Speech Calculator

Mohith R.
Computer Science and Engineering
PES University
Bengaluru, India
mohithraj9301@gmail.com

Sanjiv Sridhar
Computer Science and Engineering
PES University
Bengaluru, India
sanjivsridhar01@gmail.com

Chinmay Kulkarni
Computer Science and Engineering
PES University
Bengaluru, India
camkoolkarni@gmail.com

Shylaja S. S.
Computer Science and Engineering
PES University
Bengaluru, India
shylaja.sharath@pes.edu

***Abstract*—Mankind has always looked for faster and more efficient ways of doing things that usually take a tedious amount of human effort and a lot of time to do. Controlling and automating through voice has been a hot topic in research, and a lot of innovation has come up. This project utilizes the power of Natural Language Processing and Speech analysis to ease the way of tedious mathematical calculations. Mathematical calculations are one of such things which are extremely important but tiring. In this paper, we present our method of calculating faster and efficiently in the form of an app with only voice input. The app would return the results of the mathematical query.**

Keywords—natural language processing, speech to text, hands-free

I. Introduction

Automation and hands-free experience are in huge demand in today's era. The equipment with voice control such as Alexa, Siri, and the Google assistant is getting extremely popular. Much world-class technology goes into these, one of the most important being Natural Language Processing, Text Processing, and Speech-to-Text technologies. Keeping these demands of faster, hands-free, and efficient computation in mind, this paper presents a novel approach to solving mathematical queries using only voice.

We demonstrate the working in the form of a flutter app and with a python backend, with flask as the middleware. The app focuses on solving the mathematical equations with multiple operands and operators with precedence and associativity are taken care of. Operators supported are +, -, *, /, trigonometric, log with any base, power, square root, cube root.

II. Literature Survey

In this paper[2] by Ansul Gupta et al., a new method for robust speech recognition technique is introduced. The proposed solution is to remove the presence of noise from the input taken as a single audio and match the patterns to recognize the input audio. The percentage of the pattern matched gives a value which is then calculated by summing up the weights, which has to cross the threshold value for any word in the database for its recognition to be successful. An error tolerance mechanism has been implemented so that it can be robust against human errors, and the surrounding environment or background noise.

Mathifier [6], by Salim N. Batlouni et al., constricts speech recognition to mathematical equations. It takes the mathematical formulas presented in the form of speech by the user and then it gives the equations in the digital mathematical form. So essentially this is a speech recognizer that is focusing on recognizing only the mathematical equations and has a very specific dictionary to be followed, with exact grammatical sentences thus allowing for lower processing time and better results.

III. Methodology

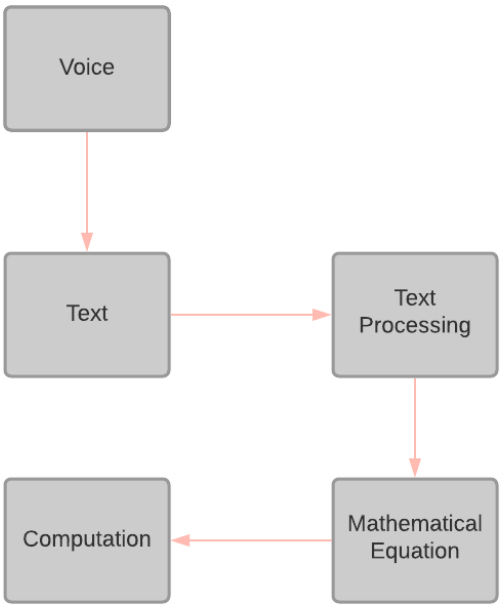

figure 1: Complete system with each phase

A. *Speech-to-Text Engine*

Attempting an end to end model to take the input voice of the mathematical query and predict the result of the mathematical equation may seem like the best choice, but the problem is the data required to train and the model itself may not learn mappings from input to output as it is hard to find the pattern in such a case.

