Project
Report

A Chatbot Using Natural Language Processing

Submitted in fulfillment of the requirements for the

Minor Project

Submitted by

Roll No	Name of Student
13107020	Priyanshu Jain

Under the guidance of

Mr. Deepak kant Netam
Assistant Professor

Department of Information Technology and Engineering
Institute of Technology, Guru Ghasidas Vishwavidalaya
Bilaspur, Chattisgarh, India – 495009

2015-2016

DECLARATION

I, hereby declare that the work presented in this dissertation entitled 'A Chatbot Using Natural Language Processing' has been done by me, and this dissertation embodies my own work.

Priyanshu Jain

Date:

Department of Information Technology,
Institute of Technology,
Guru Ghasidas Vishwavidyalaya, Bilaspur
(Central University)

Certificate

This is to certify that this is a bonafide record of the project entitled 'A Chatbot Using Natural Language Processing' presented by Priyanshu Jain during 6th Semester in fulfilment of the requirements of the Minor Project of Bachelor of Technology in Information Technology and Engineering.

Mr. Deepak kant Netam
Assistant Professor
(Project Guide)

Dr. Amit Khaskalam
Assistant Professor
(Head of Department)

Date:

DEPARTMENT OF INFORMATION TECHNOLOGY,
INSTITUTE OF TECHNOLOGY,
GURU GHASIDAS VISHWAVIDYALAYA, BILASPUR
(CENTRAL UNIVERSITY)

Certificate by Examiners

The project entitled 'A CHATBOT USING NATURAL LANGUAGE PROCESSING' submitted by *Priyanshu Jain, Roll No: 13107020* has been examined by the undersigned as a part of the examination and fulfills the requirement of *Department of Information Technology, Guru Ghasidas Vishwavidyalaya, Bilaspur.*

Examiner 2

Examiner 1

ACKNOWLEDGEMENT

This project aims for Chatbot using Natural Language Processing. The project scope was very vast, but I've tried to it to the best of my ability and knowledge.
I would like to thank a few people who have helped me for this project.

- My guide, *Mr. Deepak kant Netam* for directing me as to how to go about the project and helping me in figuring out different approaches for the problem at hand.
- *Dr. Amit Khaskalam*, Head of Information Technology Department, for giving us the opportunity to work on this project.
- All the esteemed faculty members, *Mr. Rajesh Mahule, Mr. Santosh Soni, Mr. Abhishek Jain, Mr. Agnivesh Pandey, Mr. Pankaj Chandra, Mr. Suhel Ahamed, Mr. Anand Rawal and Ms. Akanksha Gupta* for their continued support and effort.
- The director of "Institute of Technology", *Prof. Shailendra Kumar*, for providing all the necessary facilities.
- The dean of "Institute of Technology", *Prof. Mukesh Singh*, for providing constant support to students.
- My *friends*, for all the fruitful discussions on this topic, which proved very helpful.
- Last but not the least, my *parents*, for their constant support and encouragement.

I would like to thank all those who spared their valuable time to discuss and helped me present the project in it's current shape and form.

Abstract

Chatbots are computer programs that interact with users using natural languages. Just as people use language for human communication, chatbots use natural language to communicate with human users. In this project, we begin by introducing chatbots and emphasize their need in day-to-day activities. Then we go on to discuss existing chatbot systems, namely ELIZA. We evaluate what we can take from each of the systems and use in our proposed system. Finally, we discuss the proposed system. The system we intend to develop can be used for single human chatting. The database will be fed with related data which will be coded using Python. My implementation is based on one originally written by Joe Strout. I have updated it significantly to use a more modern and idiomatic form of Python This system can be used in operating systems in a similar manner to Siri for effective information retrieval just by telling various queries.
Index Terms– chatbot, natural language processing, Eliza.

Contents

List of Figures

Chapter 1

Problem Definition

1.1 Need

Need of chatbot?

- Widespread use of personal machines.
- Better Human Computer Interaction.

