International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

PreLogue: A Smart Voice Controlled Presentation

Amrutha D[1], Ruthvik Vijayakumar[2], Deepika R[3], Chandini C[4]

[1, 2, 3, 4]PESIT Bangalore South Campus, Department of Information Science and Engineering,
Hosur Road, Electronic City, Bangalore – 560100, Karnataka, India

Abstract: ***For a Presentation to be effective, the presenter must completely engage the audience while giving a discourse with the assistance of a visual aid. Conventionally, a mouse, keyboard or expensive presentation tools are used for navigating the slides. Our aim is to minimize the effort and the cost for the presenter and let him focus completely on engaging the audience. The proposed application will scan through the document and starts gathering intents and keywords of specific sentences. While presenting, the presenter's voice will be processed using NLP, NLU, Keyword Extraction and Machine Learning Algorithms and based on the intent of the speech the appropriate slide is put up. In addition to navigating slides, a natural language parser is built, which processes complex queries for real-time data analysis and visualization for added effectiveness of the presentation. The proposed application aims to provide a hands-free, smart, voice controllable presentation system which is intuitive, simple and efficient.***

Keywords: Presentation, Slide Navigation, Voice Recognition, NLP, Machine Learning, Keyword Extraction, NLU

1. Introduction

Many people make use of presentation software to promote their work or to support their discussion. But what if the same software tends to pose problem rather than providing intended support. Although presentation software can offer visual support for your discussion, they also have certain disadvantages, which may lower the impact of your presentation.

While delivering a presentation, as we are aware, the presenter is forced to hover over the screen to control the slides, this hinders the effect of presentation, as the presenter may fail to use the space effectively around him and as a result would not be able to connect to the audience at his best. Thus, our approach provides an effective solution, as the presentation slides can be controlled just by issuing voice commands.

Basic Idea / Our Aim: Firstly, the Input Presentation is converted into web pages so that the presenter has the freedom to continue using the application whatever the platform is. Once this is done, he has to trigger microphone input and his speech is continuously processed. Now he has control over the slides, and can change to next or the previous slide or to any other slide for that matter, by voice. And also by extracting keywords from the speech, and based on its confidence, the slide corresponding to that particular keyword is put up.

1.1 Why This Approach?

As the technology is growing drastically day by day, things are getting simplified to make life easier and comfortable. One of our main objective is to make the presenter feel free and comfortable while delivering the presentation, rather than going through any inconvenience as in case of existing system. Thus, the proposed solution aims to solve this. Who would have imagined that one day machines would be able to turn our talks into written text that makes tasks much simplified and convenient.

1.2 Drawbacks of the Existing System

1. Delay -The presenter encounters a delay in his presentation when in case he wants to refer to the slide which is not just previous or after the current slide, there's a delay when searching for the slide by scanning the whole presentation top to bottom.
2. Cost - Buying expensive presentation control tools such as wireless presenters, which cost a fortune which is also platform dependent.
3. Disconnect Between Speaker and Audience - Rather than facing your audience, you might face the projection screen to constantly refer to the presentation and risk reading from the screen rather than truly engaging with them.

2. Implementation

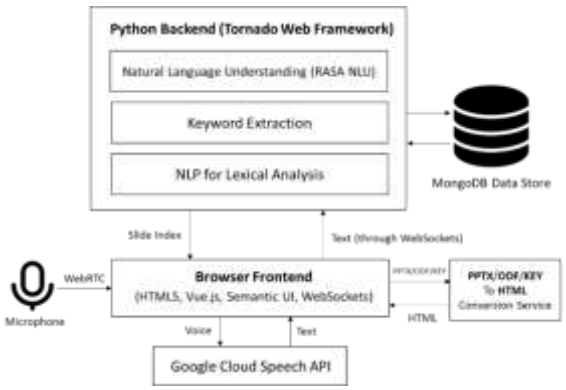


Figure 1: Block diagram of the entire Application

The presenter is shown a web page in a browser which forms the front end which is implemented using HTML with additional UI frameworks to beautify it. To start off with the presentation, the presenter is prompted to upload a presentation in any of the following formats (.key, .pptx, .ppt, .odf) as long as it is in accordance with Open XML format. The file uploaded is now;

1. Pre-Processed - for gathering intents and significant keywords and is mapped to the respective slide number, and then stored in a MongoDB data store in the back end, for easy access to extracted data.

