Artificial Intelligence for Open Source Dialogue Based Natural Language Processing System

Shakti Phartiyal

Abstract— This paper is an overview of 'Artificial Intelligence and Natural Language Processing' technologies along with 'Speech to Text' and vice versa to build a dialogue-based system.The objective of this paper is to idealize the implementation of Artificial Intelligence along with or in the form of Natural Language Processing to develop fully functional Dialogue Systems – which will be elaborated in the next paper of the series – to make the virtual world more user friendly. This paper discusses some techniques that can be used or implemented to realize such systems. The paper also considers the techniques required in the Dialogue Based Natural Language Processing System and describes each technique briefly. The objective of this paper is to efficiently bring around the concepts of Dialogue Based Systems to the general public using pre-defined libraries developed by some institutions and organizations (e.g. ALICE AIML, CMU Sphinx etc.) for the tasks such as Natural Language Processing and Speech Recognition.

Index Terms— Artificial intelligence, AIML, CMU Sphinx, Dialogue based system, Natural language processing,Natural language processing, Speech synthesis

1 Introduction

Imagine you reach home after a tiring day in office and tell your "computer" that you would like to hear some soft music. The computer begins playing tracks off your music library. When are refreshed you tell your computer to stop playing music and open up the Internet for you and it does so.

This may seem like a scene from a science fiction film but can actually be implemented very effectively by the use of some techniques of Artificial Intelligence, Natural Language Processing and Speech Synthesis. Let us start with the basics of each technique to be implemented in the fully functional Dialogue System ranging from Artificial Intelligence to Speech Synthesis.

2 What is Artificial Intelligence

Artificial Intelligence (AI) is a branch of engineering and science devoted to constructing machines that can think. AI can be defined as "the design and study of intelligent agents"; an intelligent agent is a system (or a computer) that observes its environment and performs actions that increase the chances of its success. John McCarthy coined the term Artificial Intelligence in 1955, calling it "the science and engineering of making intelligent machines".

The main objectives of AI include reasoning, knowledge, planning, learning, communication, perception and the ability to move and manipulate objects. Popular approaches include statistical methods, computational intelligence and traditional symbolic AI. There are a large number of tools used in AI, including search and mathematical optimization, logic, methods based on probability and economics, and others.

• *Shakti Phartiyal is a Bachelor of Technology Graduate from Uttarakhand Technical University, India. E-mail: shaktiphartiyal@gmail.com*

The central approach of AI is to simulate human logic and intelligence in an electronic system. Today AI is an essential part of the technology industry and many of the most complex problems in computer science sector. There is no one theory or rule that guides AI research. Researchers disagree about certain issues. Some of the longest standing questions that remain unanswered are: Should Artificial Intelligence simulate natural intelligence by studying psychology or neurology? Or is human biology as irrelevant to AI research as bird biology is to aeronautical engineering? Can intelligent behavior be described using simple logic or optimization? Or does it essentially require solving a variety of completely unrelated problems? Can intelligence be reproduced using high-level symbols, similar to words and ideas?

A lot of tasks can be achieved through the implementation of AI. These tasks include Planning, Learning, Motion and Manipulation, Natural Language Processing, Deduction, Reasoning and Problem Solving.

This paper will, however, focus on implementing Natural Language Processing by the efficient and effective use of AI for the construction of a Dialog Based System (DBS).

3 What is Natural Language Processing

Natural Language Processing (NLP) is the computerized method to analyze text that is based on a set of theories and technologies. Quite simply NLP can be defined as – Natural Language Processing is a method of human - computer interaction, it enables computers to extract meaning from the words and phrases that people use in their daily lives – and thus respond in the same manner (human understandable) when presenting information back to them.

Natural language processing gives machines the ability to read and understand the languages that humans speak. A

powerful natural language processing system would enable the acquisition of knowledge directly from human-written sources, such as Internet.

I define NLP as the branch of computer science focused on developing systems that allow computers to communicate with people using everyday language.

