Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

Artificially Intelligent Chatbots to Resolve Manageable and Frequently Asked Hardware/Software Queries

Srishti Gupta, PhD Scholar,
IIT Patna, India

Shweta Yerram, Sachit Tiwari,
Infosys, India

***Abstract*:- S-BOT, an acronym for Service BOT, is meant to provide services to end users in a multinational company. It serves the purposes of all the employees who might be in the need of technical enhancements without any fuss. This functionality provides redirection of service requests to the vendors, who will thereby be able to process it appropriately. All the tangible requests are accepted by the BOT and carried forward to the task manager. It is a user friendly application, in a messenger-like service that helps you access task manager and his services. After acquiring the services of the BOT, a constructive feedback is given by the user that helps the administrators expand their technological advancements. S-BOT is a well trained, advanced BOT that has assisted an MNC to empower its customer service, by providing an advanced and adequate version of help to its employees. This highly technical BOT will serve as the baseline and contribute and enlighten the industry about the existence of BOTs and therefore machine learning. Hence, S-BOT paves the way to machine learning and its applications in various fields of the real world.**

Keywords— Artificial intelligence; Chatbots; QnA about resolving issues; Microsoft bot framework; Bot emulator; Node js; LUIS framework; frequently asked questions.

I. INTRODUCTION

Replying to frequently asked questions (FAQ) or providing simple and timely information are perfect scenarios for a chatbot [8]. Responding to the same requests over and over is a major burden for customer support. A chatbot will be of great help by providing correct answers, directly replying, or even escalating the request to a person freeing up agents' time to work on more complex issues. Chatbots make life even easier for consumers. There's no more long waits on hold to talk to a person on the phone or going through multiple steps to research and complete a purchase on websites with the help of Chatbots.

Machine learning Chatbot [1] works using various algorithms of artificial intelligence. Users can be liberal and not to be more specific while talking with a bot because it understands the natural language, not only machine commands. These kinds of bots get continuously better or smarter by learning from past conversations with people.

In the near future, the simpler Chatbots are predicted to dominate with less Artificial Intelligence and more rule-based development (like websites or apps that you click or do something that triggers another action). One can already see various apps' latest adjustment to use more buttons (vs. primarily texting). For the mainstream to understand every user input, people and technology are just not quite there yet. So buttons and occasional text commands are comparatively easier to use and develop. With evolution in technology, more and more opportunities arise for integrations (like you see every day with APIs, IOT etc.). Chatbots can become a great hub for these, translating to us, keeping us updated and waiting for us to give certain commands.

Technically, now everything that you interact with in your everyday life could be communicated with you and each other via a Chatbot in the future. In case of longer term, as technology evolves and we go with the flow, more AI and auditory (speech based) Chatbots will be invited to our lives. Speech recognition has also improved enormously in the last couple of years, but it's nowhere near where it should be. Therefore, as years pass by, more efficient and convenient Chatbot will hopefully surround us. Most likely there is going to be a point where humans and bots partly join each other. From here on we must optimize our brains as well to a point where we can just call information with a thought; don't have to use inefficient words or texts anymore.

Every small business that is barely hanging in there, with just a website and FB page as of now, must have at least a basic Chatbot so people can interact with them via their systems. That is why these days; certain tools enable people who have no clue about how technology works to build pretty usable websites. Some solutions that already exist for bots as well and it is only going to get better.

Working Modules

1. Bot Module

The S-Bot module provides a platform for the interaction between the user and the LUIS API. In this S-Bot module, the user actively interacts with the bot by posing his queries. The bot processes these user queries and thereby connects it to the backend of the bot called the LUIS API. It checks if the query entered by the user matches any of the existing utterances of the database. Once it matches a particular utterance, the solution for the matched intent is passed to the bot. The bot processes the solution and thereby displays it as a Hero card format by adding the necessary links to be sent to the user.

2. Bot Training Module

The LUIS module provides the prebuilt domains that help the bot application by providing intents and entities that can be mixed in and modified to create better language understanding required for the bot to interact with the user effectively. Through LUIS, the bot works interactively with the user by

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

retrieving necessary information according to the query provided by the user. LUIS acts like a backend to the bot application. It also provides the option to train our application with different user utterances by enabling suggested utterances to be displayed in the database. This module thereby helps the bot application to be updated from time to time easily.