2. Converted to PDF - Since various presentation programs have proprietary features, for example: Smart Art in PowerPoint, the PDF format ensures that this feature is embedded without any changes in its position, moreover font compatibility is also a big advantage since fonts used are embedded into the PDF file This PDF is later converted to HTML using pdf2htmlEX[1] which uses the Open Standard PDF specification for the conversion, this conversion allows us to have a better DOM manipulation which further enhances the experience of the presentation, this also ensures one of our main aim "Platform Independence", i.e. One does not need to have a specific platform to show his presentation.

Once the presenter starts presenting, we access the Chrome Web Speech API [2] which will take in the voice from the microphone and convert it to text, which is then sent through the JavaScript WebSocket client, implemented in every modern browser to the WebSocket server, running as localhost. WebSockets provide full duplex data transfer with minimum latency, unlike the REST API.

The raw text sent through the client is then handled by a WebSocket server i.e. Tornado (Python Web Framework) where further processing to the spoken text is done. The input text goes through three essential algorithms i.e. Keyword extraction, Natural Language Understanding (NLU) and Semantic Similarity, which is defined in detail later in the paper. The raw text first goes through NLU algorithm, if the intents and the entities have confidence greater than 95%, the following analysis is done, consider the following two cases,

Case I --
raw input = "in the next slide"
Where,
slide.action = 'next'
slide.target = 'slide'
and corresponding confidence is,
next = 100%
slide = 100%

Case II –
raw input ="the next generation of processors"
Where,
slide.action = 'next'
slide.target=?
and the corresponding confidence is,
next =100%

Since one of the properties is < 95% or missing this algorithm is not considered instead we switch over to keyword extraction, which will return significant keywords from the raw text and checks with the extracted keywords in the pre-processing stage. Now the semantic similarity, using the algorithm which is explained later in the paper of the extracted keyword from the slide and the raw text is computed and if they have a confidence of 75%, the slide number mapped to the keyword is sent back as a response to the front end through WebSockets, as a result the DOM is manipulated to show the intended slide.

One should note that the speech is recorded and processed at all times during the course of the presentation, and actions are triggered only when the text complies with all the above algorithms.

Figure 2: The UI of the proposed system

3. Algorithm Specification

3.1 Natural Language Understanding

NLU forms the crux of the application, using which we analyze text and extract 'meta-data' from the unstructured content like concepts, entities, keywords, categories and sentiments etc. The unstructured data is converted into a structured form that a machine can understand, and the appropriate action is taken. This analysis can be customized for specific needs, by specifying a domain, when a domain is defined; it improves the accuracy of the meta-data extraction.

To do this we must distill the spoken text into a structured ontology using a combination of rules and statistical modeling. Entities must be extracted, identified and semantic meaning must be derived within the context. For example,

"I need a flight and a hotel in Bangalore from May 1 to 5"
is parsed to --
need: flight{intent} / need: hotel{intent} / Bangalore{city} / May 1{date} / May 5{date}

RASA NLU [3] an open source toolkit which is a set of high level API's for building a language parser, which provides the necessary functionality mentioned above, it essentially has two parts –

1. Intent Classification – An Intent is what the user intends to do, such as goToSlide, gotoNextSlide, getCurrentExchange actions.
2. Entity Extraction – Entities are variables that contain details of the user's task, for intent goToSlide, the entity is the 'slide number', using which the intent can be operated.

[4] We need to convert a block of text into tokens into structured content, which requires more than just splitting on white space, one approach would be using a combination of NLP and ML libraries, SpaCy/testacy and scikit–learn respectively. The other approach would be use of MITIE (MIT Information Extraction), which has both the ML and NLP parts built in.

We go with the former approach since as it allows us to train custom models, pertaining to our data set, and also gives us the control over entity extraction. Using which we

1. Categorize the Text.
2. Recognize the Entities.

We define the intents, and assign these intents to the training data, we then tag the possible entities that is to be extracted., in the following format -

```
{
"text":"let's move on to the next slide ",
"intent": "slide. Action",
"entities": [
 {
"start": 21,
"end": 24,
"value": "next",
"entity": "slide. Direction"
 },
 {
"start": 25,
"end": 30,
"value": "slide",
"entity": "slide.target"
 }
 ]
}
```

We now train the model; here the training data is to be defined by us initially, further as a feedback mechanism we supply the predicted data back to the training set to further train the data set making it more accurate with each iteration.

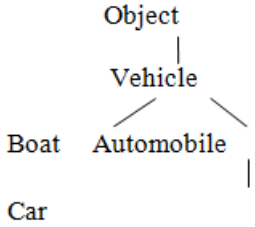
The MITIE library is quite sophisticated which uses spectral word embedding rather that GloVe i.e. custom word embedding, and the text categorizer is a straight forward SVM (Support Vector Machine), a supervised learning model used for classification and regression analysis, and the entity extraction uses structural SVM which promises better accuracy.

