IJCSE International Journal of Computer Sciences and Engineering Open Access

Research Paper Vol.6, Special Issue.10, Nov 2018 E-ISSN: 2347-2693

BOT–ENGINEER: A system to Create Chatbots

Ravino De Souza*[1], Kai DaCosta[2], Deston Cardozo[3], Maria Christina Barretto[4]

*1,2,3,4 Department of Computer Engineering, Don Bosco College of Engineering, Fatorda, Goa, India

Corresponding Author: ravinodesousa@gmail.com , Tel.: +919673630869

Available online at: www.ijcseonline.org

Abstract— Digital Assistants driven by Artificial Intelligence (A.I.) are becoming increasingly popular .These assistants can send users updates on live data, help them know about the traffic situation on the way to work or home. All that the user has to do is ask. Due to increasing technologies, large amount of data gets generated. Digital assistants are changing the way people search for information, making it part of regular conversation. In this paper, we present a BOT-ENGINEER, a system that is designed to create chatbots. It uses intent classification and entity extraction to find what the user has said and meaningful keywords that represent actual data respectively.

Keywords- casual intents, business intents, utterance

I. INTRODUCTION

Now (2012), Alexa (Amazon 2015) are among the top voice-enabled digital assistants. These assistants can send users updates on the weather, help them know about the traffic situation on the way to work or home, book a weekend getaway and even order a pizza. All that the user has to do is ask. Digital assistants are changing the way people search for information, making it part of regular conversation. These existing assistants however are mostly productivity oriented, designed to support users in completing a range of tasks. An offshoot of personal digital assistants are A.I. powered text-messaging chatbots. A chatbot is an artificially intelligent chat agent that simulates human-like conversation by allowing users to type questions and, in return, generating meaningful answers. Among recent popular chatbots there is Microsoft's Xiaoice in China, available on messaging apps Line and WeChat. Xiaoice is meant for casual chitchat. She has about 20 million registered users, who are said to be drawn to her for her sense of humor and listening skills. Users often turn to Xiaoice when they have a broken heart, have lost a job or have been feeling down. Xiaoice can tell jokes, recite poetry, share ghost stories, and relay song lyrics. She can adapt her phrasing and responses based on positive or negative cues from the user. Currently Xiaoice is also a live weather TV host on Chinese TV, a job she has held for the past year. Similar to Xiaoice, there's chatbot Rinnain Japan on Line messenger, and more recently chatbotZo in the U.S. released on Kik messenger. All of these are general purpose chatbots, meant to be more like a "friend" than an "assistant". In India, there is chatbot Natasha on Hike Messenger, who can tell the user about a movie rating, the weather, send quotes, and search Wikipedia.

Looking at the increasing usage of chatbot by the young and the recent trend that has surfaced for the elderly crowd to use chatbots to get what they like.This paper proposes a method to design a system that can be used to create a chatbot. A MovieBot that would help a user find about a particular movie, the director, the cast, release date of a particular genre of concern. The proposed system uses
'intents', 'utterances' and 'entities' to design a chatbot. Flask framework is used to connect the python code to the user interface.

A. *INTENT*

Intents are the intentions of the end-user that are conveyed to the bot. Intents can be categorized into:

1. Casual Intents
2. Business Intents

1. Casual Intents—They may also be referred to as **'Small talk'** Intents. These intents mark the beginning or ending of a conversation. The Greetings like "hi", "hello", "Hola", "Ciao" or "bye" are the opening or closing statements in a conversation. These intents should direct the bot to respond with a small talk reply like "Hello what can I do for you today" or "Bye thanks for talking to me". The casual intents also comprise of Affirmative and Negative intents for utterances like "Ok", "yes please",etc. Having General affirmative and negative intents helps to handle all such intents and rather take them in context with the conversation bot just had with the client. For example—if the Bot just asked a question to end- user—it expects either an affirmative or a negative intent and if it is anything else, bot can ask the same question again. Affirmative and negative intents should be able to handle most such utterances.