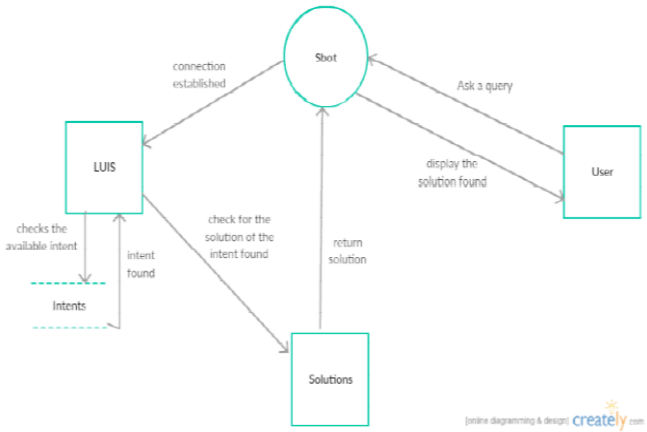


Fig. 1. Data Flow Diagram

3. User Module

This module helps the users to ask queries regarding service issues to the bot. The bot directly interacts with the user to provide the necessary solution by searching in the intents in the LUIS module. When the user receives the apt solution he can exit the chat with the bot. Otherwise, the user is provided with an option to search the web and thereby refer to the solutions provided on the web or he can also transfer the request to the FMS department by sending them his employee ID through the bot application.

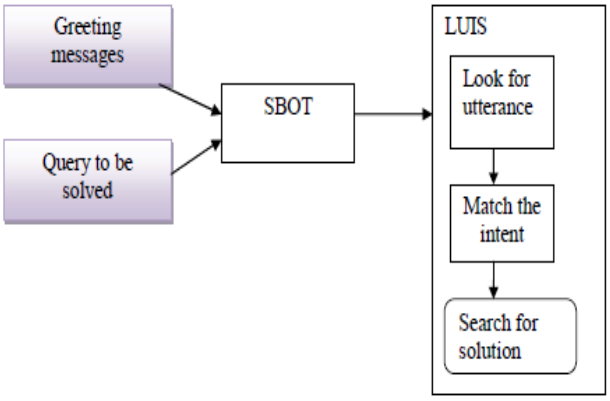


Fig. 2. User module

II. REQUIREMENTS

Initially one must understand how a Chatbot works, internally. Simply put, given a user input, a Chatbot returns a response. A simple principle but in practice, things are not so easy.

Understanding what the user says

Assume that you are dealing with a travel Chatbot. So first, the Chatbot needs to understand the input. There are two main techniques to tackle this situation: *pattern matching and intent classification*.

The good point with this approach is that the patterns can be read by humans, so the input modeling phase can be somehow straightforward. The problem is that patterns are built manually. It is not a trivial task. In several real use cases it does not scale. An intent classification approach generally relies upon machine learning techniques. Among all the possibilities you need a set of examples to train a classifier that will choose, given a user input. To extract the important entities like locations, airlines, airports, dates, etc, the Chatbot needs to perform previously information extraction on the input. Classifying the input and extracting information from it are two key concepts that you have to keep in mind.

Responding to the user

It can choose or generate a response, once the Chatbot understands what the user says, based on the current input and the context of the conversation.

Key software/frameworks in building a solution for the pertaining problem are:

A. Microsoft Bot Framework

The Bot Framework [2] is a platform that helps to build, connect, test, and deploy powerful and intelligent bots. You can get the Bot Builder SDK and quickly start building bots with the Bot Framework and with support for .NET, Node.js, and REST.

The Bot Framework helps in building bots that support different types of interactions with users. You can design conversations in your bot to be free form and more guided interactions where it provides the user choices or actions. Their conversations can just use simple text strings or more complex rich cards that contain text, images, and action buttons. One can also add natural language interactions that let your users interact with your bots in a natural and expressive way.

B. Microsoft Bot Emulator

The Bot Framework Emulator [3] is a desktop application that helps in testing and debugging your bot, be it locally or remotely. Using the emulator, you can chat with your bot like a channel and inspect the messages that your bot sends and receives as if it were published. The emulator is a desktop application, not a mobile one that lets you test and debug your bot on localhost or running remotely through an ngrok tunnel.

C. Language Understanding Intelligent Service (LUIS)

Language Understanding Intelligent Service (LUIS) [4] enables developers to build smart applications that enable understanding human language and reacting accordingly to user requests. Just so that your application doesn't have to, LUIS uses the power of machine learning to solve the difficult problem of extracting meaning from natural language input.

Generally, you create an intent to trigger an action in a client application or bot and create an entity to model some parameters required to execute an action.

