Programming challenges of Chatbot: Current and Future Prospective

AM Rahman[1], Abdullah Al Mamun[1], Alma Islam[2]
IEEE[1], International Islamic University Chittagong[2]
amrahman@ieee.org[1], almamun@ieee.org[1], alma.iiuc@gmail.com[2]

***Abstract*— In the modern Era of technology, Chatbots is the next big thing in the era of conversational services. Chatbots is a virtual person who can effectively talk to any human being using interactive textual skills. Currently, there are many cloud base Chatbots services which are available for the development and improvement of the chatbot sector such as IBM Watson, Microsoft bot, AWS Lambda, Heroku and many others. A virtual person is based on machine learning and Artificial Intelligence (AI) concepts and due to dynamic nature, there is a drawback in the design and development of these chatbots as they have built-in AI, NLP, programming and conversion services. This paper gives an overview of cloud-based chatbots technologies along with programming of chatbots and challenges of programming in current and future Era of chatbot.**

Keywords - Chatbot, NLP, Machine Learning (ML), Artificial Intelligence (AI).

I. Introduction

A chatbot is an instant messaging account that able to provide services using instant messaging frameworks with the aim of providing conversational services to users in an efficient manner. A chatbot is fast with less confusing web and mobile application which is easy to install as there is no need to have installation packages. These packages are easy to manage and distribute [1]. Chatbots are totally different from the human accounts as they do not have any online status or last seen timestamps nor initiate the conversations and calls with any other accounts. Figure.1 shows some types of chatbots which are in used in different domains.

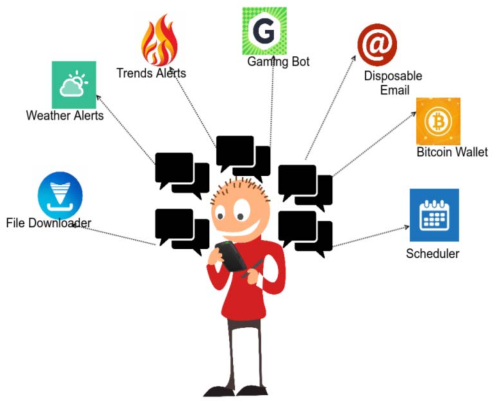


Figure.1 Chatbots in different domain

The overall chatbot Architecture is shown in Figure.2. Intent classification module identifies the intent of user message. Entity recognition module extracts structured bits of information from the message. The candidate response generator is doing all the domain-specific calculations to process the user request. The response selector just scores all the response candidate and selects a response which should work better for the user.

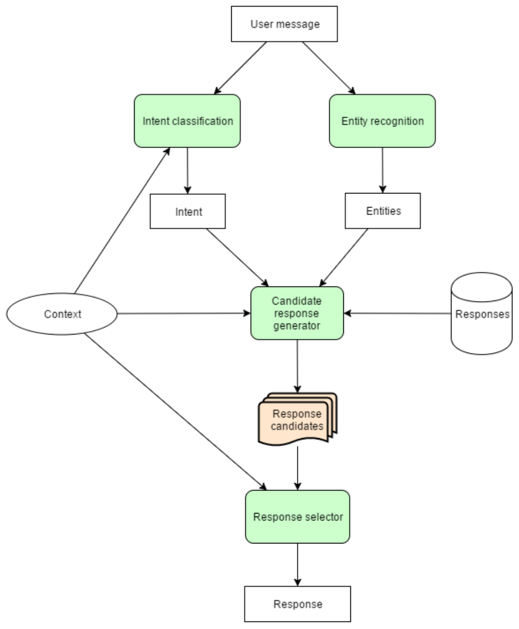


Figure.2 Chatbots Architecture [2]

Before making communication with a chatbot, the user on the instant messaging platform initiates the communication by adding the usernames on its IM account. After initiation, the signal with chatbot user ID forward to bot server using HTTP and finally bot push out welcome or out of service message and after that communication start and the user starts writing the query as shown in Figure.3

