.

CHAPTER 8

METHODOLOGY

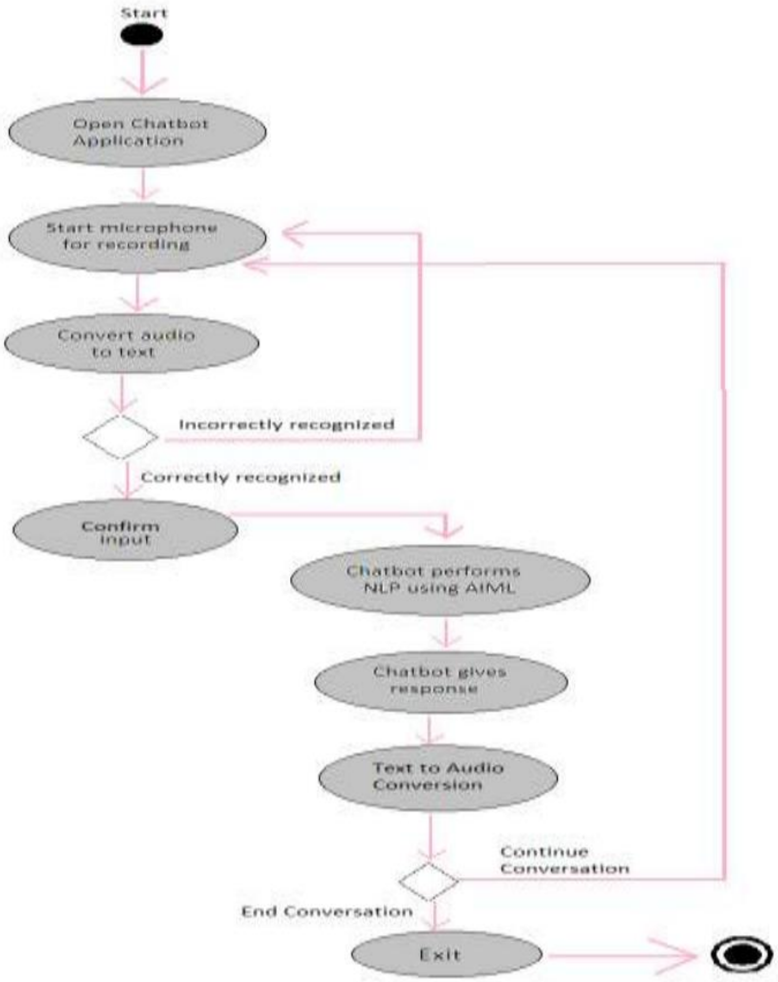


Figure No. 2: System Flowchart

The following methodology will be used for working of the system:

- Once the system knows what output is to be given to the user, the output is generated which is in a textual format.
- This textual format is then converted to voice using pre-existing libraries and a voice note is generated. This voice clip will be given to the user as an output.

CHAPTER 9

ANALAYSIS

Advantages:

- Reduce searching time
- Available 24 hrs
- Support availability anytime
- The application will provide better management

CHAPTER 10

DETAILS OF HARDWARE AND SOFTWARE

Hardware Requirements:

- PC with following Specifications:

1. 8 GBRAM
2. 64-bit Processor
3. Hard disk— 320 GB or above
4. USB Cable (debugging)

Software Requirements:

Google Collaboratory:

Colab is a free Jupyter notebook environment that runs entirely in the cloud. Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by your team members - just the way you edit documents in Google Docs. Colab supports many popular machine learning libraries which can be easily loaded in your notebook.

Google colab can be used to:

- Write and execute code in Python
- Document your code that supports mathematical equations
- Create/Upload/Share notebooks
- Import/Save notebooks from/to Google Drive
- Import/Publish notebooks from GitHub
- Import external datasets e.g. from Kaggle
- Integrate PyTorch, TensorFlow, Keras, OpenCV
- Free Cloud service with free GPU

CHAPTER 11

WORKING

Code:

intents.json

```
{
  "intents": [
    {
      "tag": "greeting",
      "patterns": [
        "Hi",
        "Hey",
        "How are you",
        "Is anyone there?",
        "Hello",
        "Good day"
      ],
      "responses": [
        "Hey :-)",
        "Hello, thanks for visiting",
        "Hi there, what can I do for you?",
        "Hi there, how can I help?"
      ]
    },
    {
      "tag": "goodbye",
      "patterns": ["Bye", "See you later", "Goodbye"],
      "responses": [
        "See you later, thanks for visiting",
        "Have a nice day",
        "Bye! Come back again soon."
      ]
    },
    {
      "tag": "thanks",
      "patterns": ["Thanks", "Thank you", "That's helpful", "Thank's a lot!"],
      "responses": ["Happy to help!", "Any time!", "My pleasure"]
    },
    {
      "tag": "items",
      "patterns": [
        "Which items do you have?",
        "What kinds of items are there?",
        "What do you sell?"
      ],
      "responses": [
        "We sell coffee and tea",
        "We have coffee and tea"
      ]
    },
    {
      "tag": "payments",
      "patterns": [
        "Do you take credit cards?",
        "Do you accept Mastercard?",
        "Can I pay with Paypal?",
        "Are you cash only?"
      ],
      "responses": [
        "We accept VISA, Mastercard and Paypal",
        "We accept most major credit cards, and Paypal"
      ]
    },
    {
      "tag": "delivery",
      "patterns": [
        "How long does delivery take?",
        "How long does shipping take?",
        "When do I get my delivery?"
      ],
      "responses": [
        "Delivery takes 2-4 days",
        "Shipping takes 2-4 days"
      ]
    },
    {
      "tag": "funny",
      "patterns": [
        "Tell me a joke!",
        "Tell me something funny!",
        "Do you know a joke?"
      ],
      "responses": [
        "Why did the hipster burn his mouth? He drank the coffee before it was cool.",
        "What did the buffalo say when his son left for college? Bison."
      ]
    }
  ]
}
```

nltk_utils.py

```
import numpy as np
import nltk
#nltk.download('punkt')
from nltk.stem.porter import PorterStemmer
stemmer = PorterStemmer()

def tokenize(sentence):
    """
    split sentence into array of words/tokens
    a token can be a word or punctuation character, or number
    """
    return nltk.word_tokenize(sentence)


def stem(word):
    """
    stemming = find the root form of the word
    examples:
    words = ["organize", "organizes", "organizing"]
    words = [stem(w) for w in words]
    -> ["organ", "organ", "organ"]
    """
    return stemmer.stem(word.lower())


def bag_of_words(tokenized_sentence, words):
    """
    return bag of words array:
    1 for each known word that exists in the sentence, 0 otherwise
    example:
    sentence = ["hello", "how", "are", "you"]
    words = ["hi", "hello", "I", "you", "bye", "thank", "cool"]
    bog   = [  0 ,    1 ,    0 ,   1 ,    0 ,    0 ,      0]
    """
    # stem each word
    sentence_words = [stem(word) for word in tokenized_sentence]
    # initialize bag with 0 for each word
    bag = np.zeros(len(words), dtype=np.float32)
    for idx, w in enumerate(words):
        if w in sentence_words:
            bag[idx] = 1

    return bag
```

train.py

```
import numpy as np
import random
import json

import torch
import torch.nn as nn
from torch.utils.data import Dataset, DataLoader

from nltk_utils import bag_of_words, tokenize, stem
from model import NeuralNet

with open('intents.json', 'r') as f:
    intents = json.load(f)

all_words = []
tags = []
xy = []
# loop through each sentence in our intents patterns
for intent in intents['intents']:
    tag = intent['tag']
    # add to tag list
    tags.append(tag)
    for pattern in intent['patterns']:
        # tokenize each word in the sentence
        w = tokenize(pattern)
        # add to our words list
        all_words.extend(w)
        # add to xy pair
        xy.append((w, tag))

# stem and lower each word
ignore_words = ['?', '.', '!']
all_words = [stem(w) for w in all_words if w not in ignore_words]
# remove duplicates and sort
all_words = sorted(set(all_words))
tags = sorted(set(tags))

print(len(xy), "patterns")
print(len(tags), "tags:", tags)
print(len(all_words), "unique stemmed words:", all_words)

# create training data
```

```
X_train = []
y_train = []
for (pattern_sentence, tag) in xy:
    # X: bag of words for each pattern_sentence
    bag = bag_of_words(pattern_sentence, all_words)
    X_train.append(bag)
    # y: PyTorch CrossEntropyLoss needs only class labels, not one-hot
    label = tags.index(tag)
    y_train.append(label)

X_train = np.array(X_train)
y_train = np.array(y_train)

# Hyper-parameters
num_epochs = 1000
batch_size = 8
learning_rate = 0.001
input_size = len(X_train[0])
hidden_size = 8
output_size = len(tags)
print(input_size, output_size)

class ChatDataset(Dataset):

    def__init__(self):
        self.n_samples = len(X_train)
        self.x_data = X_train
        self.y_data = y_train

    # support indexing such that dataset[i] can be used to get i-th sample
    def_getitem_(self, index):
        return self.x_data[index], self.y_data[index]

    # we can call len(dataset) to return the size
    def_len_(self):
        return self.n_samples

dataset = ChatDataset()
train_loader = DataLoader(dataset=dataset,
                          batch_size=batch_size,
                          shuffle=True,
                          num_workers=0)

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

model = NeuralNet(input_size, hidden_size, output_size).to(device)

# Loss and optimizer
```

```
criterion = nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate)

# Train the model
for epoch in range(num_epochs):
    for (words, labels) in train_loader:
        words = words.to(device)
        labels = labels.to(dtype=torch.long).to(device)

        # Forward pass
        outputs = model(words)
        # if y would be one-hot, we must apply
        # labels = torch.max(labels, 1)[1]
        loss = criterion(outputs, labels)

        # Backward and optimize
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

    if (epoch+1) % 100 == 0:
        print (f'Epoch [{epoch+1}/{num_epochs}], Loss: {loss.item():.4f}')


print(f'final loss: {loss.item():.4f}')

data = {
"model_state": model.state_dict(),
"input_size": input_size,
"hidden_size": hidden_size,
"output_size": output_size,
"all_words": all_words,
"tags": tags
}

FILE = "data.pth"
torch.save(data, FILE)

print(f'training complete. file saved to {FILE}')
```