D. Node JS

Node.js [5] is a platform built on Chrome's JS runtime for easily building fast yet scalable network applications. Node.js uses an event-driven and non-blocking I/O model that makes it lightweight and efficient. It is efficient for data-intensive real-time applications that perform across distributed devices.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

Node.js is an open source and a cross-platform runtime environment in order to develop server-side and networking applications. Node.js applications are coded in JavaScript, and can be executed and run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

E. Visual Studio Code

The Visual studio [6] is an innovative launching Integrated Development Environment (IDE) pad which is used to edit, build code, debugs, and publish an application. It is one of the characteristic-rich programs used for the software development. Compared to the many IDE's the Visual Studio comprises of compilers, code completion tools, graphical designers and many more characteristics to simplify the process of developing an application software.

III. WORKING OF THE CHATBOT

BOT is a means to provide the engineers of an MNC, a moment of respite from gathering all calls from various departments of the MNC. Employees can enter their problems and chat with the bot, hence providing few pre defined solutions. If the problem still persists, then the engineer may be of utmost help.

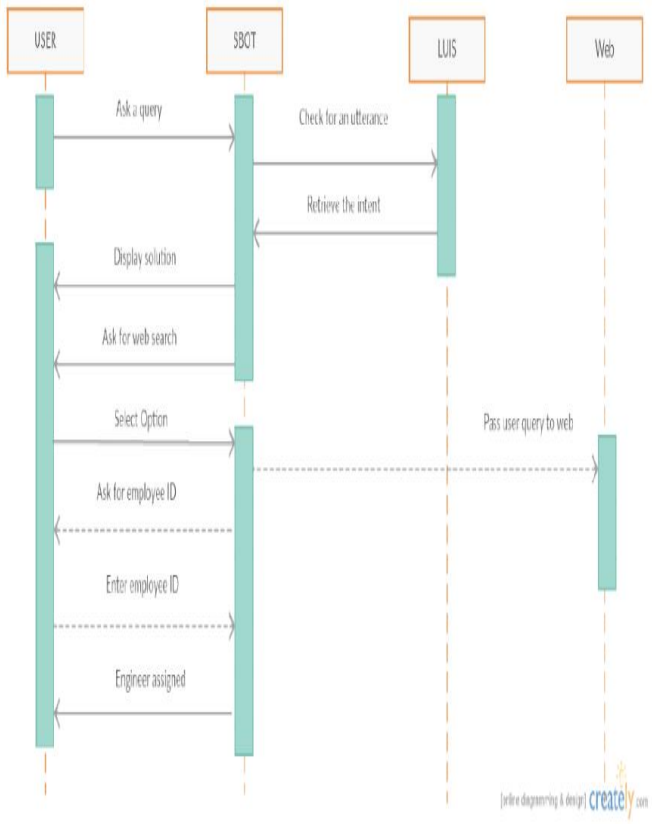


Fig. 3. Sequence Diagram

A. Initial greeting

For the bot to initially respond, one must trigger the conversation by initially telling a 'hi' like greeting to the bot. Once this is received by the bot, LUIS connection is triggered and the intent is matched. Intent matching leads to the Node JS code being checked and the following matter is displayed to the user. It also shows what kind of queries can be asked.

Fig. 4. Initial greet

Node JS code to show how the backend works:

```
bot.dialog('greetmsg',
function (session,args,next)
{
session.send("Hi! I'm S-bot.\n"+"\n"+"\nTell me what is the problem you are facing so that I can help you provide a solution.\n"+"\n"+"\n I can convert speech to text as well. \n"+"\n"+"\nPlease check your keywords before submitting");
session.beginDialog('/choice')
}).triggerAction({
matches: 'greetmsg'
});
```

B. Hardware/Software problem

The initial greeting follows up with various aspects the user is informed about. A set of queries are provided to him so he has some idea of the type of queries that he can ask the bot about. Once he enters his statement of doubt, the bot searches for the type of intent and the corresponding Node JS code with the predefined solution is displayed.

Fig. 5. Solutions

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

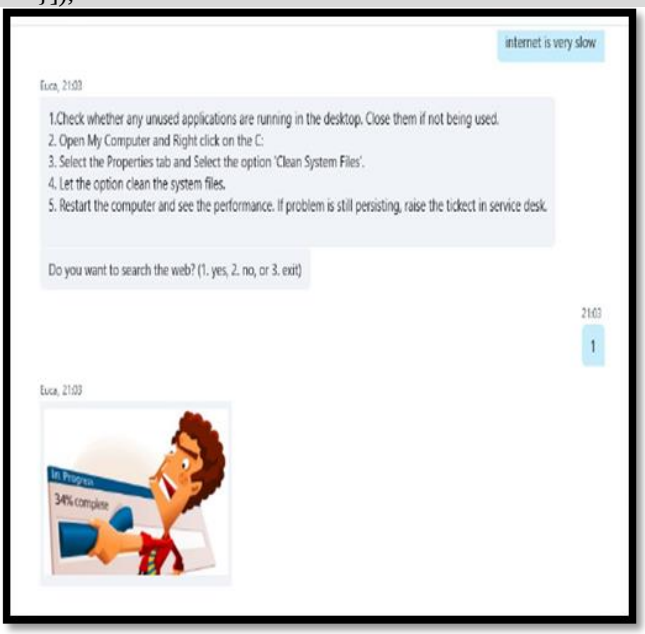
International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