Natural language processing systems convert human language into formal semantic representations which computer applications can interpret, and respond with easily understood grammatical sentences. Thus, enabling humans to interact more efficiently and naturally with computer using normal, familiar and day to day expressions – instead of using pre-constructed computer languages.

Natural Language Processing requires two important steps:

*. *Natural Language Understanding:*

The task of NLU is to understand and reason out while the input is a natural language.

*. *Natural Language Generation:*

The task of NLG is to generate a Natural Language which a user can understand.

Figure 1 represents the working of a typical NLP system.

Fig. 1. Working of a NLP System

3.1 Goals of Natural Language Processing

The goals of NLP are "to accomplish human-like language processing".

The choice of word 'processing' is on purpose, and should not be confused with 'understanding'. Although, the field of NLP was referred to as Natural Language Understanding (NLU) in the early days of AI, it is well agreed today that while the goal of NLP is true NLU, it is yet unaccomplished.

A full NLU System would be able to:

a. Interpret an input text
b. Convert the text into another language
c. Answer questions about the contents of the text
d. Draw conclusions from the text

The goal of the NLP system here is to correspond to the true meaning and intent of the user's query, which can be expressed naturally in our everyday language. The goal in the production and comprehension of natural language is communication between the user and the computer system.

3.2 Steps involved in Natural Language Processing

Natural Language Processing is done at 5 levels as shown in the figure 2 below:

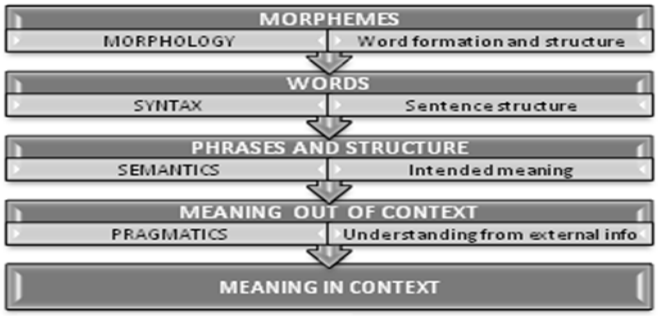

Fig. 2. Steps of Natural Language Processing.

3.3 Natural Language Text Processing Systems

Manipulation of text for information extraction, for auto indexing and summary, or for producing text in a desired format, has been recognized as an important area of application and research in Natural Language Processing. This is classified as the area of natural language text processing that allows organization of large bodies of text-based information with a view of retrieving particular information or of deriving knowledge structures that can be used for a specific purpose. Automatic text processing systems take some form of text input and transform it into an output of another form. The central task for natural language text processing systems is the translation of ambiguous natural language queries and texts into unambiguous internal representations on which matching and retrieval can take place. A natural language text processing system may begin with morphological analysis. Stemming of terms, in both the queries and documents, is done in order to get the morphological variants of the words involved. The lexical and syntactic processing involves the use of lexicons for shaping the characteristics of the words, recognition of their parts-of-speech, determining the words and phrases, and for parsing of the sentences.

3.4 Natural Language Processing Applications

There is a vast sector in the applications of Natural language Processing. The most frequent applications utilizing NLP include the following:

• Information Retrieval – given the significant presence of text in this application, it is surprising that so few implementations utilize NLP. Recently, statistical approaches for accomplishing NLP have seen more utilization, but few systems other than those by Liddy[1] and Strzalkowski[2] have developed significant systems based on NLP.

• Information Extraction (IE) – a recently emerging application area, it focuses on the recognition, tagging, and extraction

into a structured representation, certain key elements of information, e.g. persons, companies, locations, organizations, from large collections of text. These extractions can then be utilized for a range of applications including question-answering, visualization, and data mining.

• Question-Answering – in contrast to Information Retrieval, which provides a list of potentially relevant documents in response to a user's query, question-answering provides the user with either just the text of the answer itself or answer-providing passages.

• Summarization – the higher levels of NLP, particularly the discourse level, can empower an implementation that reduces a larger text into a shorter, yet richly-constituted abbreviated narrative representation of the original document.

• Machine Translation – perhaps the oldest of all NLP applications, various levels of NLP have been utilized in MT systems, ranging from the 'word-based' approach to applications that include higher levels of analysis.