1.2 Goals

What are the goals?

- Make a conversational agent that interacts with users using natural language.
- Started as an attempt to fool humans.

Chapter 2

Introduction

What is chatbot?

Chatbot stands for chatter robot. It is a computer program which has the ability to chat with human users. Audio or text can be used as medium of communication. The primary aim of writing these conversational programs is to have a powerful tool that chat in such a way that the human user cannot realize that he is talking to a software program. Chatbot programs are also referred to as Artificial Conversational Entities, talk bots, chatterbots and chatterboxes.

2.1 Background and Recent Research

Where is the state of art?

2.1.1 Literature Survey

There are many types of Chat Bots, all with varying levels of complexity and strategies for tricking a human that they are not speaking to a computer.

ELIZA being the first and most well known created in 1966, would play the part of a doctor and continue to push back the conversation to the user by asking how previous responses would make one feel by trying to dig deeper to the root of the problem. It was clever and also simple.

CleverBot was brought online in 1997. It was designed to reply with input from some point from the past. Today, there are several million replies across many topics and domains. The input is sometimes on topic and relevant, and other times wildly way off topic. This approach is much like search results and without knowing the reason for asking the question in the first place, is it hard to provide accurate results.

2.1.2 Current scenario

After reviewing the current landscape, it lead me to believe that fooling a human into thinking a bot is real was not because the bot was great, rather it employing silly tactics like deliberating using broken english, typos and deflection.

The chatbot heyday, to the extent that it had one, peaked at the dawn of the Internet and with AOL chatrooms. Chatbots surfaced in programs, like Microsoft Word, which had Clippy, the virtual assistant. But chatbots never proved more useful than annoying and mostly faded, until today. Messaging apps like Facebook Messenger, Kik, Snapchat, WeChat, Viber, Skype and Line will collectively reach 2 billion users in the next two years. And chatbots are seen as one way for big brands and publishers to reach these users, one on one.

Eugene Goostman made the news a few months back as being turing complete which created controversy. This bot employs several tricks such as pretending to be a 13-year-old boy whos english is his second language.

Chat bots have to be great at answering questions, this is usually how they are challenged, and IBMs Watson is probably the best question and answer system. However unlike Watson, we dont need to be 100 % accurate when it comes to answering questions.We can significantly reduce the knowledge base that drives the bot, after all, nobody likes a no-it-all.

2.2 Motivation

Human interaction has always fascinated me: social awkwardness, communication style, how knowledge is transferred, how relationships are built around trust, story telling and knowledge exchange.

What if a machine invoked an emotional response?

First the back story

I want to write about a project I have been working on, and how it has engulfed the last few years of my life, but ultimately, this post is about creating a real chat bot.

Work to date has been an extension of the work done in NLP (natural language process), currently a collection of tools now forming the foundation of Node/Natural.

My initial goal was to build a IBM Watson type clone; something that could parse input in some form and sort out candidates from a data source.

I did some research in language and how communication is affected. For example people who are tired, or introverted tend to give shorter answers when conversing. I spent a fair amount of time thinking about how this would play into a Chat Bot.

Chapter 3

Work Done

3.1 Toolkit

3.1.1 NLTK

NLTK has been called a wonderful tool for teaching, and working in, computational linguistics using Python, and an amazing library to play with natural language.

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries.

3.2 Python packages

3.2.1 random

This module implements pseudo-random number generators for various distributions.

For integers, uniform selection from a range. For sequences, uniform selection of a random element, a function to generate a random permutation of a list in-place, and a function for random sampling without replacement.

3.2.2 re

This module provides regular expression matching operations similar to those found in Perl. Both patterns and strings to be searched can be Unicode strings as well as 8-bit strings.

Regular expressions use the backslash character ('\') to indicate special forms or to allow special characters to be used without invoking their special meaning. This collides with Pythons usage of the same character for the same purpose in string literals; for example, to match a literal backslash, one might have to write '\\\\' as the pattern string, because the regular expression must be \\, and each backslash must be expressed as \\ inside a regular Python string literal.

