Derek: A chatbot that shows intelligence in behavior using NLP

[1]Ameya Vichare, [2]Yashika Shrikhande, [3]Ankur Gyani, [4]Nilesh Rathod
[1]Student, IT Department, RGIT, Mumbai
[2]Student, IT Department, RGIT, Mumbai
[3]Student, IT Department, RGIT, Mumbai
[4]Assistant Professor, IT Department, RGIT, Mumbai

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 paper, 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, ALICE and Siri. We evaluate what we can take from each of the systems and use in our proposed system. Finally, we discuss the proposed system. Our system can be used on Android OS in a similar manner to Siri for effective information retrieval just by speaking various queries.

Keywords— *natural language processing, AIML, chatbot.*

*****

I. Introduction

The need of chatbot systems has become important with the ubiquitous use of personal systems which wish to communicate and the desire of their makers to provide natural language for them to communicate with humans [1]. A chatbot system is a software program that interacts with users using its own language called the natural language. The purpose of a chatbot system is to simulate a conversation with a human which is so human-like that the person gets fooled into believing that he's talking with a human [12]. The chatbot architecture includes a language model and pattern matching algorithms to emulate informal conversation between a human and a system using natural language [3][7].

Initially, developers built and used chatbots for entertainment purposes and used simple keyword matching algorithms to find an answer to a user query, such as ELIZA (Weizenbaum, 1966, 1967). The seventies and eighties, before the arrival of graphical user interfaces, saw rapid growth in text and natural-language interface research, e.g. Cliff and Atwell (1987), Wilensky (1988). Since then, a plethora of new chatbot architectures have been developed, such as: MegaHAL (Hutchens, 1996), CONVERSE (Batacharia, 1999), HEXBOT (2004) and ALICE (2007) [1][2].

In our project we are using AIML. AIML is an XML- based language which can be used for interaction between chatbots and humans. The atomic unit in AIML is category, which consists of attributes called as pattern and template.

We are also using a speech to text/text to speech recognition to recognize the Indian accent more efficiently. During construction of the soundbase of the chatbot, the following can help heighten its speech recognition rate: the soundbase should be built to match user speech input based on non-native-specific, area-specific, age-group-specific. gender-specific parameters [6][14].

II. Existing systems

A few chatbots with useful applications in our system are presented. We begin by discussing the ELIZA chatbot system architecture and the working. Then we continue discussing about other systems like ALICE [14] and Siri [5].

ELIZA

It was one of the first chatbots, designed in 1966. It acts like a therapist by rephrasing statements of the user and posing them back as questions. ELIZA works by simple parsing and substitution of key words into reframed phrases. People get emotionally caught up by ELIZA's confident replies forgetting that it's a machine [9]. ELIZA was written at MIT by Joseph Weizenbaum between 1964 and 1966. The most famous application of ELIZA was DOCTOR, a simulation of a Rogerian psychotherapist. It had almost no information about human thought or emotion and still DOCTOR sometimes provided a startlingly human-like interaction. DOCTOR might prompt a generic response, for example, responding to "I am sad" with "Why are you sad?", a possible response to "My sister hates me" would be "Who else in your family hates you?". ELIZA was taken seriously by several of its users, even after Weizenbaum explained to them that it's not a human [15].

ALICE

ALICE (Artificial Linguistic Internet Computer Entity) is inspired by ELIZA. It is a natural language processing chatbot—a program that engages in a conversation with a human by applying some pattern matching rules to the user's query and then generating an appropriate response [6][14]. It has won the Loebner Prize, three times. However, the program is unable to pass the Turing test, as even the casual user will often expose its flaws in short conversations. ALICE is consists of two parts, Chatbot engine and Language Model. Language model is stored in AIML (Artificial Intelligence Mark-up

Language) files. As discussed earlier, AIML consists of pattern and templates. Pattern is used for matching of the user query whereas the response will be provided in template. Following is an example: <category><pattern> how are you </pattern> <template>I am fine</template></category>. In this example, user's input is matched against the pattern and when a match is found, the pattern matching process stops, and the template that belongs to that category is processed by the interpreter to construct the output. AIML has wildcards – denoted by star – which are used for matching a string of one or more words in the input. AIML also has srai (symbolic reduction) tags, which can be used for reducing the amount of pattern and templates. Using srai a number of questions can be grouped and a similar response can be provided in the template [6][7].