2. Business Intents—These intents directly map to business of the bot. For example—if it is a Movie information Bot then an utterance from client like "When was Schindler's list released?" is a business intent that intends to find out the Release year of Schindler's list and it should be labelled accordingly with an understandable name like "GetReleaseYearByTitle".
Ideally you should think more about business intents because rest of small talk like saying hellos or affirming choices is taken care by general casual intents.

B. *UTTERANCE*
Utterances are the specific phrases that people use when making a request to chatbot. These can be hugely varied.There are number of ways people can ask for the time:

- "What time is it?"
- "What's the time?"
- "How late is it?"
- "What's the time now?"
- "Do you have the time?"

When developing a skill, utterances have to be coded to tell the bot what to expect. This can mean typing many questions with slight variations .

C. *ENTITY*
Business intents have metadata about the intent called "**Entities**".For example,consider a sample utterance "When was Schindler's list released?"
Intent "GetReleaseYearByTitle"

Here "Schindler's list" is the title of the movie for which the user "intends" to find out the release year.

II. RELATED WORK

Customer service chatbots can be roughly categorized into two types: first-party and third-party. First-party chatbots refer to conversation engines developed by large enterprises for their own business to improve customer service quality and reduce overall customer service budget. This often happens in consumer-driven industries such as banking, telecoms, and e-commerce. One example is the recently launched chatbot Erica from Bank of America, which helps customers with banking-related problems.

III. METHODOLOGY

Text input is processed by a "classifier". A classifier is used for categorizing a piece of data (a sentence) into one of several categories (intents).This classification associates an input sentence with an "intent" which produces a response.The input "how are you?" is classified as an intent, which is associated with a response such as "I'm good" or "I am well."

We learnt about classification early in elementary science: a chimpanzee is in the class "mammals, the earth is in the class "planets" and so on.
Generally, there are 3 different ways of text classification: Pattern Matchers, Algorithms and Neural Networks. Regardless of which type of classifier is used, the end-result is a response

A. *INTENT CLASSIFICATION USING ALGORITHMS*
The proposed method uses **Multinomial Naive Bayes** algorithm classify intents.
Multinomial Naive Bayes is a classic algorithm for text classification and natural language processing (NLP).
This classifier is "naive" because it assumes independence between "features", in this case: words. Each word is treated as having no connection with other words in the sentence being classified.
For example, in the sentence "the fox jumped over the log", there is no relationship between "jumped" and "fox" or "log". This ensemble of words is often referred to (by NLP types) as "a bag of words". While this is incredibly naive, the classifier isn't attempting to understand the meaning of a sentence, it just classifies it.

Following steps are used to classify a sentence using **Multinomial Naive Bayes Algorithm:**
Step 1:
NLTK (natural language toolkit) for **Tokenization and Stemming**:
Tokenization: breaking up sentences into words.
For example. "have a nice day" tokenizes to a list of individual words: "have", "a", "nice", "day"
Stemming: reducing words to their stem.
For example. "have" stems to "hav" which allows it to be matched with "having" (same stem).

Step 2:
Provide some training data with few sentences which are associated with each intent (a "class"). If the user says "good day", it should be classified as "greeting" intent.
For example,
Training_data = {
"greeting" : ["how are you?", "how is your day?", "good day", "how is it going today?"],
"goodbye" : ["have a nice day", "see you later", "have a nice day", "talk to you soon"] }
Step 3:
Organize data in structures that can be worked algorithmically.
Two data structures are used:

1. **corpus_words** : consists of each stemmed word and its occurrence in the corpus.
2. **class_words** : each class and the list of stemmed words within it.