The solution is to use Pythons raw string notation for regular expression patterns; backslashes are not handled in any special way in a string literal prefixed with 'r'. So r"\n" is a two-character string containing '\' and 'n', while "\n" is a one-character string containing a newline. Usually patterns will be expressed in Python code using this raw string notation.

3.3 Implementation

Since there have been various implementations, more or less similar to the original one. Emacs ships with an ELIZA-type program built in. The CIA even experimented with computer-aided interrogation of officers using a very similar, but rather more combative, version of the program.

My implementation is based on one originally written by Joe Strout. I have updated it significantly to use a more modern and idiomatic form of Python, but the text patterns and data structures are copied essentially verbatim.

This is fundamentally a pattern matching program, in this most of the source code is taken up by a dictionary called and a list of lists .

Dictionary maps first-person pronouns to second-person pronouns and vice-versa. It is used to reflect a statement back against the user.

Pairs is made up of a list of lists where the first element is a regular expression that matches the users statements and the second element is a list of potential responses. Many of the potential responses contain placeholders that can be filled in with fragments to echo the users statements.

main is the entry point of the program. Lets take a closer look at it.

```
def main():
    print "Hello. How are you feeling today?"

    while True:
        statement = raw_input("> ")
        print analyze(statement)

        if statement == "quit":
            break
```

First, we print the initial prompt, then we enter a loop of asking the user for input and passing what the user says to the analyze function to get the therapists response. If at any point the user types quit, we break out of the loop and the program exits.

Lets see whats going on in analyze .

```
def analyze(statement):
    for pattern, responses in psychobabble:
        match = re.match(pattern, statement.rstrip(".!"))
        if match:
            response = random.choice(responses)
            return response.format(*[reflect(g) for g in match.groups(
```

We iterate through the regular expressions in the psychobabble array, trying to match each one with the users statement, from which we have stripped the final punctuation. If we find a match, we choose a response template randomly from the list of possible responses associated with the matching pattern. Then we interpolate the match groups from the regular expression into the response string, calling the reflect function on each match group first.

There is one syntactic oddity to note here. When we use the list comprehension to generate a list of reflected match groups, we explode the list with the asterisk (*) character before passing it to the strings format method. Format expects a series of positional arguments corresponding to the number of format placeholders 0, 1, etc. in the string. A list or a tuple can be exploded into positional arguments using a single asterisk. Double asterisks (**) can be used to explode dictionaries into keyword arguments.

Now lets examine the reflect function.

```
def reflect(fragment):
    tokens = fragment.lower().split()
    for i, token in enumerate(tokens):
        if token in reflections:
            tokens[i] = reflections[token]
    return ' '.join(tokens)
```

There is nothing too complicated going on in it. First, we make the statement lowercase, then we tokenize it by splitting on whitespace characters. We iterate through the list of tokens and, if the token exists in our reflections dictionary, we replace it with the value from the dictionary. So I becomes you, your becomes my, etc.

This was all for one type of chatbot. We included in this chatbot five datasets consisting rudechat, elizachat, casualchat, warchat, spiritualchat. Then we randomized the result of all five using random module randint function.

```
def main():

  while True:
        statement = raw_input("> ")
        x = random.randint(1,5)
        if x == 1 :
            print eliza_chat(statement)

        elif x == 2:
            print rude_chat(statement)

        elif x == 3 :
            print suntsu_chat(statement)

        elif x == 4:
```

```
        print iesha_chat(statement)

    else:
        print zen_chat(statement)

    if statement == "quit":
        break
```

This deviates mood of person who is chatting and able to answer more questions.

3.4 Test

Sample chating to our bot is the following.

