INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 5, Issue 2)
Available online at: www.ijariit.com

Chatbot generation and integration: A review

Aarsh Trivedi
trivediaarsh98@gmail.com
Parul University, Vadodara, Gujarat

Vatsal Gor
vatsalkgor@gmail.com
Parul University, Vadodara, Gujarat

Zalak Thakkar
zzalak.thakkar06@gmail.com
Parul University, Vadodara, Gujarat

ABSTRACT

In the modern era of technology, chatbot is the next big thing in the domain of conversational services. A chatbot is a virtual person who can effectively talk to any human being using interactive textual as well as verbal skills. There are many statistics available which suggest that integration of chatbot in any business as a part of their customer service increases the business progress and customer satisfaction exponentially. Hence it becomes crucial to understand the crux of the chatbot technology. Owing to extensive research in this field, there are numerous methodologies available to create a conversational entity. It becomes quite confusing to decide a perfect method to generate conversational agent for the desired purpose. Also, generation of chatbot is one issue and successful integration is in itself another problem which is many times overlooked. This paper provides some valuable insights on how to generate as well as how to integrate a chatbot.

Keywords— *Chatbot, AIML, ChatScript, REST API, Human-computer interaction*

1. INTRODUCTION

Studies have shown that customer services are one of the major aspects of customer satisfaction in any business. Because of poor customer services US companies have lost more than $62 billion annually. [1]. Chatbots can be the digital age solution to address the improved customer service issue. There are major studies that support the acceptance of chatbot in the customer service business. In general, this provides the motivation to explore the fields of Chatbots and Human-Computer Interaction. According to the definition provided by Google, a chatbot is a computer program designed to simulate conversations with human users, especially over the internet. Many computer scientists acknowledge the inception of the idea behind Human-Computer Interaction to Alan Turing who asked, "Can Machines Think?" in 1950. [2] With the advancements in Computational Sciences and Artificial Intelligence, adoption of chatbot has increased exponentially, especially with the adoption by different social media and services giants. For instance, Facebook, in their last year's annual developer conference announced that its bot platform has been used to create more than 300,000 bots. The major aspects of a paradigm shift to chatbot technology are because of major optimization of time and cost factors of the business. A study released by a UK research company, Juniper, predicts that chatbots will help business firms save more than $8 billion annually by 2022. [3]

There are different kinds of chatbots available in the market which can be majorly categorized as follows: Text-based chatbots, Voice-based virtual personal assistants and chatbots supporting both the methods of interaction.

In this paper, we have focused on technologies available related to text-based chatbots only. And hence, the term chatbot refers to text-based virtual assistants only in this paper.

This paper provides brief summaries of currently available technologies for creation to integration. This paper is segregated in 2 major sections.

The first section discusses the generation of the chatbot using different technologies. We will be discussing following methodologies and their usage in the design of different types of chatbots available.

Also, we will be comparing these technologies to provide a better understanding of their respective use according to the requirements.

The second section discusses the different integration methodologies. To design a chatbot is one problem but to integrate it to their respective platforms is a different issue in itself. So, to provide a better understanding of different methodologies we have reviewed widely used methodologies available nowadays. Also, we have provided some insights of the preference for specific methodology in a specific situation.

2. CHATBOT GENERATION

Chatbots are used for many different purposes, for example, customer service, general purpose chatbots, personal assistance, FAQs chatbot etc. For many of the application, the knowledge of chatbot varies respectively. Hence there are many different methods available to generate a chatbot which is efficient for a specific kind of application.

In this paper, we will be providing an overview of all the methods and their possible application domains. We will me

majorly categorizing the generation methods into following respective categories:

- AIML
- ChatScript
- Heuristic approach
- Online Platforms

2.1 AIML

In different methodologies of generating chatbot, pattern recognition aims to model a computer system which imitates human to human dialogues. The Pattern Recognition is based on representative stimulus-response type blocks, which the user enters a sentence (stimuli) and the software makes an output (response) according to the user input. [4] If the pattern recognition paradigm is used to create the chatbot, AIML is used to develop chatbot's knowledge base. ALICE was the first Chabot developed by using AIML as a backend. [4].