model.py

```
import torch
import torch.nn as nn


class NeuralNet(nn.Module):
    def_init_(self, input_size, hidden_size, num_classes):
        super(NeuralNet, self)._init_()
        self.l1 = nn.Linear(input_size, hidden_size)
        self.l2 = nn.Linear(hidden_size, hidden_size)
        self.l3 = nn.Linear(hidden_size, num_classes)
        self.relu = nn.ReLU()

    def forward(self, x):
        out = self.l1(x)
        out = self.relu(out)
        out = self.l2(out)
        out = self.relu(out)
        out = self.l3(out)
        # no activation and no softmax at the end
        return out
```

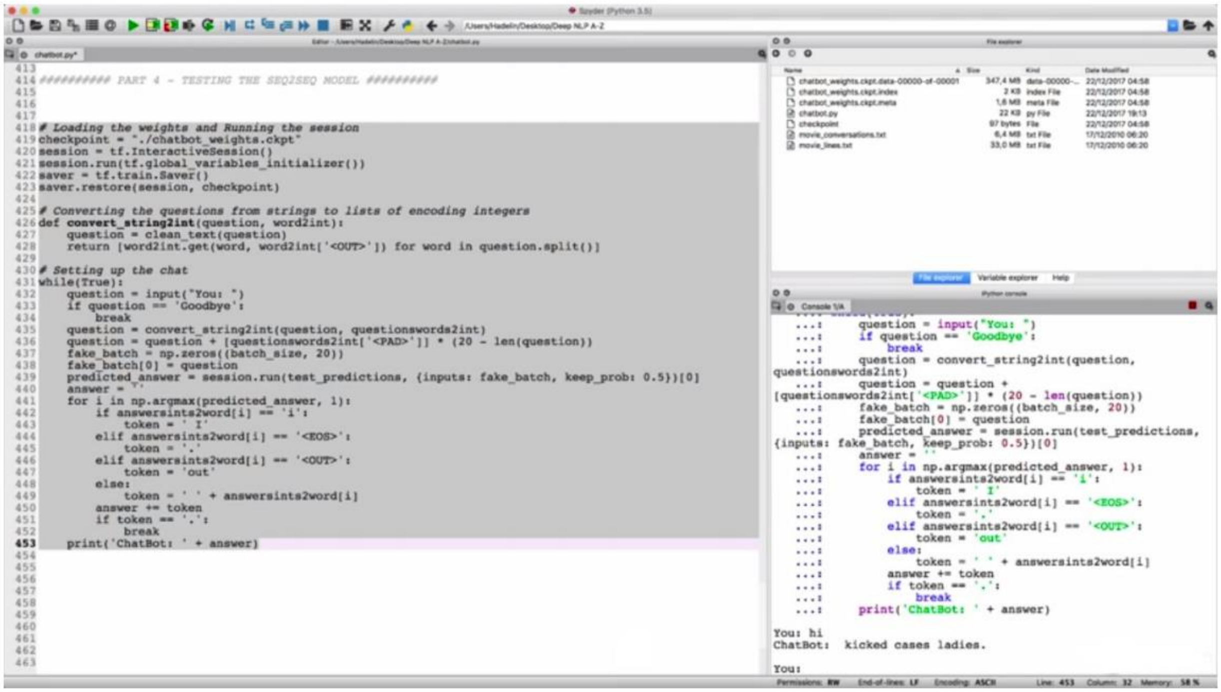
chat.py

```
import random
import json

import torch

from model import NeuralNet
from nltk_utils import bag_of_words, tokenize

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

with open('intents.json', 'r') as json_data:
    intents = json.load(json_data)

FILE = "data.pth"
data = torch.load(FILE)

input_size = data["input_size"]
hidden_size = data["hidden_size"]
output_size = data["output_size"]
all_words = data['all_words']
```

```
tags = data['tags']
model_state = data["model_state"]

model = NeuralNet(input_size, hidden_size, output_size).to(device)
model.load_state_dict(model_state)
model.eval()

bot_name = "Sam"
print("Let's chat! (type 'quit' to exit)")
while True:
    # sentence = "do you use credit cards?"
    sentence = input("You: ")
    if sentence == "quit":
        break

    sentence = tokenize(sentence)
    X = bag_of_words(sentence, all_words)
    X = X.reshape(1, X.shape[0])
    X = torch.from_numpy(X).to(device)

    output = model(X)
    _, predicted = torch.max(output, dim=1)

    tag = tags[predicted.item()]

    probs = torch.softmax(output, dim=1)
    prob = probs[0][predicted.item()]
    if prob.item() > 0.75:
        for intent in intents['intents']:
            if tag == intent["tag"]:
                print(f"{bot_name}: {random.choice(intent['responses'])}")
    else:
        print(f"{bot_name}: I do not understand...")
```