1.

```
==========================================================================

Talk to the program by typing in plain English, using normal upper-
and lower-case letters and punctuation. Enter "quit" when done.
==========================================================================
Hello. How are you feeling today?
> good
Form is emptiness, and emptiness is form.
> why
Very interesting.
> still interesting
Random talk is useless.
> I want that
It is the rule in war.
> good to talk you
I see. And what does that tell you?
> information
One may know how to conquer without being able to do it.
> I see
Can you elaborate on that?
> no
Very interesting.
>
```

2.

```
===============================================================================

Talk to the program by typing in plain English , using normal upper−
and lower−case letters and punctuation .  Enter "quit" when done .
===============================================================================
Hello .  How are you feeling today?
> not good
I 'm getting bored here . Become more interesting .
> How I can
How do you suppose?
> I can suppose by your answers
One may know how to conquer without being able to do it .
> yes
How do you feel when you say that?
> bad
do u like anime??
> yes
I 'm getting bored here . Become more interesting .
> quit
Good−bye .
```

During test we evaluated that this machine can hold up some time until the machine's true lack of understanding became apparent.
So main target of chatbot is attempt to fool humans, and that is fulfilling at some extent.
Clearly the results will improve with the quality and quantity of training data and the sophistication of heuristic rules. Therefore, over time I hope to add to the amount of statistical training data as well as develop a more comprehensive set of pattern matching rules.

Chapter 4

Issues

There are many issues with this model. It has limited knowledge base from which it answers the questions asked to it. This model requires hand crafted rules. Model does not remember facts and previous answers. This is not a cold, flawlessly accurate machine. And here sentences are not as much parsed means Each one is chunked and broken into separate message objects for the bot to interpret.

Here are specific test set of question on which this bot fails to answer appopriately.

```
My name is Bill. What is your name?
How many letters are there in the name Bill?
How many letters are there in my name?
Which is larger, an apple or a watermelon?
How much is 3 + 2?
How much is three plus two?
What is my name?
 If John is taller than Mary, who is the shorter?
 If it were 3:15 AM now,
what time would it be in 60 minutes?
My friend John likes to fish for trout.
What does John like to fish for?
What number comes after seventeen?
What is the name of my friend who fishes for trout?
What whould I use to put a nail into a wall?
What is the 3rd letter in the alphabet?
What time is it now?
```

When Asking questions for several minutes machine's true lack of understanding became apparent. It is unable to understand user's emotions.

Chapter 5

Future Work

Conversational modeling is an important task in natural language understanding and machine in- telligence. Although previous approaches that is implemented , that are often restricted to specific domains and require hand-crafted rules. Now I will move to a sim- ple approach for this task which uses the recently proposed sequence to sequence framework.

5.1 seq2seq framework

This approach is based on recent work which pro- posed to use neural networks to map sequences to se- quences (Kalchbrenner & Blunsom, 2013; Sutskever et al., 2014; Bahdanau et al., 2014). This model converses by predicting the next sentence given the previous sentence or sentences in a conversation. The strength of This model is that it can be trained end-to-end and thus requires much fewer hand-crafted rules. We find that this straightforward model can generate simple con- versations given a large conversational training dataset. Our preliminary suggest that, despite op- timizing the wrong objective function, the model is able to extract knowledge from both a domain specific dataset, and from a large, noisy, and gen- eral domain dataset of movie subtitles. This is implemented by google brain team On a domain-specific IT helpdesk dataset, the model can find a solution to a technical problem via conversations. On a noisy open-domain movie transcript dataset, the model can perform simple forms of common sense reasoning. As expected, we also find that the lack of consistency is a com- mon failure mode of our chatbot or conversational model.