C. Case 1: Web search option

In case, the bot is unable to provide answers, an option of web search is available. If the employee proceeds to check the web for better solutions, the query asked by the user is automatically copied on the Google Search page.

Node JS code to show how the backend works:

```
bot.dialog('/sdsrcard',
[ function(session,sdsr)
{
avg=sdsr;
builder.Prompts.choice(session,'Do you want to search the web?',options);
session.message.text=avg;
},
function(session,results,avg)
{
var sc = results.response.entity;
if(sc==='no')
{session.beginDialog('None');}
else if(sc==='yes')
{
var card = createHeroCardsdsr(session,sc,session.message.text);
var msg = new builder.Message(session).addAttachment(card);
session.send(msg);
session.endDialog();
}
else
{
session.send("I hope we were able to assist you.");
session.endDialog();
}
}]);
```



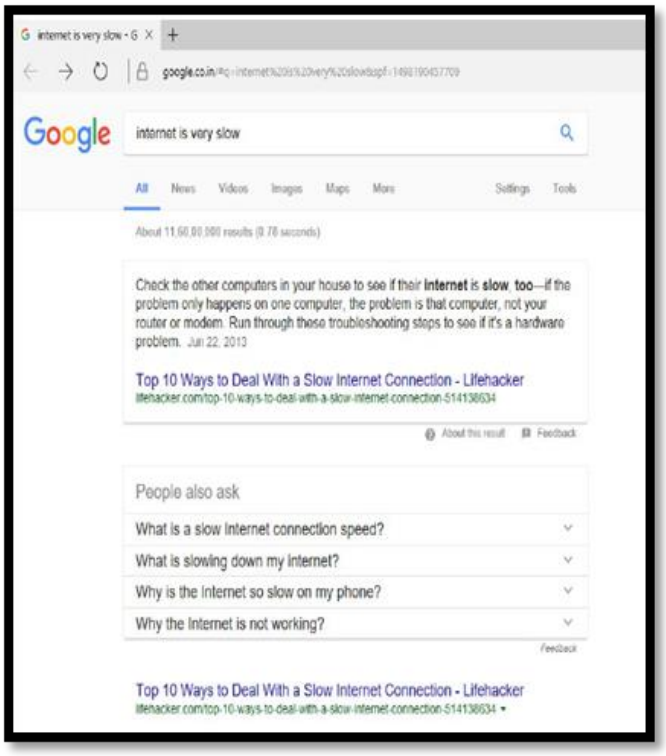


Fig. 6. Web search

D. Case 2: Employee ID

The other option available to the employee is of asking for an engineer to help them out. In case they say no to the search web option, the bot apologises for the inconvenience and asks the employee for their ID number so the FMS department knows which desk or employee needs their help.

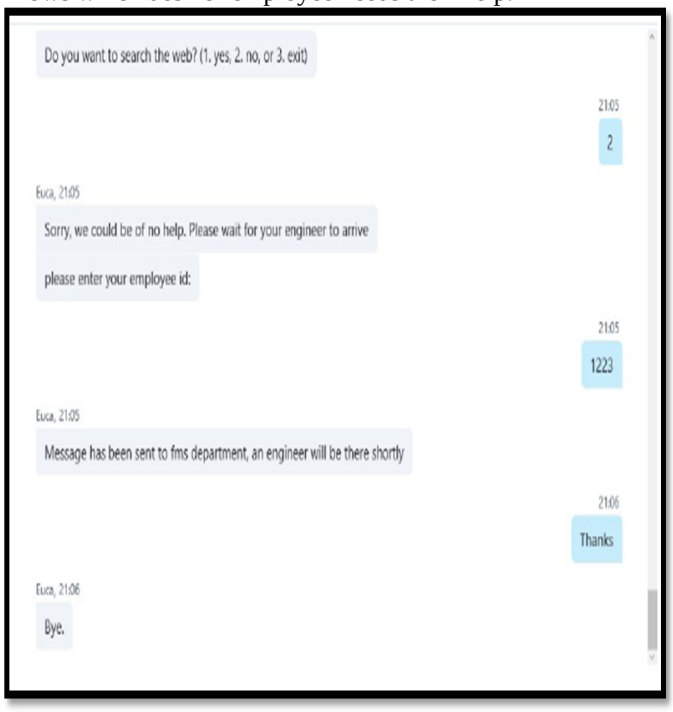


Fig. 7. ID provision

E. Case 3: Exit

In case the employee is satisfied with the solutions provided by the Bot, they can end the chat by just a Goodbye

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

message. Once the bot understands that the employee needs no more help, either if he types exit or types any other exit message, the bot tends to end the session begun for that particular employee.