• Dialogue Systems – perhaps the omnipresent application of the future, in the systems envisioned by large providers of end-user applications. Dialogue systems, which usually focus on a narrowly defined application (e.g. The dialogue exchange between your refrigerator or home sound system), currently utilize the phonetic and lexical levels of language.

4 Speech Recognition

Speech recognition is the computerized translation of spoken words into text. It is also known as "speech to text"[3].

According to the Wiki. of CMU Sphinx[4] "Speech is a complex phenomenon. People rarely understand how is it produced and alleged. The naive perception is often that speech is built with words, and each word consists of phones. The reality is unfortunately very different. Speech is a dynamic process without clearly distinguished parts. "

4.1 Structure of Speech

Speech is a continuous stream of audio where somewhat stable states mix with dynamically changed states. In this sequence of states, one can define more or less similar classes of sounds, or phones. It is often understood that words are built of phones, but this is not true. The acoustic properties of a waveform corresponding to a phone can greatly differ depending on factors such as - phone context, speaker, style of speech and so on.

Phones build sub-word units, like syllables, sub-words form words. Words are important in speech recognition because they restrict combinations of phones considerably. If there are 60 phones and an average word has 7 phones, there must be 60^7 words. Fortunately, even a very educated person rarely uses more than 20k words in his general speaking, which makes recognition much easy or feasible.

4.2 Speech Recognition process

The speech recognition process is performed by a software component known as the speech recognition engine [5]. The main function of the speech recognition engine is to process spoken input and translate it into text that an application understands.

Silence is a very important concept in speech recognition as it identifies the start and end of an utterance – an utterance is what a user speaks – when the speech recognition engine detects audio input i.e. a lack of silence – the beginning of an utterance is signaled. Similarly, when the engine detects a certain amount of silence following the audio, the end of the utterance occurs. If the user doesn't say anything, the speech recognition engine returns a silence timeout which is an indication that there was no speech.

A speech recognition engine requires a grammar with which it converts the spoken frequencies into valid phones thereby combining them to form words and sentences. A speech recognition engine also requires a phonetic dictionary which contains a mapping from words to phones. Although this mapping is not very effective so one can also use some complex function learned with a machine learning algorithm.

Accuracy of speech recognition engine is a major issue.

Some commonly used speech recognition engines are:

- Microsoft Speech Recognizer 8.0 for Windows (SPAI5).
- Sphinx4.[6]
- Pocket Sphinx.

5 Speech Synthesis

Speech synthesis is the artificial production of human like speech. A program/system used for this purpose is known as a speech synthesizer, and can be implemented in software or hardware products. A text-to-speech (TTS) system converts normal language text into speech. [11]

A text-to-speech system comprises of two major components viz. a front-end and a back-end. The front-end has two major tasks. First, it converts raw text containing symbols like numbers and abbreviations into the equivalent of written-out words. This process is often called text normalization, pre-processing, or tokenization. The front-end then assigns phonetic transcriptions to each word, and divides and marks the text into prosodic units, like phrases, clauses, and sentences. The process of assigning phonetic transcriptions to words is called text-to-phoneme or grapheme-to-phoneme conversion. Phonetic transcriptions and prosody information together make up the symbolic linguistic representation that is output by the front-end. The back-end—often referred to as the synthesizer—then converts the symbolic linguistic representation into sound. In certain systems, this part includes the computation of the target prosody (pitch contour, phoneme durations), which is then im-

posed on the output speech.