In future with a combination of the above two models a hybrid model can be implemented, which will result in the flexibility of (SpaCy/textacy and scikit-learn) and the accuracy of MITIE.

3.2 Semantic Similarity of Sentences

The NLTK (Natural Language Toolkit) Python Library provides an interface to work with human language data, the word net interface and the corpus statistics from Brown corpus contains several samples of English language text which are a part of NLTK's API, are used.

[5]We initially started with word similarity which is derived from combination of two functions f(l) and f(h), where l is the shortest path between two words in the word net (the semantic network) containing a lexical database of nouns, verbs, adjectives and adverbs, grouped into sets of cognitive synonyms, which are linked by lexical relations, and h is the height of the LCS (lowest common subsumer), which is a concept tree such that two concepts A and B have the same ancestry which has the most specific concept.



Here, the LCS of Boat and Automobile is Vehicle, which has the least height to common subsume. and this is calculated from the root of the semantic network. So, value of 'l' gives us as to how similar the words are and 'd' gives the specificity of the LCS, the closer the LCS nodes, the broader abstract is the concept.

We then normalize the values i.e. f(l) and f(h) to the range [0,1]

Similarity

$$(w_1, w_2) = f(l).f(h)$$

$$\text{Where, } f(l) = e^{-\alpha l}$$

$$f(h) = \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} \quad (1)$$

Sentence Similarity is computed as combination of semantic similarity and word order similarity, which is given by the cosine similarity between the semantic vectors of two sentences, to build this vector, the union of words which is tokenized is taken and treated as vocabulary, if a word occurs in a sentence, then its value is 1 else the similarity is computed against all the other words in the sentence, if this occurs beyond ψ then the value of the element is ψ, or else it is 0, this value is further reduced by content found in the corpus.

$$S_s = \frac{s_1 . s_2}{|s_1| . |s_2|} \quad (2)$$

Where,

$$s_i = s.I(w_i).I(w_j)$$

$$I(w) = 1 - \frac{\log(n+1)}{\log(N+1)}$$

Where,

n = number of times the word occurs in the corpus
N = number of words in the corpus

Now the word order rectifies the fact that some words can have radically different meanings when used in a sentence, This is done by computing the word vector and finding normalized similarity, word order vector like the semantic vector is based on the joint word set. If the word occurs in a sentence, its position is recorded. If not, the similarity to the most similar word in the sentence is recorded if it crosses a threshold η else it is 0.

$$S_r = 1 - \frac{|r_1 - r_2|}{|r_1 + r_2|} \quad (3)$$

Where,

r_1 = word position vector for sentence 1
r_2 = word position vector for sentence 2

Hence the similarity between two sentences is modeled as a linear combination of semantic and word order similarity, i.e.:

$$S = \delta S_s + (1 - \delta) S_r \quad (4)$$

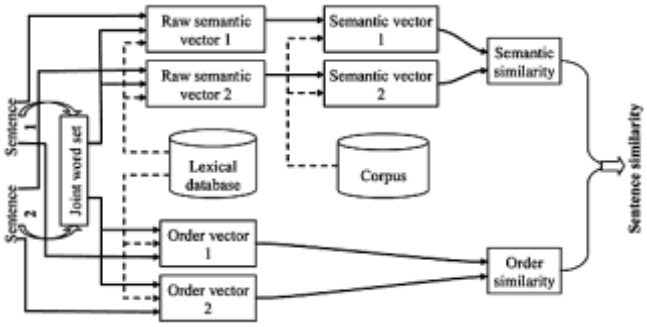


Figure 3: [6] Block diagram of Semantic Similarity Analysis

3.3 Keyword Extraction

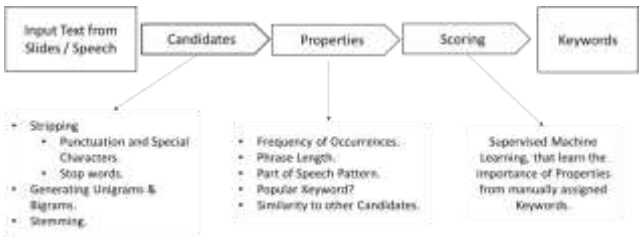


Figure 4: Stages of Keyword Extraction

Since we are not working with a large dataset, but rather short utterances and short sentences, we adopt a different approach for keyword extraction. The former requires multi-level word clustering and term frequency evaluation, while the latter needs a term weighing approach.

[7]TF-IDF is the simplest approach for this, where the keyword with highest occurring frequency is chosen as keywords, hence we have selected MAUI as the classification model, which extracts a list of keyword candidates and trains them over a large decision tree. Once trained it returns a ranked list of keywords.