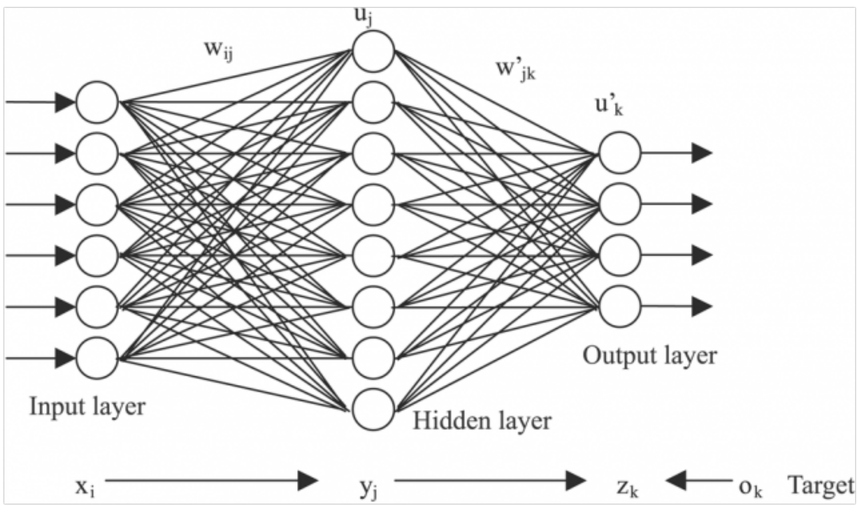

Figure 5.1: Neural modeling

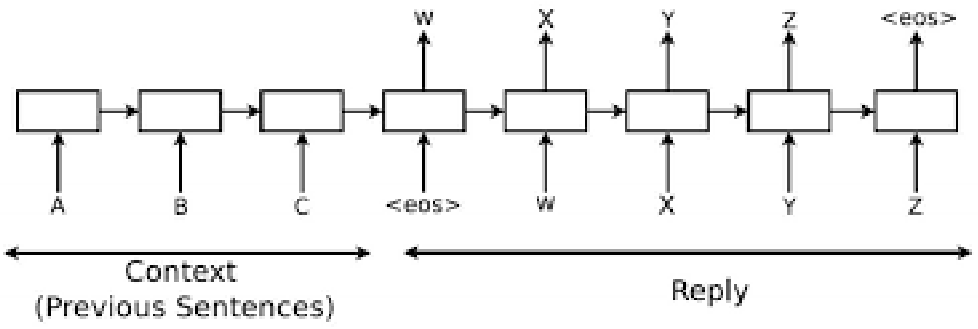

Figure 5.2: Using the sequence to sequence framework for modeling conversations

Chapter 6

Conclusion

To summarize, we have learned about all the existing systems which included ELIZA. We found out ways in which these systems can influence our system. ELIZA helped us understand how reframing the questions will make the conversations more human-like. We decided to build a system to implement natural language processing on CLI(command line interface). In this system, the user will be prompted to ask a query which is related to daily life. The user will text out his query using the command line. A range of related questions and their responses will be coded and stored into the dataset. When the user will ask a query, this query will be matched against the various patterns present in the dataset and the template corresponding to that pattern will be returned in form of text to the user.

References

[1] Turing, A. M. Computing machinery and intelligence. Mind, pp. 433460, 1950.

[2] SPEECH and LANGUAGE PROCESSING by Daniel Jurafsky and James H. Martin , `http://www.cs.colorado.edu/~martin/slp2.html`

[3] Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, `http://ibug.doc.ic.ac.uk/media/uploads/documents/courses/CBR-AamodtPlaza.pdf`

[4] NLTK documentation, `http://www.nltk.org`

[5] Python documentation, `https://www.python.org/doc/`

[6] Python Eliza implementation, `http://www.jezuk.co.uk/cgi-bin/view/software/eliza`

[7] The Genealogy of Eliza, `http://elizagen.org/index.html`

[8] Peter Norvig's blog, `http://norvig.com`

[9] Resource for Latex, `http://en.wikibooks.org/wiki/LaTeX/`

[10] Neural Modeling, `http://www.extremetech.com/wp-content/uploads/2015/07/NeuralNetwork.png`

[11] A Neural Conversational Model, `http://arxiv.org/pdf/1506.05869.pdf`

[12] Google Research blog, `http://googleresearch.blogspot.in/2015/11/computer-respond-to-this-email.html`