The overview of a TTS System is as shown in figure 3:

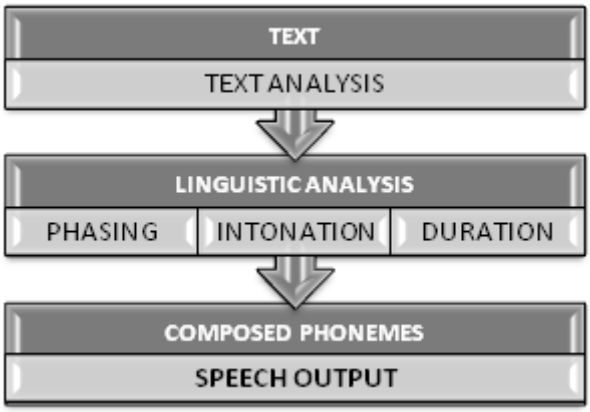

Fig.3. Processing of a TTS System

Some of the free and/or open source speech synthesizers are:

- eSpeak[10]
- Gnuspeech
- Festival Speech Synthesis System
- FreeTTS

6 Dialogue Based Systems

A Dialogue Based System (hereafter referred to as DBS) is a combination of computer software and hardware that is used to talk or exchange dialogues between the computer and the user these dialogues are understandable by both the machine and the human. A Chabot is an example of a DBS some examples include CharlieBot, AliceBot and so on.

Systems in which human users speak to a computer in order to achieve a goal are called spoken dialogue systems. Such systems are some of the few realized examples of open ended and real-time interaction between humans and computers, and are therefore an important and exciting sector for AI and machine learning research. Spoken dialogue systems often integrate components such as a speech recognizer, a database backend (so as to extract information as required by the user), and a dialogue strategy (in the case discussed in this paper we will use AIML as a dialogue strategy).

The main question that arises is "What is the need of a DBS"? The answer to this question is the ease of use i.e. accomplishing our tasks without having to click or type through a series of programs.

As stated before, imagine a scenario when you just need to ask a computer to play a specific song for you - the computer either plays the song or if the song is not found it tells you that the said song was not found. This could be a much better interactive approach for using our computer systems.

One example of one complex dialogue based system is MATCH (Multimodal Access To City Help) [7].

7 Approach to develop a full scale DBS

Here I present you with the tools needed to develop a fully functional Dialogue Based System :

- Apache OpenNLP[8]
- ALICE AIML[9]
- CMU Sphinx4[6]
- eSpeak[10]

A brief description of the above mentioned tools:

7.1 Apache OpenNLP:

Apache OpenNLP is an open source collection of tools and APIs for processing Natural Language.

The Apache OpenNLP library is a machine learning based toolkit for the processing of natural language text. It supports the most common NLP tasks, such as tokenization, sentence segmentation, part-of-speech tagging, named entity extraction, chunking, parsing, and co-reference resolution. These tasks are usually required to build more advanced text processing services. OpenNLP also included maximum entropy and perceptron based machine learning.

The libraries and APIs of Apache OpenNLP can be very easily integrated together with a programming language such as Java to accomplish the NLP task of a DBS.

The libraries present in Apache OpenNlp are as follows:

- Sentence Detector
- Tokenizer
- Name Finder
- Part of Speech Tagger
- Chunker
- Parser

7.2 ALICE AIML:

Artificial Intelligence Mark-up Language i.e. AIML was developed by the Alicebot free software community and Dr. Richard S. Wallace during 1995-2000. It was originally adapted from a non-XML grammar also called AIML, and formed the basis for the first Alicebot (a chat bot used in chat based systems), A. L. I. C. E., the Artificial Linguistic Internet Computer Entity.

AIML describes a class of data objects called AIML objects and partially describes the behavior of computer programs that process them. AIML objects are made up of units called topics and categories, which contain either, parsed or unparsed data. Parsed data is made up of characters, some of which form character data, and some of which form AIML elements. AIML elements encapsulate the stimulus-response knowledge contained in the document. Character data within these elements is sometimes parsed by an AIML interpreter, and sometimes left unparsed for later processing by a Responder.

The elements of AIML are:

- CATEGORIES <category>:

The category consists of a question that a user might ask and it

also contains the possible answer/answers to that specific question.