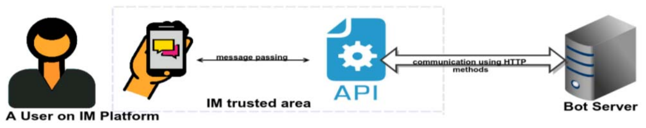


Figure.3 Chatbot working [2]

II. Chatbot using ML in Python

Chatbots are mainly developed is conversational dialog engine which is built in Python and able to makes it possible to respond based on the collections of all the known conversations. A chatbot is now designed in different languages such as English, Spanish and French.

An overview of typical input would be something as:

| ***user:*** Hi Good morning! How are you and What can I help you?

| ***bot:*** I am doing very well, thank you and what about you.

| ***user:*** Thanks and You welcome.

| ***bot:*** Do you like coats?

A chatterbot is the type of chatbot which can use install using the following basic guidelines at service side. The installation can be done from "PyPi" by running the
Pip install chatterbot

The basic configuration code is as follows.

```
from chatterbot import ChatBot

chatbot = ChatBot(
'Chat bot,
trainer='chatterbot.trainers.ChatterBotCorpusTrainer'
)

# english corpus
chatbot.train("chatterbot.corpus.english")

# Get a response to an input statement
chatbot.get_response("Hi, how are you today?")
```

Training data of chatbot comes with the data utility module which is used to train chatbot and currently it is based on English, Portuguese and Spanish languages and installation work are shown as follows.

```
.. _PyPi: https://pypi.python.org/pypi/ChatterBot

.. |Package Version| image::
https://img.shields.io/pypi/v/chatterbot.svg
:target: https://pypi.python.org/pypi/chatterbot/
.. |Requirements Status| image::
https://requires.io/github/gunthercox/ChatterBot/requirements.
svg?branch=master
:target:
https://requires.io/github/gunthercox/ChatterBot/requirements/
?branch=master
.. |Build Status| image:: https://travis-
ci.org/gunthercox/ChatterBot.svg?branch=master
:target: https://travis-ci.org/gunthercox/ChatterBot
.. |Documentation Status| image::
https://readthedocs.org/projects/chatterbot/badge/?version=sta
ble
:target:
http://chatterbot.readthedocs.io/en/stable/?badge=stable
.. |Coverage Status| image::
https://img.shields.io/coveralls/gunthercox/ChatterBot.svg
:target: https://coveralls.io/r/gunthercox/ChatterBot
.. |Code Climate| image::
https://codeclimate.com/github/gunthercox/ChatterBot/badges/
gpa.svg
:target: https://codeclimate.com/github/gunthercox/ChatterBot
.. |Join the chat at https://gitter.im/chatter\_bot/Lobby| image::
https://badges.gitter.im/chatter_bot/Lobby.svg
:target:
https://gitter.im/chatter_bot/Lobby?utm_source=badge&utm_
medium=badge&utm_campaign=pr-
badge&utm_content=badge

Platform: any
Classifier: Development Status :: 4 - Beta
Classifier: Intended Audience :: Developers
Classifier: License :: OSI Approved :: BSD License
Classifier: Environment :: Console
Classifier: Environment :: Web Environment
Classifier: Operating System :: OS Independent
Classifier: Topic :: Software Development :: Libraries ::
Python Modules
Classifier: Topic :: Communications :: Chat
Classifier: Topic :: Internet
Classifier: Programming Language:: Python :: 3.4
Classifier: Programming Language :: Python :: 3.5
```

The chatbot industry is still in the Era of development, but growing very fast.

Chatbot has the ability to understand the user says and it is able to choose or generate a response which can be based on the current input and on the context of the conversations.

Chatbot response can be static or dynamic. The simplest way of static response is that chatbot use the static responses from the templates such as The train time is <ft> hours where <ft> is a variable which can be computed at the time by the chatbot [3].