SIRI

Siri was developed by Apple. Siri is a computer program that works as a virtual personal assistant and knowledge navigator. The feature uses a natural language user interface to answer questions, give suggestions, and perform actions by using a set of Web services. The software adapts to the user's individual language usage and individual searches with continuous use, and returns results that are personalized. Siri was originally introduced as an iOS application available in the App Store by Siri, Inc., which was acquired by Apple on April 28, 2010. Siri has been an integral part of iOS since iOS 5 and was introduced as a feature of the iPhone 4S on October 14, 2011 [5][8]. Siri evaluates your speech locally on your device. A recognizer installed on your phone communicates with a server in the cloud to determine whether the query can be best handled with the information in its databases or if it must connect to the network for an advanced search [13].

III. Issues In Existing Systems

We discuss the issues of the existing systems (discussed in previous section) in this section.

After close observation of A.L.I.C.E chatbot system it is apparently clear that (a) ALICE does not save the history of conversation and (b) ALICE does not truly understand what you said; it gives you the responses from the knowledge domain stored in her brain.

After close observation of voice-to-text agents like Siri it is clear that (a) At times Siri can't understand what we told her to do and (b) Siri cannot answer for complex queries. For eg, Siri knows "Where was Abraham Lincoln born?". It also knows "What is the population of Kentucky?" but it would fail if someone asks "What is the population of the place where Abraham Lincoln was born?".

IV. Proposed System

We propose a system which will work as an application and give users information about different kinds of sports present in the world. This application will work using a pattern matching algorithm using depth first search (DFS). It will also convert user's queries from speech to text using a speech-to-text converter and back to speech as an output to the user. Firstly, the user will speak his/her query out loud; the application will convert this query from speech into text. The system finds out what the user actually wants by retrieving the semantic meaning of the query. Then the system will pass on this semantic text as input to the pattern matching algorithm. The pattern matching system will make use of knowledge that is already fed into the databases in order to generate a response. Once this response is generated we will convert it back into speech and the application will read the content out.

V. DESIGN

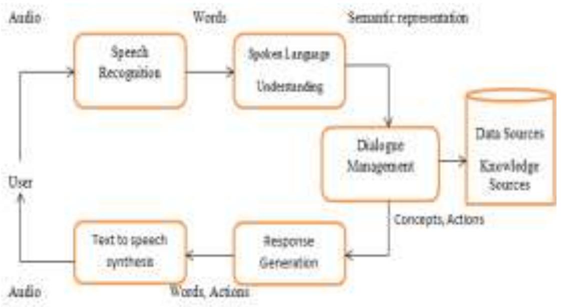


Fig 1: Block Diagram of system

The user will speak out his query on his/her phone using the application system. The application sends this speech as input to the speech recognition module which will convert the speech into text. This will be implemented using Google speech recognition API. There are two ways to do so, using RecognizerIntent or by creating an instance of SpeechRecognizer [6]. Normalization is applied for each input first removing all punctuations and it is then converted to uppercase. E.g.: Can you, or can you not do this? is converted to: CAN YOU OR CAN YOU NOT DO THIS. The spoken language understanding module will understand the meaning of the query using semantic representation. This is done so the application system can recognize what the user actually wants. Semantic representation of a query will help the system realize the connotation as well as denotation of the query. Semantic representation is implemented using either of two methods, orthographic similarity or phonetic similarity [10]. Confidence levels, placed in arrays, are assigned to queries to know if what the user spoke and what the system understood are the same thing. Confidence ranges from 0 to 1, with 1 being most desirable. This refined form is then fed to the dialogue management module. This module will make use of the knowledge already fed to the knowledgebase as well as the various web services in order to generate a response which will be appropriate to the query asked. The knowledge base will be fed a set of prior information about sports related questions using AIML. In the response generation module, the pattern matching algorithm uses DFS to match a query and it continues till a match is found [11]. The interpreter processes the template that belongs to that particular category and generates the output. System appends the constructed output as input to the text-to-speech synthesis module. The response in form of speech which is generated will get fed to the user as input in the form of words. The conversion of text back to speech is implemented using Google text-to-speech synthesis. We use Google TTS synthesis by importing the package android.speech.tts [6]. System prompts the user to ask if he has another query. If the user has another query, the system will repeat the entire process again otherwise the system will terminate processing.