As AIML is an interpreted language, it has different interpreters associated with it which will be briefly described further in this document. Because of its XML based markup approach, AIML makes the task of dialogue modeling very easy. For modeling of patterns of conversations, AIML defines data objects which are known as AIML objects. Like XML and other markup languages, AIML also follows the following structure: a start tag (<command>) and an end tag (</command>) enclosed in between the list of parameters.

The basic unit of AIML script is called category tag which is formed by user input patterns and chatbot responses according to the input. It defines a unit of knowledge that a bot has. Tag <pattern> and <template> represent user input pattern and the chatbot response for the pattern respectively. As AIML uses pattern matching wildcard characters "_" and "*" are of utmost importance in AIML script. Precedence is given to the "_" symbol than "*".

A brief description of AIML tags is provided below in the table.

Table 1: Basic AIML tags

Tags	Use
<aiml>	Root tag of the AIML Document
<category>	Represents basic unit of AIML Dialogue
<pattern>	corresponds to the user input pattern
<template>	corresponds to the bot response to the specific pattern
<star>	captures a particular text fragment in the user input
<srai>	provides ability to target different patterns for single template
<random>	used for responding to the user in random ways
	possible responses are established using this tag
<set>	used to set AIML variables
<get>	used to get AIML variables
<that>	tag tells the system to analyze the last sentence presented by the bot
<condition>	is used to compare AIML variable with a desired value
<bot>	define some properties of the bot

Use of the above important tags is elaborated using the example shown in table 2.

2.2 ChatScript

One of the major drawbacks of AIML is it has no good support for data organization of large quantities. Also, AIML is a rule-based matching a sequence of words to generate either a completely canned response or substitution of input words and is extremely redundant in handling large knowledge bases.

Table 2: AIML example

```
<aiml version="1.0.1" encoding="UTF-8"?>
<category>
        <pattern> HELLO BOT </pattern>
        <template>
        <random>
                <li> Hi! Nice to meet you </li>
                <li> Hello, How are you? </li>
                <li> Hello! </li>
        </random>
        </template>
</category>
<category>
        <pattern> MY NAME IS * </pattern>
        <template>Hello <set
name="nameUser">    <star/> </set> </template>
</category>
<category>
        <pattern> GOOD NIGHT </pattern>
        <template>Good night<get
name="nameUser"/></template>
</category>
<category>
        <pattern> BYE * </pattern>
        <template> <srai> GOOD NIGHT <star>
</srai> </template>
</category>
<category>
        <pattern> BOT'S PROPERTIES
</pattern>
        <template>
                <bot name="age"/>
                <bot name="gender"/>
                <bot name="nationality"/>
        </template>
</category>
</aiml>
```

To overcome the above issues with AIML, chat script was introduced by Bruce Wilcox in 2011. ChatScript is a combination Natural Language engine and dialog management system designed initially for creating chatbots but is currently also used for various forms of NL processing.[5]

ChatScript follows simple visual syntax and discards the need to match all the words of the input. In that script, there are different kinds of keywords which have their own specific tasks assigned to it. Some of the keywords are overviewed below in the table.

Table 3: ChatScript keywords

Keyword	Task
s:	Matches user statement with its corresponding output
?:	Matches user question with its corresponding answer
u:	Matches user input with its corresponding output irrespective of the type of sentence
t:	Defines a sentence to say within a topic without waiting for user input.
[]	In any continuous sequence of bracketed phrases, the bot may select one at random

()	Matches the user input
([ ])	Brackets [] inside the pattern mean disjunction
^noerase()	It is a macro which tells bot not to discard the rule immediately
^repeat()	Indicates to repeat the output explicitly
topic:	Defines a topic for conversation. the topic name always starts with ~
*	Wildcard character that matches zero or more words
_*	Matches zero or more characters and stores it in the short term memory of bot
_0 to _9	Access the short term memory of bot.
$ variables	Store the value in bot's long term memory.

Besides above mentioned most used keywords, chat script also allows decision making using conditional statements which inculcate the usage of chat script variables and conditional logic.