Notice the "corpus" (an NLP term) is a collection of all stemmed words. For example, the stem of "make" is "mak" so that it matches the stems for "making".
Corpus words and counts: {'how': 3, 'ar': 1, 'see': 1, 'is': 2, 'can': 1, 'me': 1, 'good': 1, 'hav': 3, 'talk': 1, 'soon': 1, 'yo': 1, 'you': 4, 'day': 4, 'to': 1, 'nice': 2, 'lat': 1, 'a': 5, 'what': 1, 'for': 1, 'today': 2, 'it': 1, 'going': 1}
Class words: {'goodbye': ['hav', 'a', 'nice', 'day', 'see', 'you', 'lat', 'talk', 'to', 'you', 'soon'], 'greeting': ['how', 'ar', 'you', 'how', 'is', 'yo', 'day', 'good', 'day', 'how', 'is', 'it', 'going', 'today']}

Step 4:
Consider an input sentence and keep track of score
At first, Score is initialized to zero.
Tokenize the given sentence and consider each stemmed word at a time.
Then for each class if a word matches any of the words is in class_words then score is updated.

Score + = $\frac{1}{\text{he occurrence of that word in corpus\_words}}$

For example,
Consider an input sentence: "see you"
the word "see"is in class_word under class: goodbye
Score+= (1/occurrence of "see" in corpus_words)
Score+= (1/2) = 0.5

match : see(0.5)
match: you(0.25)
Class: goodbye Score: 0.75

match: see (0)
match: you (0.25)
Class: greeting Score: 0.25

Then consider the class with highest score and score greater than 0.5.
If above conditions are not satisfied then don't consider any class.
So,in above example consider the class goodbye.

B.ENTITY EXTRACTION

We have used the algorithm proposed in "Assessing sentence similarity through lexical, syntactic and semantic analysis" to extract entities from a sentence.
Steps used to extract entities:
Step 1:
By giving a sentence as input, a list of sentence tokens as output is obtained. It does the following things:

1. Tokenization: This step splits the sentence into a list of tokens, including punctuation.
2. Stop word removal: Words with little representative value to the document, such as articles and pronouns, and the punctuation marks are suppressed.
3. Lemmatization: This step translates each of the tokens in the sentence into its basic form.

For example,
am, are, is -> be or car, cars, car's, cars' -> car
For example,
If we provide a sentence, "When was Titanic released?" we get,
tokens: ['When', 'was', 'Titanic', 'released','?']
Remove Stop Words: ['Titanic', 'released']
lemmas: ['Titanic', 'released']

Step2:
```
Similarity_between_words
{  if Pathmeasure(word1, word2) < 0.1 then
similarity = LevSimilarity(word1, word2)
else
similarity = Pathmeasure(word1, word2) }
```

Pathmeasure :-It is the length of the path between two concepts in WordNet Graph to score their similarity.
LevSimilarity :- It is based on Levenshtein distance, which counts the minimum number of operations of insertion, deletion, or substitution of a single character needed to transform one string into the other. The similarity is calculated as:
LevSimilarity = 1.0−(LevenshteinDistance(word1,word2) / maxLength(word1, word2))

Step 3:
Next step is to calculate a matrix that contains the similarities of the words in the sentence using *Similarity_between_words* from step 2
For example, matrix for "who directed Assassin's Creed?" when matched with entity "Assassin Creed" is as follows

	directed	Assassin	Creed
Assassin	0	1	0
Creed	0.5	0	1

Select the maximum number and if its >=0.9 then append it to a list.
Next we delete the row with max element. For previous example we get,

	directed	Assassin	Creed
Creed	0.5	0	1

Again check for max element and if its >=0.9 append it to a list.
So for above sentence,
list = [1, 2]

And if length of list is equal to number of tokens of entity then it matches.

VI. CHATBOT INTERFACE

The proposed system consists of a main page that has a floating chat icon, which when clicked displays the typical message window where the top part consists of Chatbot name, the middle part or the body of the messaging window consists of the conversation between user and chatbot and lastly at the bottom of the window there is input bar where the user can type questions. The chatbot then displays the answer in the body of messaging window.
It also consists of a login page which only the administrator can access and after typing his id and password he/she can get access to the Administrative panel.