On the other hand, dynamic response is knowledge base response which is used to get potential responses and then score them to choose the better response. Chatbot generates responses using deep learning techniques to train generative model which is used to input and generate the answer. Keep in view of this need, there are millions of examples which are needed to learn by Chatbot to deliver responses.

Currently, the input is not enough to give the correct answer to the users and the model and implementation logic of chatbot along with the notion of context is important [4,5,6].

III. CHATBOT PROGRAMMING CHALLENGES

There are a lot of challenges which are associated with chatbots. Some of them are as follows.

A. Natural language processing

The first and foremost challenge [7] of the chatbot is to handle NLP issue by mastering their syntax. If we ask them that " what's the weather?". You will get an answer but what if we ask "Could you check the weather?" you might not get the proper answer. Such type of programming issues falls in natural language processing category which is a key focus for the companies like Facebook, Google with Deep Text and Syntax Net respectively.

B. Machine learning

Getting NLP is one aspect of designing and development of Chatbots while Machine Learning is another aspect of the Chatbot design and development. Our computer systems should able to learn the correct response should be which can be achieved with efficient programming with AI concepts [8].

IV. EXISTING PLATFORMS

The selection of platform selection is depended on the chatbot which is needed to develop. We have to decide that whether chatbot will be goal oriented, conversational or goal oriented with conversational abilities in chatbot?
A conversational chatbot [10] has the focus on the user conversations and it does not need to deeply understand that what the user says and there is no need to remember all the context of the conversation. Such types of chatbot can be used in entertainment. On the other hand, a goal-oriented chatbot is the most frequent kind of the chatbot for the business and it helps users to achieve tasks such as buying tickets, getting particular information or ordering grocery.
Chatbot platforms can be divided into three major categories. Which is:

- Nonprogramming chatbots
- Conversation-Oriented chatbots and
- Platforms by tech giants' chatbots.

A. Nonprogramming chatbots

The basic type of chatbot is not technically oriented platforms. Such type of codes is easy to develop without having strong programming skills and without having machine learning or the natural language processing expertises. The basic idea for such type of platform is to not worry about the technical details. There are a lot of on programming platforms and some of them are Chatfuel, ManyChat, and Motion.ai.

B. Conversational -Oriented- chatbots

In such type of chatbots, the users are able to communicate with the bot more proactively. These platforms use specification languages such as AIML (Artificial Intelligence Markup Language) which is used to model the user interactions. The examples give the code to get connected with the bot with AIML.

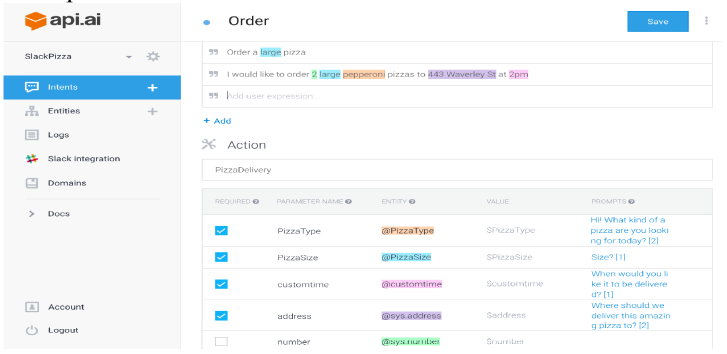
```
<aiml version="1.0.1" encoding="UTF-8">
<category>
<pattern>MY University Name is
*</pattern>
<template>
That is interesting that you have a
your University named <set
name="University"><star/></set>
</template>
</category>
<category>
<pattern>WHAT IS MY University
NAME</pattern>
<template>
Your dog's name is <get
name="University"/>.
</template>
</category>
```

C. Chatbots by tech giants

Tech giants such as Google develop Api.ai, Facebook develops Wit.ai, Microsoft develops LUIS, Amazon develops Lex and IBM develops Watson.