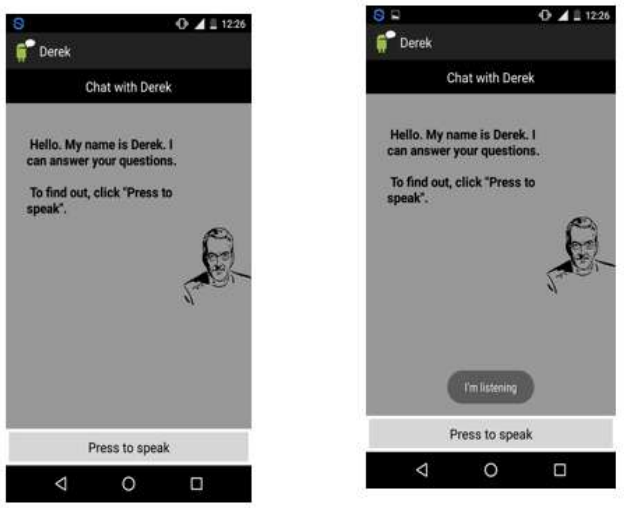
VI. IMPLEMENTATION

The main module initially calls the setSpeakButton() to initialize the GUI components. The main module also calls the createRecognizer() to initialize the speech recognizer. An instance of myTts is created in order to initialize text to speech. An instance of bot is also created in order to connect to the online database using bot = new Bot(). The processAsrReadyForSpeech() method is called to show a “I’m listening” message to the user. The application then tries listening to the user’s query. If query can’t be recognized, an error message is given back to the user using processAsrError() method. Error messages include audio recording error, client side error, insufficient permissions, network related error, network operation timeout, recognition service busy, server sends error status, no matching message, input not audible. If there are no errors, the main module calls the processAsrResults() method which will generate the XML query by replacing spaces with a ‘%20’. An initiateQuery() method is called which will send in the query to the online server using the link ‘http://www.pandorabots.com/pandora/talk-xml?input=" + query + "&botid=’. The onDestroy() method is called to shut down the TTS engine after the user has asked all the queries.

VII. SYSTEM IN ACTION

Dialogue management module

The purpose of this module is for taking the input from the user and converting it into text and then back to speech for the output. The application sends speech as input to the speech recognition module which will convert the speech into text. This is done so the application system can recognize what the user actually wants. The spoken language understanding module will understand the meaning of the query using semantic representation. Semantic representation of a query will help the system realize the connotation as well as denotation of the query. This refined form is then fed to the dialogue management module. This module will make use of the STT and TTS engines.



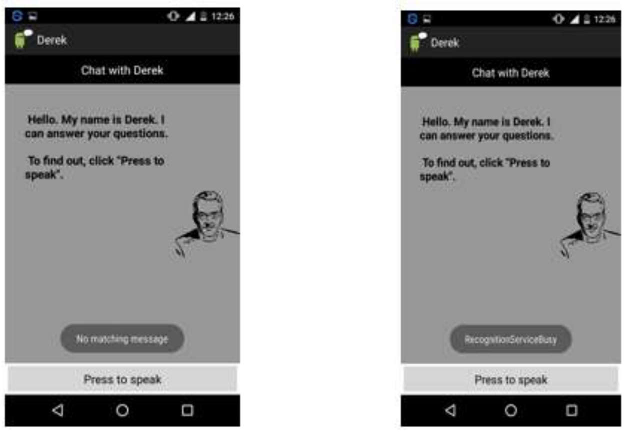
Hello. My name is Derek. I can answer your questions.

To find out, click "Press to speak".

Fig 2: Home screen of chatbot system

Error detection module

This module makes use of various constraints to detect errors that the user may encounter while using the application. Error messages include audio recording error, client side error, insufficient permissions, network related error, network operation timeout, recognition service busy, server sends error status, no matching message, input not audible. These messages are displayed to the user as a pop up on the Android device.



Hello. My name is Derek. I can answer your questions.

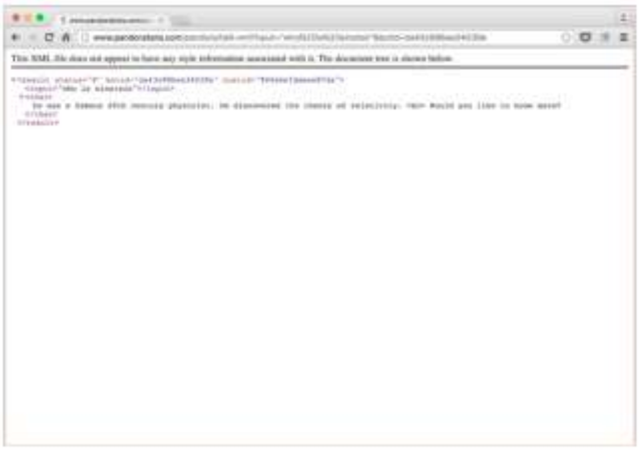
To find out, click "Press to speak".