2.3 Heuristic approach

With the inception of the idea of thinking machines, computer intelligence has become one of the major research areas. In the late 1950s, researcher John McCarthy introduced the term "Artificial Intelligence" in a conference. Since then there has been immense research and progress in the field of AI is going on. As a product of the research, different AI methodologies have also been applied in the generation of intelligent conversational entities.

The methods discussed in topics 2.1 and 2.2 were rule-based and cannot generate responses on their own. With the inception of the idea of computer intelligence, many advancements have taken place to actually understand the language and create humans like a dialogue between human and computer or computer and computer from silos of conversational data which in AI term known as training sets.

There are many different methodologies to create chatbots. Each of them has its pros and cons and applications in specific areas. Widely used techniques and methodologies are summarized below.

2.3.1 Long short term memory: A branch in different AI methodologies is Neural Networks. They are a framework which is inspired by biological neural network formed inside the human brain.

Although being comparatively a young research topic, the concept of Artificial Neural Networks has been in discussion since the late 1950s. And there are many notable works done in this era.

One of the simplest implementations of neural networks is a feedforward neural network where information moves only in a single direction, forward to produce the output through different layers of artificial neurons.

Feed forward neural networks may consist of single or multiple layers of artificial neurons. These neurons process the unit of information with different mathematical models. To describe the working verbally would be an exhaustive task and would be non-intuitive to visualize. Hence, to visualize and summarize the working, a graphical representation is given below:

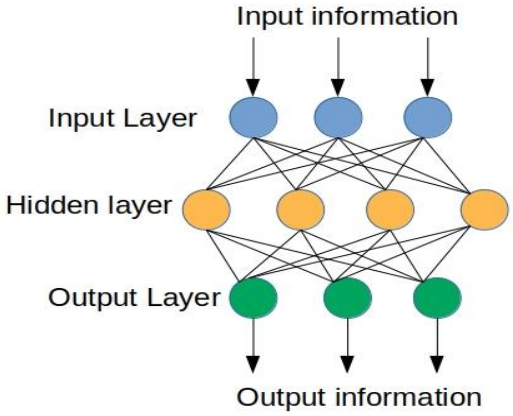

Fig. 1: An Abstract Neural Network

Unlike the feedforward neural network, LSTMs inculcates use of recurrent neural networks in which information flow is not linear.

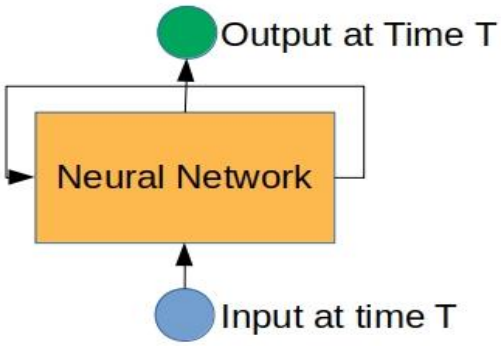

Fig. 2: Recurrent Neural Network Model

LSTMs are preferred where there is a need to deal with sequential data or data with a temporal relationship. However, LSTMs have their own disadvantages: they are comparatively slow and require a large high-quality dataset in order to get decent results.

There are many different frameworks available for creating chatbots using heuristics apart from LSTMs like Seq2Seq model, Seq2Seq with Reinforcement learning, etc. But to even summarize them would require more than one review paper and hence are just listed here for reader's enlightenment.

2.4 Online Platforms

For users who are not aware of the programming fundamentals and come from a non-technical background who have specific need of chatbot without any deep technical knowledge, the best option is to opt for a chatbot providing service that is available on the internet.

As there is a major increase in usage of chatbot for commercial and enterprise applications, there are many big companies who have come with such solutions. We have listed down some of the popular services provided by trustworthy IT giants.

- Dialog Flow by Google
- Wit.ai by Facebook
- LUIS by Microsoft

Documentation for using the above-mentioned services is available on their respective websites.

3. CHATBOT INTEGRATION

A good chatbot generated but not integrated properly can create major issues in any respective application. Hence integration is also important and should be done properly.