Fig. 8. Exit dialogue

F. Unintelligible utterances

In case the user types a query the bot doesn't understand, it doesn't end the session. Rather, it gives the employee a hint that the command wasn't understood and the employee id expected to ask again.

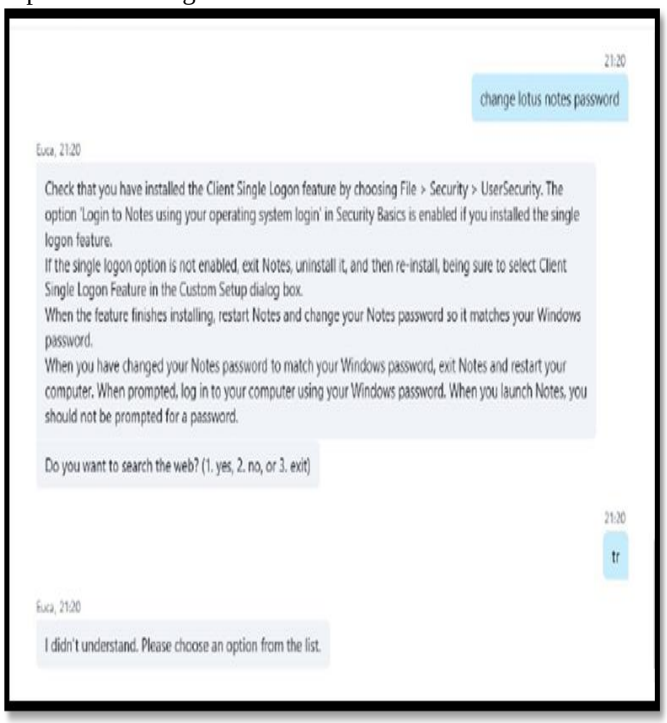


Fig. 9. Unintelligible utterance

IV. TRAINING THE BOT

Before you start creating it in the LUIS web interface, plan your LUIS app by preparing an outline or schema to describe intents and entities in your application. Generally, you create an intent to trigger an action in a client application or bot and create an entity to model some parameters required to execute an action. For example, for booking a plane ticket, "BookFlight" intent could trigger an API call to an external service, which requires entities like the travel destination, date, and airline. See guidance on how to choose intents and entities to reflect the functions and relationships in an app.

The LUIS module provides the prebuilt domains that help the bot application by providing intents and entities that can be mixed in and modified to create better language understanding required for the bot to interact with the user effectively. Through LUIS, the bot works interactively with the user by retrieving necessary information according to the query provided by the user. LUIS acts like a backend to the bot application. It also provides the option to train our application with different user utterances by enabling suggested utterances to be displayed in the database. This module thereby helps the bot application to be updated from time to time easily.

The LUIS model begins with sets of user sentences called intents. Each intent needs examples of user utterances. Each utterance can provide a variety of data that the user might enter as a response.

An intent represents a task or action the user wants to perform. Your first task in the app is to add intents. Intents are the intentions or requested actions conveyed by the user's utterances. They are the main building block of your app. You now need to define the intents (for example, book a flight) that you want your application to detect. Go to the Intents page in the side menu to create your intents by clicking the Add Intent button.