Fig 3: Error detection module of chatbot system

XML query processing module

The third module is for the query processing. If there are no errors encountered by the user, this module calls the processAsrResults() method which will generate the XML query by replacing spaces with a ‘%20’. An initiateQuery() method is called which will send in the query to the online server using the link ‘http://www.pandorabots.com/pandora/talk-xml?input=" + query + "&botid=’

The output of this module is then sent to the TTS engine.



www.pandorabots.com[illegible]

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<result status="0" botid="[illegible]" custid="[illegible]">
  <input>Who is einstein</input>
  <that>
    He was a famous 20th century physicist; he discovered the theory of relativity. [illegible] Would you like to know more?
  </that>
</result>
```

Fig 4: XML query processing process

Response generation and pattern matching module

In the response generation module, the pattern matching algorithm uses DFS to match a query and it continues till a match is found. The interpreter processes the template that belongs to that particular category and generates the output. System appends the constructed output as input to the text-to-speech synthesis module. The response in form of speech which is generated will get fed to the user as input in the form of words or actions. System repeats this process till client closes the session.

VIII. CONCLUSION

To summarize, we have learned about all the existing systems which included ELIZA, ALICE and Siri. 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. ALICE helped us understand how we can make use of AIML in our system. Finally, Siri helped us understand the limitations of language processing in speech to speech agents. We decided to build a system to implement natural language processing on Android OS. In this system, the user will be prompted to ask a query. The user will speak out his query on his/her phone using the application UI. A range of questions and their responses will be coded into AIML and stored into the database. When the user will ask a query, this query will be matched against the various patterns present in the database and the template corresponding to that pattern will be returned in form of speech to the user.

REFERENCES

[1] B. Abu Shawar and E. Atwell, "Chatbots: are they really useful?," LDV-Forum 2007, Vol.31, No.50, 2007.

[2] B. Abu Shawar and E. Atwell, "Using the corpus of spoken afrikaans to generate an afrikaans chatbot," SALALS Journal: Southern African Linguistics and Applied Language Studies, pp. 283–294, 2003.

[3] P. Langley, "Intelligent Behavior in Humans and Machines," Advances in Cognitive Systems , Vol. 3, No. 12, December 2012.

[4] R. Berwick, "Learning structural descriptions of grammar rules from examples" Proceedings of the Sixth International Conference on Artificial Intelligence, pp. 56–58, 1979.

[5] Shuen-Shing Lee, "The Making of Task-Specific Chatbots for English Conversation Practice: A Preliminary Proposal" in Globalization and Digitalization: Pedagogical Challenges and Responses. Ed. Chi- lung Wei. Taipei: Crane Publishing. pp. 163-177, 2012.

[6] Michael F. McTear, Zoraida Callejas, "Voice Application Development for Android," pages 269-335, UK: Packt Publishing Ltd, 2013.

[7] D. Jurafsky and J. H. Martin, "Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition," pages 86-105, Upper Saddle River: Pearson Education, 2009.

[8] R.S. Russell, "Language Use, Personality and True Conversational Interfaces," Project Report, AI and CS, University of Edinburgh, Edinburgh, 2002.

[9] J. Jia, "The Study of the Application of a Keywords-based Chatbot System on the Teaching of Foreign Languages," University of Augsburg, Augsburg, Germany 2002.

[10] S. Han and Y. Kim, "Intelligent Dialogue System for Learning," presented at International Conference on Computers in Education, Seoul, Korea, 2001.

[11] M. A. Pasca and S. M. Harabagiu, "High Performance Question/Answering," presented at Annual ACM Conference on Research and Development in Information Retrieval, New Orleans, LA, pp. 366-374, 2001.

[12] J. Jia, "The study of the application of a web-based chatbot system on the teaching of foreign languages," In Proceedings of the SITE 2004 (The 15th annual conference of the Society for Information Technology and Teacher Education), pages 1201-1207. AACE press.

[13] Steve Wozniak Complains About Siri; Says Apple Ruined it, But Did They Really? (2012, June 15). Retrieved August 7, 2015, from http://www.decryptedtech.com/editorials/steve-wozniak-complains-about-siri-says-apple-ruined-it-but-did-they-really

[14] Be Your Own Botmaster - 2nd Edition, ALICE A.I. Foundation. (2005). Retrieved August 10, 2015, from http://www.pandorabots.com/pandora/talk?botid=f5d922d97e345aa1

[15] J. Weizenbaum, "ELIZA A Computer Program For the Study of Natural Language Communication Between Man And Machine," Communications of the ACM, Vol.9, No.1, September 1995.