The extraction process essentially has three steps:

1) Pre-processing-

- Striping punctuation, special characters
- Strip 'simple words' i.e. stop words.
- Generate uni-grams and bi-grams to be considered as keyword candidates.
- Stemming i.e. reducing the word to their root word in the word stem.

For example: going > go, cars > car

2) Property Calculation - For each candidate, we need to calculate the property by which it can be classified as a keyword.

3) Scoring and Selecting -

- The candidates can be scored by combining their properties into a formula
- Using Machine Learning techniques to determine the probability of a candidate being a keyword, a score or probability threshold is said to refine the quality of the keywords.

This model suited our application, since the data set is short and needs better accuracy, since it enabled us to extract not only from the text, but also with references to a controlled vocabulary. Here the vocabulary is the Brown Corpus and the slide data itself.

4. Advantages

1. Access– It is extensively useful for physically disabled people that prevent them from using a keyboard or a mouse, being able to issue voice commands and dictate words into a text document is a significant advantage of this approach.
2. Freedom – When a particular slide is addressed by a specific keyword present in that slide, the application also considers the synonyms of those keywords and gives out the expected result. Thus the presenter has the freedom to refer to closely related keywords.
3. Generality- Anybody and everybody can upload their presentation on to the system, and start presenting. No time needed for setup nor previous experience is required.
4. Cost efficient – To implement this application, as no hardware components are required, the cost is kept minimum.
5. Platform independence - There are now more than eight major platforms for applications, each of which presents its own programming paradigms. Providing a high-quality experience to as many users as possible requires platform-independent application designs. In this particular application, since the presentation format is converted to HTML, the application becomes platform independent.

5. Future Prospects

The proposed application can be extended to add few utilities that can further enhance the presentation; one such utility is as stated:

Processing the real-time queries independent of the processed data, where the query is processed by NLU techniques, and results are acquired from the given data or the data gathered from the internet i.e. data scraped from the web, data repositories and other API's. We present two use cases for the above utility –

Use Case I –

Suppose sales pitch needs the gross profit of the fourth quarter of a company, given the data the speech is processed to extract meta-data such as gross profit, loss, turnover etc. and is immediately put up on the slide, either as a table or a chart for effective visualization of the data.

Use Case II –

A Math professor is giving a lecture on Integration Techniques, we process his speech, tokenize and convert it to an equation and evaluate the equation using the WolframAlpha API.

This is not only limited to the above use cases but can extend to various others, as long as the data is readily available, in the form of API's, data repositories, or data scraped from websites.

This eliminates the load to acquire data by launching a separate task i.e. Browser >Search Engine> Data.

6. Limitations

These are some of the limitations identified, which can very well be tackled upon further research and implementing the results.

1. Stable network connectivity, at all times is required for efficient and accurate processing of the input speech; this can be solved by implementing an offline speech recognizer with limited capabilities.
2. The application demands quite some system resources, as various modules have to work in conjunction with each other, constantly communicating data; this is avoided as long as the target machine is capable.
3. When the pronunciation is not clear or if some of the sentences are overlapped during the speech, complexity in the application may arise.

7. Related Work

1. Automatic Slide Navigator By Voice Commands Using Microcontroller- B. Arun Kumar, CH Sridevi, P. Jeevan, V. Anuragh.
2. The Design of Speech-Control power point presentation tool using ARM7 - P. Vishnu Kumar, Dr.T. Jaya Chandra Prasad.

8. Conclusion

Thus, we are now able to navigate to any slide in the presentation in a smarter way by just giving voice commands irrespective of any hardware components by adopting NLP, NLU and Machine Learning techniques. The cost and risk is kept minimum, achieving efficiency and consistency.

References

[1] PDF to HTML Conversion – http://coolwanglu.github.com/pdf2htmlEX/
[2] Chrome Web Speech API Documentation – https://www.google.com/intl/en/chrome/demos/speech.html
[3] RASA NLU Documentation - https://rasa-nlu.readthedocs.io/en/stable/
[4] RASA NLU Implementation – https://conversations.golastmile.com/do-it-yourself-nlp-for-bot-developers-2e2da2817f3d
[5] Semantic Similarity for Short Sentences – http://sujitpal.blogspot.in/2014/12/semantic-similarity-for-short-sentences.html
[6] Sentence Similarity Based on Semantic Nets and Corpus Statistics – DOI: 10.1109/TKDE.2006.130 · Source: IEEE Xplore
[7] NLP Keyword Extraction Tutorial – https://www.airpair.com/nlp/keyword-extraction-tutorial