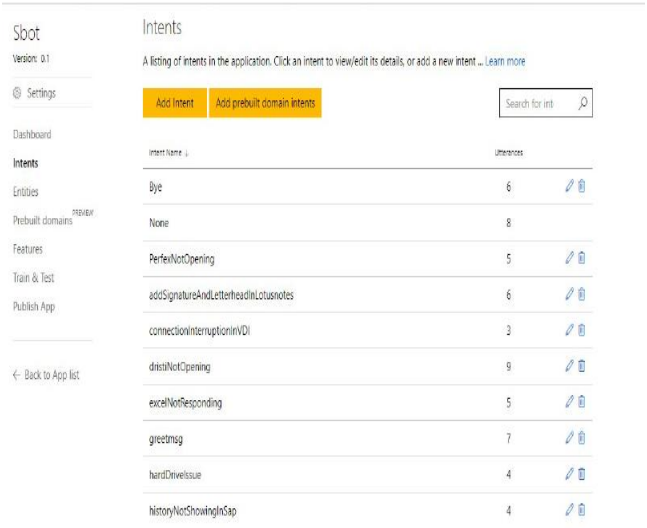


Fig. 10. Intents

Utterances are sentences from the user that your app needs to process. It's important to capture a variety of different example utterances for each intent, to train LUIS to extract intents and entities from them. Utterances aren't always well formed Now that you've defined intents, you can start seeding examples to each intent to teach the machine learning model the different patterns (for example, "book a flight to Seattle

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

departing on June 8th".) Select an intent you just added and start adding and saving utterances to your intent.

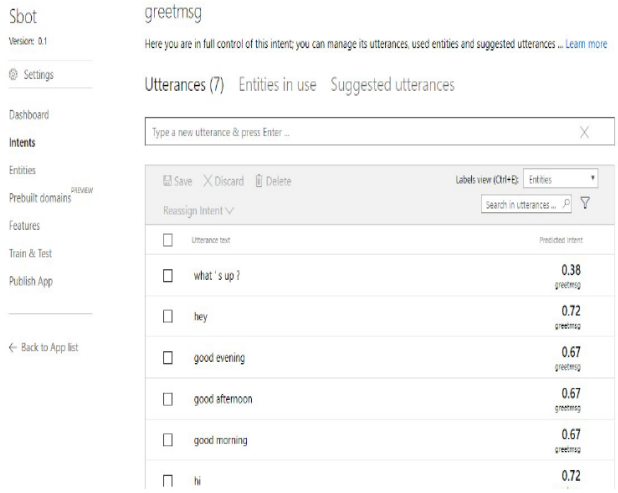


Fig. 11. Utterance for itents

Algorithm Used for Training

Multinomial Naive Bayes [7] is the classic algorithm for text classification and NLP. Each word is counted for its occurrence and is accounted for its commonality and each class is assigned a score with every new input sentence. The highest scored class is the most likely to be associated with the input sentence.

Using classification score one can identify the class with the highest term matches but it also has some limitations. This score helps to signify which intent is most likely to the sentence. This does not guarantee a perfect match so the highest score only provides the relativity base.

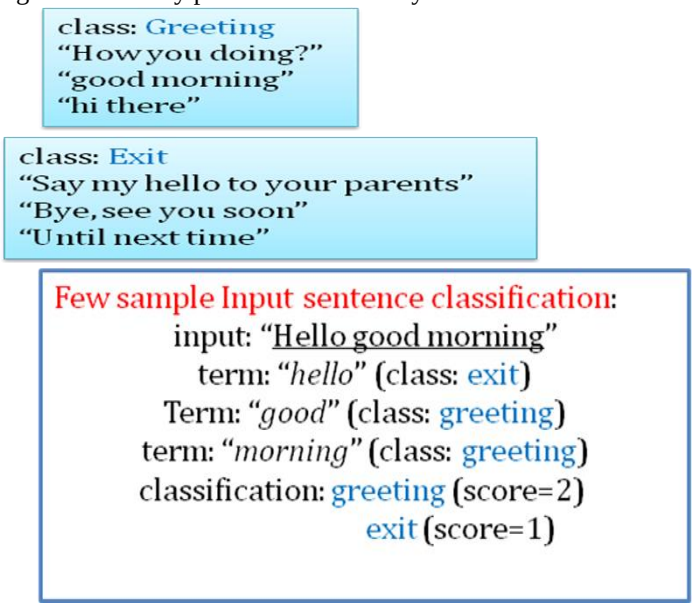


Fig. 12. Pictorial classification

Fig. 13.

Classification score identifies the class with the highest term matches but it also has some limitations. The score signifies which intent is most likely to the sentence but does not guarantee it is the perfect match. Highest score only provides the relativity base.

Features:

- In text classification, our goal is to find the best class for the document.
- It is better to perform the computation by adding logarithms of probabilities instead of multiplying probabilities.
- The class with the highest log probability score is still the most probable.
- The preprocessing necessary for computing the parameters (extracting the vocabulary, counting terms, etc.) can be done in one pass through the training data. The time complexity of this component is therefore,

O(|D|Lave)

where |D| is the number of sentences in the training set and Lave is the average length of the sentences.