Api.ai Chatbot

Contexts and Intents are the key concepts of the model of chatbot using Api.ai as shown in Figure.4. Intents are used to create the links that what user is saying and what action should be taken by the chatbot while the context is the string values which are used to differentiating the requests which can have different meaning depending on requests. When an Api.ai received a request from the user, it is first classified to determine whether it is matched with the intents while Api.ai proposes default fall back intent which is needed to deal with the request that does not match with the user intents.



Order

Order a large pizza
I would like to order 2 large pepperoni pizzas to 443 Waverley St at 2pm
Add user expression...

Action

PizzaDelivery

REQUIRED	PARAMETER NAME	ENTITY	VALUE	PROMPTS
✓	PizzaType	@PizzaType	$PizzaType	Hi! What kind of a pizza are you looking for today? [2]
✓	PizzaSize	@PizzaSize	$PizzaSize	Size? [1]
✓	customtime	@customtime	$customtime	When would you like it to be delivered? [1]
✓	address	@sys.address	$address	Where should we deliver this amazing pizza to? [2]
☐	number	@sys.number	$number	

Figure.4 Api.ai chatbot

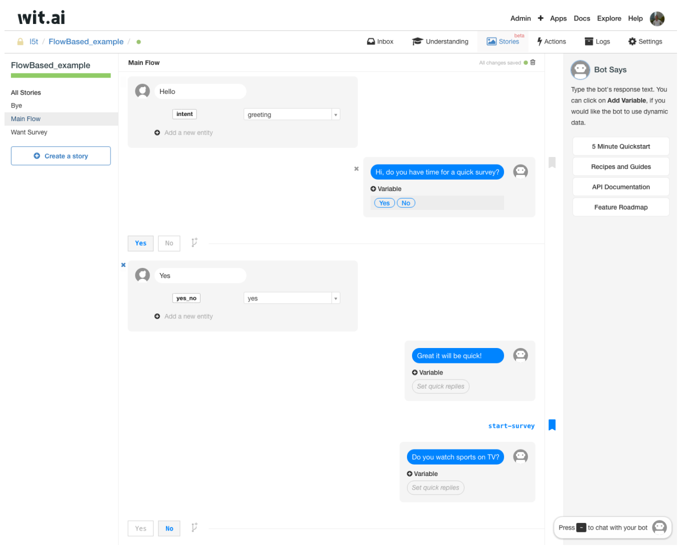
Api.ai chatbot features

- Using Intents and Contexts, Api.ai can propose a very powerful way of the modeling of large and complex systems.
- Chatbot has the ability to handle the code proactively and decreases server-side coding.
- APi.ai offer one-click integration with the several platforms such as Twitter, Facebook.

On the other hand, training set is still a big issue and it is not possible to match the intents when the context is there.

Wit.ai Chatbot

Wit.ai work on stories as a key concept to model the behavior of Chatbot. Every story gives an example of all the possible conversation and it is based on entity concept only instead of intents. In this bot the developers basically teach the Wit.ai using examples and when a user writes about the similar type of object. Wit.ai will able to process the request and get the extract entities and apply developer logic on the scenario. Wit.ai chatbot is shown in Figure.5.



Hello
intent: greeting

Hi, do you have time for a quick survey?
Yes / No

Yes
yes_no: yes

Great it will be quick!

Do you watch sports on TV?

Type the bot's response text. You can click on **Add Variable**, if you would like the bot to use dynamic data.

Figure.5 Wit.ai chatbot

In this type of chatbot, a story can be seen as a user intents graphs. Chatbot can have branches that are used to trigger on specific conditions which can be existence or these can be not specific values which are extracted from the user input. In such way, it leads to define the conversational flow and moreover it defines the bookmark mechanism which allows jumping between the intents and stories.
To interact with the server side, it use Bot send commands which is a call to functions and it is possible to set the role of phrases such as in " I want to go London, England from Dhaka, Bangladesh on 30 August", in this way you can state that the first is departure city and second is the destination.
Wit.ai allows defining the own entities or able to use the predefined entities [10].
Wit.ai is based on the webhook integration in which information "bot sends" command into web service and get the results from it. On the other hand, on the server side, we are going to expand or create the context of the conversation. The outcome sent to Wit.ai can add, modify and delete the context variables which are used in chatbot side.