Output:

- The result of the chatbot that we have implemented is shown below.
- The Chatbot has been trained using seq2seq architecture and has been tested.



```python
########## PART 4 - TESTING THE SEQ2SEQ MODEL ##########



# Loading the weights and Running the session
checkpoint = "./chatbot_weights.ckpt"
session = tf.InteractiveSession()
session.run(tf.global_variables_initializer())
saver = tf.train.Saver()
saver.restore(session, checkpoint)

# Converting the questions from strings to lists of encoding integers
def convert_string2int(question, word2int):
    question = clean_text(question)
    return [word2int.get(word, word2int['<OUT>']) for word in question.split()]

# Setting up the chat
while(True):
    question = input("You: ")
    if question == 'Goodbye':
        break
    question = convert_string2int(question, questionswords2int)
    question = question + [questionswords2int['<PAD>']] * (20 - len(question))
    fake_batch = np.zeros((batch_size, 20))
    fake_batch[0] = question
    predicted_answer = session.run(test_predictions, {inputs: fake_batch, keep_prob: 0.5})[0]
    answer = ''
    for i in np.argmax(predicted_answer, 1):
        if answersints2word[i] == 'i':
            token = ' I'
        elif answersints2word[i] == '<EOS>':
            token = '.'
        elif answersints2word[i] == '<OUT>':
            token = 'out'
        else:
            token = ' ' + answersints2word[i]
        answer += token
        if token == '.':
            break
    print('ChatBot: ' + answer)
```

Name	Size	Kind	Date Modified
chatbot_weights.ckpt.data-00000-of-00001	347,4 MB	data-00000-...	22/12/2017 04:58
chatbot_weights.ckpt.index	2 KB	index File	22/12/2017 04:58
chatbot_weights.ckpt.meta	1,6 MB	meta File	22/12/2017 04:58
chatbot.py	22 KB	py File	22/12/2017 19:13
checkpoint	97 bytes	File	22/12/2017 04:58
movie_conversations.txt	6,4 MB	txt File	17/12/2010 06:20
movie_lines.txt	33,0 MB	txt File	17/12/2010 06:20

```
   ...:     question = input("You: ")
   ...:     if question == 'Goodbye':
   ...:         break
   ...:     question = convert_string2int(question,
questionswords2int)
   ...:     question = question +
[questionswords2int['<PAD>']] * (20 - len(question))
   ...:     fake_batch = np.zeros((batch_size, 20))
   ...:     fake_batch[0] = question
   ...:     predicted_answer = session.run(test_predictions,
{inputs: fake_batch, keep_prob: 0.5})[0]
   ...:     answer = ''
   ...:     for i in np.argmax(predicted_answer, 1):
   ...:         if answersints2word[i] == 'i':
   ...:             token = ' I'
   ...:         elif answersints2word[i] == '<EOS>':
   ...:             token = '.'
   ...:         elif answersints2word[i] == '<OUT>':
   ...:             token = 'out'
   ...:         else:
   ...:             token = ' ' + answersints2word[i]
   ...:         answer += token
   ...:         if token == '.':
   ...:             break
   ...:     print('ChatBot: ' + answer)

You: hi
ChatBot:  kicked cases ladies.

You:
```

```
Permissions: RW   End-of-lines: LF   Encoding: ASCII   Line: 453   Column: 32   Memory: 58 %
```

CHAPTER 12

REFERENCES

1. https://link.springer.com/chapter/10.1007/978-981-13-8581-0_7
2. https://www.hurekatek.com/blog/uses-of-chatbots.
3. https://thescipub.com/abstract/jcssp.2010.1212.1218#:~:text=Problem%20statement%3A%20Artificial%20intelligence%20chatbot,machines%20using%20natural%20language%20possible.&text=Chatbot%20can%20give%20different%20responses,according%20to%20current%20conversation%20issue.