Algorithm:

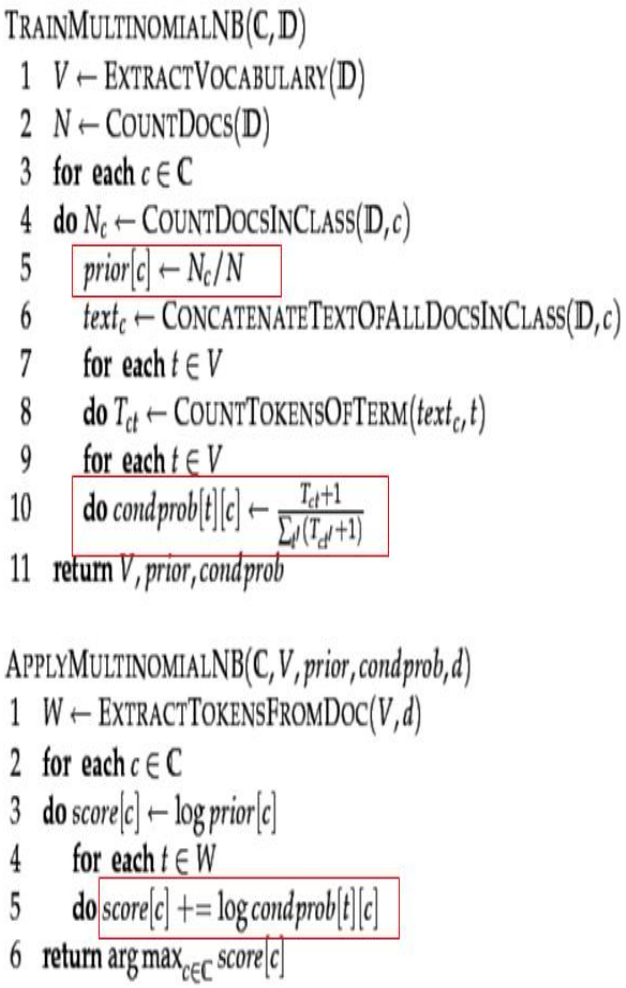


Fig. 14. Algorithm used

Multinomial Naïve Bayes [9] Algorithm suggests that every word of a sentence must belong to a particular class, in case of LUIS, these classes are called INTENTS. To differentiate every word to a class and see its association with various classes, a score is calculated. In order to calculate this score, the following function is implemented:

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

$$c_{map} = \arg\max_{c \in C} [\log \hat{P}(c) + \sum_{1 \le k \le n_d} \log \hat{P}(t_k|c)].$$

This function is divided in to three parts:

A. Prior Probability

$$P(C) = \frac{Nc}{N}$$

where, Nc = Number of documents in C

N= Total number of documents

Prior probability refers to the probability of an event before any further change is made to the data. This probability, P(C), denotes the probability of the occurrence of that particular class being calculated.

B. Conditional Probability

$$\hat{P}(t|c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}},$$

where, *Tct* = number of occurrences of *t* in training documents from class *c*, including multiple occurrences of a term in a document.

This conditional probability refers to the referential frequency of the occurrence of the word‘t’ in all the sentences present in the class ‘c’. ‘Log’ is applied to both the probabilities in order to avoid the Floating Point underflow that may occur during the calculations. *Note:* When observed in the formula of the algorithm, the conditional probability has ‘+1’ added to the numerator and the denominator. This is done to avoid the terms that might lead to 0 i.e. if a term is not present in any class that might lead to zero terms which may affect the probability calculations.

C. Score Calculations

The final score is calculated by adding these two probabilities and then the function ‘argmax’ is applied to the list of scores calculated. The function will return the highest score and also which class it came from.

V. TESTING THE BOT

Software testing attempts to execute a program or application with the intent of finding software bugs like errors or other defects. The job of testing is an iterative process, it can illuminate other, deeper bugs, or can even create new ones; as when one bug is fixed. It provides objective, independent information about the quality of software and risk of its failure to users or sponsors.

An application can never be 100% bug-free that is why it is difficult to ascertain when one can stop testing. To reduce the cost and time to rework and produce software that is bug-free so that it can be delivered to the client; testing should be started as early as possible.

Black Box Testing: The technique of testing in which the tester doesn’t have access to the source code of the software and is conducted at the software interface without concerning with the internal logical structure of the software.