The Administrative panel consists of following options:
1. Add new / Select Chatbot:
   This option allows to create new chatbot or select from a list of pre-existing chatbots. When we create a new chatbot it automatically creates the necessary files to store intents, entities and utterances.
2. Add/View Database
   Here the admin can add data to a database using CSV files and can also view data from database. The admin just has to select a table from a list of tables which are present in the database. This will be useful to write SQL queries.
3. Add / Delete / View Intents
   This section allows to add new intents or delete an intent from a list of intents already added to the intent file. json file is used to store intents where the key is the actual intent and value is a list of utterances that fall under that intent. Admin can also make a CSV file and can import it, this automatically adds intents to the json file.
   For example,
   Consider an intent.json file which consists of data as shown below

```
{
“greeting” : [“how are you?”, “how is your day?”,
“good day”, “how is it going today?”],
“goodbye” : [“have a nice day”, “see you later”,
 “have a nice day”, “talk to you soon”]  }
```

In the above example “greeting” and “goodbye” and two such intents and “how are you?”, “how is your day?”, “good day”, “how is it going today?” are utterances that fall under that intent.
1. Add / Delete Entities
   Provides functionality to add new entities and also delete old entities which are no longer needed. These newly added entities are stored in another json file.

For example,
Consider an entities.json file that consists of below data

```
{“movie_name”:{“Allied”,”BatmanvsSuperman”, “Avengers”} }
```

In above example “movie_name” is an entity id and “Allied”, “Batman vs Superman”, “Avengers” are entity values.
Suppose if we compare “get cast of Allied” with “get cast of movie_name” and use our entity extraction algorithm, it will give us entity_id as “movie_name” and entity_value as “Allied”.
2. Add / Delete Utterances:
   Here the admin can select an intent from a list of intents and can then see all the utterances present in that intent. If the admin wants to make some changes then he can click on edit button which will open a popup window that consists of old utterances. The admin can then make changes to the utterance and click on update. The admin can also delete utterance from intents.

3. Add / View / Delete Answers:
   Answers are present in 3 categories “direct_data”, “db_data” and “website_data” and each category consists of intents which in turn consists of answers.
   direct_data consists of all answers to casual intents db_data consists of SQL queries that will retrieve answers from database and website_data consists of answers that will redirect the user to a particular site.
   For example,in case we used “http://www.google.com/search?q=rotten +tomatoes+review+of”

and attach the movie name at the end of the link.
This answer is only available if the user types a question like “get review of Allied”.

```
{
"db_data": {
"best_ranking": ["Best movie is {} ", "SELECT
Title from movie_data where Rank='1'"]
},
"website_data":{"get_review":
["http://www.google.com/search?q=rotten+tomatoes
+review+of"] },
"direct_data": {
"start_greetings": ["hi", "hello", "Greetings!", "hey"]
} }
```

Here admin can add new answers and also can edit / delete answers present in answers.json file.
4. View Unanswered Questions
   Here admin can view questions which were given to the chatbot which the chatbot wasn’t able to answer. The admin can also delete the question if it has already been added by him/her.

V. CONCLUSION AND FUTURE SCOPE

The proposed system answers single sentence questions and cannot answer cascaded or complicated questions. Deep learning and neural network techniques for intent classification can be used which are much faster than normal algorithms.

REFERENCES

[1] Adhitya Bhawiyuga, M. Ali Fauzi, Eko Sakti Pramukantoro, Widhi Yahya, "Design of E-commerce chat robot for automatically answering customer question", *Sustainable Information Engineering and Technology (SIET) 2017 International Conference on*, pp. 159-162, 2017

[2] Emanuela Haller ; Traian Rebedea, "Designing a Chat-bot that Simulates an Historical Figure", 2013 19th International Conference on Control Systems and Computer Science, ISSN: 2379-0474

[3] Salto Martinez Rodrigo ; Jacques Garcia Fausto Abraham, "Development and Implementation of a Chat Bot in a Social Network", 2012 Ninth International Conference on Information Technology - New Generations, ISBN: 978-1-4673-0798-7