- RECURSION <srai>:

The <srai> tag element is used for symbolic reductions.
This can be explained with the help of an example:
Symbolic reduction refers to the process of simplifying complex grammatical forms into simpler ones. Usually, the atomic patterns in categories storing chat-bot knowledge are stated in the simplest possible terms, for example we tend to prefer patterns like "WHO IS SOCRATES" to ones like "DO YOU KNOW WHO SOCRATES IS" when storing biographical information about Socrates.
Many of the more complex forms reduce to simpler forms using AIML categories designed for symbolic reduction:

```
<category>
<pattern>DO YOU KNOW WHO * IS</pattern>
<template><srai>WHO IS <star/></srai></template>
</category>
```

Whatever input matched this pattern, the portion bound to the wildcard * may be inserted into the reply with the markup <star/>. This category reduces any input of the form "Do you know who X is?" to "Who is X?"
OR the <srai> tag can be used to check for possible synonyms such as:

```
<category>
<pattern>HI THERE</pattern>
<template><srai>HELLO</srai></template>
</category>
```

- PATTERN <pattern>:

This tag holds the question, this comes under <category> tag.

- TEMPLATE <template>:

This tag holds the answer to the question in the <pattern> tag.

So a simple question answer pattern in AIML would be:

```
<category>
<pattern>WHO CREATED AIML ?</pattern>
<template>Dr. Richard S. Wallace created AIML.</template>
</category>
```

7.3 CMU Sphinx4:

Sphinx4 is an Open Source Speech Recognition Engine developed by a joint collaboration between the Sphinx group at Carnegie Mellon University, Sun Microsystems Laboratories, Mitsubishi Electric Research Labs (MERL), and Hewlett Packard (HP), with contributions from the University of California at Santa Cruz (UCSC) and the Massachusetts Institute of Technology (MIT).

Sphinx4 is an excellent open source API for developing Speech Recognition Systems with Java.

The working of Sphinx4 can be depicted with the help of figure 4:

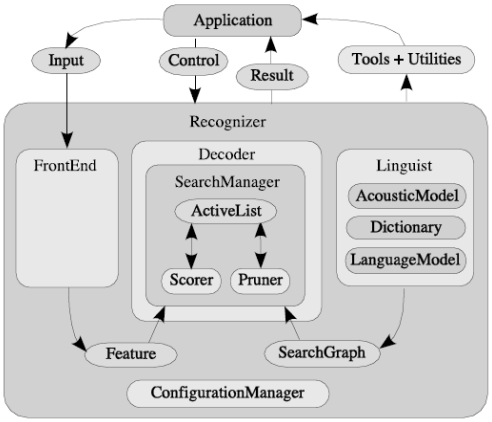

Fig.4. Working of a Sphinx4

7.4 eSpeak:

eSpeak is a compact open source software speech synthesizer for English and other languages, for Linux and Windows Operating Systems.

eSpeak uses a "formant synthesis" method. This allows many languages to be provided in a small size. The speech is clear, and can be used at high speeds, but is not as natural or smooth as larger synthesizers which are based on human speech recordings.

8 The Proposed Simple DBS

Applying all the tools given above in a sequence to perform their specific tasks a fully functional and simple Dialogue Based System can be developed as shown in the flowchart below:

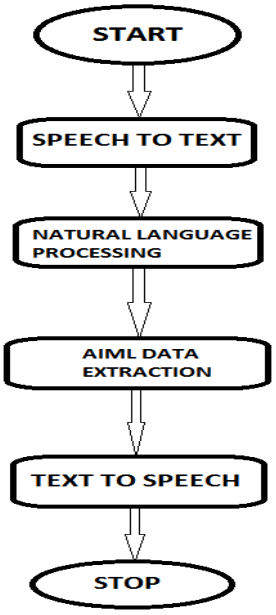

Fig.5. Working of Proposed DBS

The flowchart in Figure 5 shows the basic working of the pro-

posed simple dialogue based system.

THE WORKING OF THIS MODEL CAN BE EXPLAINED AS THE FOLLOWING 6 STEPS:

8.1 START:

This is the first phase of the DBS in which the user speech is taken as input by our model DBS system.
This phase can be achieved by a language such as Java using the method *recognize()* which can be accessed from the CMU Sphinx libraries. The best recognized and filtered (noise removed) speech is then passed on to the core Sphinx engine for speech to text conversion.