TABLE I. BLACK BOX TESTING

Test case	Check field	Objective	Expected Result	Actual Result	Status
TC-01	Skype	Install Skype app	Should indicate that Skype installed successfully	Skype installed	Pass
TC-02	Skype	Launch the app in the mobile/ desktop	App should be launched without any error	App launched	Pass
TC-03	Sign in or Sign up	Fill in all the fields and click submit	Clicks submit	Details submitted	Pass
TC-04	Submit button	Fill in all the fields and click submit	Clicks submit	Error appeared	Fail
TC-05	Submit button	Click button with correct details	Successfully redirects to the page	Login page appeared	Pass
TC-06	New chat	Choose the Bot to start the chat	New chat with the Bot starts	Chat successfully launched	Pass
TC-07	New chat	Choose the Bot to start the chat	New chat with the Bot starts	Bot not found	Fail
TC-08	Initial message	Type the greet message	Hello message appears	Hello message appeared	Pass
TC-09	Initial message	Type the greet message	Ask the query to be solved	Question asked	Pass
TC-10	Query	Question to be asked is typed	Keywords identified	Keywords are wrong, the question is asked again	Fail
TC-11	Query	Question to be asked is typed	Solutions are given and asked for web search option	Solutions displayed	Pass
TC-12	Web search	Choose Yes	On Yes, redirect the Google page	Google page appears with correct query	Pass
TC-13	Web search	Choose Yes	On Yes, redirect the Google page	Google page appears with incorrect query	Fail
TC-14	Web search	Choose No	Engineer allotment process	Engineer allotted	Pass
TC-15	Engineer	Send details with related data	Enter the employee ID	Confirmation received	Pass
TC-16	Engineer	Confirmation	Issue solved	Issue solved	Pass

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 09, September-2020

TABLE II. REGRESSION TESTING

Test case	Check field	Remarks	Expected Result	Actual Result
TC-04	Submit button	User must enter appropriate details to enter Skype.	Clicks submit	Error appeared
TC-07	New chat	Use the Skype search option or ask a friend to share the BOT as contact	New chat with the Bot starts	Bot not found
TC-10	Query	Hero Cards are provided for reference	Keywords identified	Keywords are wrong and the question is asked again
TC-13	Web search	Dynamic URL passing is used in the BOT	On Yes, redirect the Google page	On Yes, redirect the Google page

- User Module

When the user starts interacting with another BOT or isn't found it is observed that the user doesn't meet his requirements of getting his queries solved.

- Sbot Module

Initially when the users were not provided with what types of queries could be administered by the bot, an option of Hero Cards was provided to them. Without those clues to the users, it was difficult for the users to make full use of the BOT in their hands. This, thereby, was misguiding the LUIS module.

- LUIS Module

When inappropriate queries were being administered by the BOT, LUIS had to provide solutions that nearly resembled the ones already present in it. Hence, affecting the Intents and their already present utterances.

VI. CONCLUSION

It is not mere automation but a bot, in the modern context, is a piece of software that can execute a digital task that would traditionally require human interaction. But they are typically designed to mimic or simulate human behavior to help engage in realistic conversations or providing information in a naturalistic and context-sensitive fashion. Most definitely, Twitter has had bots for years in the form of Twitter bots that can send automated responses and retweets, or automatically follow users.

Bots can also be found working in chat apps like Big basket and Skype and the fresh attention of two of the world's biggest tech companies, and massive advances in AI will mean that bots are now firmly in the limelight like never before. It's not the matter of technology anymore, there's the question of whether humans will take to their new AI servants. There can also be adverse effects on employment with the rise of the bots as it will inevitably mean a reduced reliance on call centre and support staff, which means millions of jobs, will be at risk, especially in developing countries. Hence this bot is not targeting any loss of jobs. Instead, this bot helps to reduce load on the recruiters.

VII. FUTURE RESEARCH DIRECTIONS

Various issues and several other questions and aspects can be added that the employees may need help with. These bots definitely help the employees sort out issues that does not require a person's involvement. Future enhancements can be done where the message can be conveyed in a better way so that the solutions also can be enhanced.

REFERENCES

[1] International Journal of Advanced Computer Science and Applications, Vol. 10, No. 9, 2019, "Artificial Intelligence Chatbots are New Recruiters".
[2] https://dev.botframework.com/#life-cycle
[3] https://docs.microsoft.com/en-us/azure/bot-service/bot-service-debug-emulator?view=azure-bot-service-4.0&tabs=javascript
[4] https://docs.microsoft.com/en-us/azure/cognitive-services/luis/what-is-luis
[5] https://docs.microsoft.com/en-us/azure/cognitive-services/luis/luis-nodejs-tutorial-bf-v4
[6] https://code.visualstudio.com/docs
[7] https://chatbotsmagazine.com/what-is-the-working-of-a-chatbot-e99e6996f51c
[8] https://en.wikipedia.org/wiki/Chatbot
[9] https://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html

(This work is licensed under a Creative Commons Attribution 4.0 International License.)