There are many ways one can integrate a Chabot service. We have generalized those methodologies into following three categories.

- Integration using API
- Manual Integration
- Third-party Integration

3.1 Integration using API

An API is an abbreviation of Application Programming Interface. It is a set of subroutine definitions and communication protocols and tools for building software [6]. In essence, it can be said that API is a software architecture style, which provides a framework to design software which is able to communicate across devices, platforms and other software.

There are many methodologies to implement API architecture. One of the majorly used and the current standard is the RESTful API structure. REST API is briefly explained as follows.

REST is an acronym for Representational State Transfer. It defines the constraints to be used to develop and design a Web service. It provides interoperability between computing devices.

REST API uses HTTP requests such as GET, PUT, POST, DELETE, TRACE, CONNECT, OPTIONS, PATCH, etc. to provide communication and manipulation of data which resides on the Web server. It ensures communication between the Web Server and Client.

Using RESTful API, it is possible to make a single chatbot instance up and running on multiple different websites.

A chatbot service can be made by conforming to RESTful architecture and that service can then be integrated into multiple websites where the respective service is required.

The overall structure of RESTful web service can be visualized as provided in the diagram below:

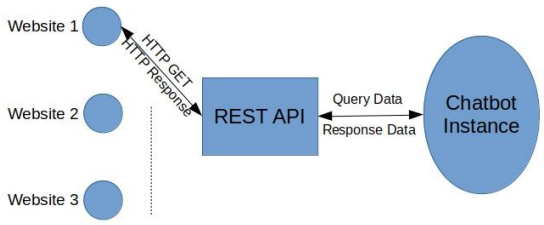

Fig. 3: REST API working model

3.2 Manual integration

Usage of API is highly recommended if there are many instances of many websites. However, if the user has only a single website and does not want any fancy integration architecture, then manual integration of the chatbot would be the best option.

In this type of integration, chatbot logic, website logic as well as server side logic resides on the same server. Hence, this is the simplest kind of integration that a chatbot can have.

3.3 Third-Party integration

If API or manual integration is technically infeasible for the user then the user can opt for any third party chatbot providing service that also provides easy integration methods. For example, all the web platforms enlisted in topic 2.4 have thorough instructions on how to integrate chatbot developed on their platform.

4. CONCLUSION

After the emergence of the internet and then to the era of web application and services, the use of intelligent conversational entity has increased exponentially and it is still trending. So it becomes crucial to be aware of the different technological paradigms related to generation as well as the integration of these intelligent agents. There are many methodologies related to the generation of chatbots ranging from simple pattern matching to complex neural networks and other heuristic solutions. Each of the solutions has its specific applications in their respective domains. In this paper, we have briefed about generation methods and their respective usage domains. Also, it would not be a good idea if you do not give enough importance to integrate your conversational entity. Hence, we have summarized different paradigms available for integration and also have provided some of the insights on where to use which paradigm.

5. REFERENCES

[1] Bucholtz, C. (2016). The $62 billion customer service scared away [INFOGRAPHIC] - Blog | New Voice Media. [online] New Voice Media. Available at: https://www.newvoicemedia.com/blog/the-62-billion-customer-service-scared-away-infographic

[2] Turing, A.M., 2009. Computing machinery and intelligence. In Parsing the Turing Test (pp. 23-65). Springer, Dordrecht.

[3] Juniperresearch.com. (2017). Chatbots, a Game Changer for Banking & Healthcare. [online] Available at: https://www.juniperresearch.com/press/press-releases/chatbots-a-game-changer-for-banking-healthcare

[4] Marietto, M.D.G.B., de Aguiar, R.V., Barbosa, G.D.O., Botelho, W.T., Pimentel, E., França, R.D.S. and da Silva, V.L., 2013. Artificial intelligence markup language: a brief tutorial. arXiv preprint arXiv:1307.3091.

[5] En.wikipedia.org. (2018). ChatScript. [online] Available at: https://en.wikipedia.org/wiki/ChatScript

[6] En.wikipedia.org. (2019). Application programming interface. [online] Available at: https://en.wikipedia.org/wiki/Application_programming_interface.