Wit.ai chatbot features

This type of chatbot offer lot of key advantages which are as follows.

- The story concept is very useful and powerful.
- Branches lead to better control the conversation also conditions on the actions.

On the other hand, it is not feasible to handle stories data and even stories are powerful concepts, there are the cases, there are cases where data flow can misunderstand the requests.

V. Limitation and Future of NLP and Machine Learning

As per our discussion, it is quite clear that chatbot needs to provide vast logic and linguistic resources which are input, output and entities phrases. Chatbot with complex queries handling need high attention in using singular and plural forms, need to take care of synonyms, hyponyms, and finally, the sentimental analysis should be done carefully [11-12].

VI. Conclusion

A chatbot is a rising trend and chatbot increases the effectiveness of business by providing a better experience with low cost. A simple chatbot is not a challenging task as compared to complex chatbots and developers should understand and consider the stability, scalability and flexibility issues along with high level of intention on human language.
In short, Chatbot is ecosystem and moving quite fast and with the passage of time new features are added in the existing platform. Recent advancements in the machine learning techniques may able to handle complex conversation issue such as payments correctly.

References

[1] X. Li, J. Niu, M. Karuppiah, S. Kumari and F. Wu, “Secure and Effi-cient Two-Factor User Authentica-tion Scheme with User Anonymity for Network Based E-Health Care Applications”, Journal of medical systems, Vol.40, No.12, (2016), pp.268.

[2] M. Karuppiah, S. Kumari, X. Li, F. Wu, A.K. Das, M. K. Khan, R. Saravanan and S. Basu, “A dynamic id-based generic framework for anonymous authentication scheme for roaming service in global mobility networks”, Wireless Personal Communications, Vol.93, No.2, (2016), pp.383–407.

[3] M. Karuppiah, “Remote user authentication scheme using smart card: a review”, International Journal of Internet Protocol Technology, Vol.9, No.2–3, (2016), pp.107–120.

[4] M. Karuppiah, S. Kumari, A.K. Das, X. Li, F. Wu and S. Basu, “A secure lightweight authentication scheme with user anonymity for roaming service in ubiquitous networks”, Security and Communication Net-works, Vol.9, No.17, (2016), pp.4192–4209.

[5] A. Graesser et al., ―AutoTutor: an intelligent tutoring system with mixed-initiative dialogue,‖ Education, vol. 48, no. 4, 2005, pp.612-618.

[6] R. Hubal et al., ―Avatalk virtual humans for training with computer generated forces,‖ presented at the Proceedings of CGF-BR. Institute for Simulation and Training, Orlando, FL.

[7] D. Field. The Senior Companion: a Semantic Web Dialogue Aamas. [Online]. Available: http://www.ifaamas.org/Proceedings/aamas09/pdf/06_Demos/d_07.pdf

[8] V. Aleven, O. Popescu, and K. Koedinger. [Online]. Available: http://pact.cs.cmu.edu/koedinger/pubs/Aleven%20Popescu%20Koedinger%20aied01.pdf

[9] C. Lee, S. Jung, S. Kim, and G. Lee. ―Example-based dialog modeling for practical multi-domain dialog system,”Speech Communication, vol. 51, 2009.

[10] The Stanford NLP (Natural Language Processing) Group. [Online] Available: http://nlp.stanford.edu/software/CRF-NER.shtml

[11] G. Pirrò and J. Euzenat: ―A feature and information theoretic framework for semantic similarity and relatedness,‖ in Proc. of the 9th International Semantic Web Conference (ISWC2010), 2010, pp. 615-630.