8.2 SPEECH TO TEXT:

In this phase the filtered speech is converted to equivalent text using the open source speech recognition and conversion engine provided by CMU Sphinx.
According to CMU Sphinx the recognition process is stated as - "We take waveform, split it on utterances by silences then try to recognize what's being said in each utterance. To do that we want to take all possible combinations of words and try to match them with the audio. We choose the best matching combination. " [4].
Three basic models are used for speech to text conversion:

- Acoustic Model – Contains acoustic properties for each sound.
- Phonetic Model
- Language Model

The converted text is now a natural language text which is then sent to the Natural Language Processing Library which is the ALICE AIML Interpreter.

8.3 NATURAL LANGUAGE PROCESSING:

This phase as the name suggests deals with the processing of Natural Language, this is achieved by the ALICE AIML Interpreter which checks and recursively matches each string broken into smaller tokens to the patterns present in the AIML file.
The matching is somewhat similar to the SQL query matching but different in the way of the recursive match.
The matching process selects a specific pattern from the AIML file according to best match which is much more enhanced form of the regular expression match.

8.4 AIML DATA EXTRACTION:

This step is actually a subpart of natural language processing but is being discussed separately for ease of understanding.

AIML data extraction deals with the matched pattern to select its specific template from the AIML file after all the recursive calls and further template processing is done (template processing is beyond the scope of this paper).

Finally the processed pattern is passed on to the next stage.

8.5 TEXT TO SPEECH:

The processed pattern is the input for this phase. The pattern is passed to a text to speech (TTS) engine such as FreeTTS or eSpeak using their specific API and according to its grammar the specific speech is produced and is given out to the user using the programming API.

8.6 STOP:

After the completion of the previous steps, this step, either halts or send control back to the first step according to the program needs.

9 CONCLUSION

This proposed DBS is one of the simplest types of DBS with full featured Artificial Intelligence implemented that can be made using the open source AI libraries and APIs.
Its lightweight architecture will enable individuals / groups to integrate it along with other applications that can prove to be very helpful in the further development of futuristic technologies.
This DBS can be used even by technical novices to search / query for relevant data with other integrations to it. With features such as multi language support this application can also be used by people in their native languages without the need of opening up browsers and clicking up buttons and icons which can be a tedious task for some.

REFERENCES

[1] Gobinda G. Chowdhury "Natural Language Processing" Dept. of Computer and Information Sciences University of Strathclyde, Glasgow G1 1XH, UK
[2] Reports of progress of the NLIR (Natural Language Information Retrieval) project are available in the TREC reports (Perez-Carballo & Strzalkowski, 2000; Strzalkowski. et al.., 1997, 1998, 1999).
[3] Speech Recognition [*http://en.wikipedia.org/wiki/Speech_recognition*]
[4] CMU Sphinx [*http://cmusphinx.sourceforge.net/wiki/*]
[5] Kimberlee A. Kemble "An Introduction to Speech Recognition" Program Manager, Voice Systems Middleware Education, IBM Corporation
[6] Willie Walker, Paul Lamere, Philip Kwok, Bhiksha Raj, Rita Singh, Evandro Gouvea, Peter Wolf, Joe Woelfel "Sphinx-4: A Flexible Open Source Framework For Speech Recognition"
[7] Michael Johnston, Srinivas Bangalore, Gunaranjan Vasireddy, Amanda StentPatrick Ehlen, Marilyn Walker, Steve Whittaker, Preetam Maloor "MATCH : An Architecture for Multimodal Dialogue Systems" AT&T Labs - Research, 180 Park Ave, Florham Park, NJ 07932, USAjohnston,srini,guna,ehlen,walker,stevew,pmaloor@research.att.com, stent@cs.sunysb.edu
[8] Apache OpenNLP Developer Documentation, The Apache Software Foundation.
[9] Dr. Richard S. Wallace "The Elements of AIML Style" 2003 ALICE A.I. Foundation, Inc.
[10] eSpeak [*http://espeak.sf.net*]
[11] Allen, Jonathan; Hunnicutt, M. Sharon; Klatt, Dennis (1987). "From Text to Speech: The MITalk system". Cambridge University Press. ISBN 0-521-30641-8.