To tackle this we have broken down the project into multiple phases, the first phase is to convert the speech into text, once we have the mathematical query in the form of text, the problem boils down to text processing, which is a well-known problem and there are effective methods to solve it, We have leveraged the Flutter Speech to text

module for the purpose of converting Speech to text, and pass for further processing.

B. Text Processing

Text processing has always been a topic of intense research. The underlying problem is the understanding of semantics by the machine and giving the expected output. Many approaches have been put forward to make the machine understand the semantics of the words and grammar, n-grams, word2vec models, LSTMs, glove models, etc. Many approaches use a combination of the above-mentioned techniques to be more effective in understanding the text.

Supervised Learning has also been profoundly used in the domain of Natural Language Processing and Text Processing. The disadvantage of Supervised Learning is that the model requires a lot of data to train on. Natural Language Processing is one of the domains that can be too abstract and non-intuitive at times. The data it requires to train should represent all the instances in the real world, which is quite a challenge.
Keeping these challenges in mind, we have resorted to a programmatic approach of processing the text, which we get from the Speech-to-Text module.

There are fixed predefined ways of saying mathematical equations. If we say it in some other way, the meaning of the query would change. We are processing these fixed ways of saying mathematical equations by iterating through the entire string of text and constructing the list of operators and operands parallelly. The elements of this list correspond to the operations and operators as they appear in the string.
The pseudo-code is as follows:

Create a Bag Of Words(operators)

For eg:
add_KW = {"sum", "add", "summation", "addition", "plus", "+"}

Initialize:

- operand_list to empty_list
- expr _list to empty_list

Iterate through the query:

- If the operand is numerical:
  Add to operand_list
- else if operand but expressed in the English word
  eg: million, generate a new operand string with the previous operands and current operand but with million replaced with its numerical value.
- Set operand_list to contain only the new operand string else:
  - Convert all the elements of operand_list to a single operand string
  - Add the obtained operand from prev step to expr _list
  - Set operand_list to empty_list
  - Add the operator to expr _list based on which bag of words the word belongs to

The next step is the conversion of these operators and operands to the corresponding postfix expression for evaluating it. The result is then displayed on the Flutter frontend.

C. Flutter Frontend

A flutter app with a simplistic and user-friendly design has been developed to take the user's voice as input. The user can perform calculations by interacting with the app with just the press of one button which serves the purpose of both inputting the mathematical expression as well as displaying the result.

This is a real-time speech calculator app that will display the text that the user is speaking in real-time and then carry out actions on them. This is advantageous as the user can see what he/she is trying to say and not forget the context of the mathematical expression they are trying to calculate.

In this application layer, the user's voice is converted to text. Then passed onto a backend powered by flask API to process the text, calculate the mathematical expression asked by the user and return the result to the frontend to be displayed.

IV. RESULTS

The app was tested on 100 queries with multiple variations:

1. 30 queries with 2 operands equations, the app was able to give correct predictions for all 30 queries.
2. 50 queries with multiple operands and operators.In this case, the app needs to take care of the precedence and associativity since the query is long, the speech to text phase itself may fail sometimes, the app was able to predict correctly 46/50 times.
3. 20 queries with queries format kept the same as the previous point but this time the app was tested with some external noise and by varying the mic distance from the user, the app was able to correctly solve 14 queries out of 20.

V. CONCLUSION AND FUTURE WORK

The effectiveness of the app is discussed in the previous section, in this section we demonstrate some of the limitations, the current logic cannot handle, the implicit precedence for instance if the user query is "add 2 and 3 multiply the result with 5", and some more cases which requires the understanding of English language itself is not handled in the current version of the app.

To tackle this, introducing NLP into the text processing phase is required as we need to understand the English language itself along with the structure of mathematical equations. To be able to do this we need a huge dataset with large amount of variations, this would be our next step and here is the form link : https://docs.google.com/forms/d/e/1FAIpQLSejwmQnFSeFqKTWo4amRNSqocErSEB1Z5XuSkhKepgLhT2gNg/viewform which have made it available for public to add the different ways one can say the mathematical query, there might be regional aspect as well that should also be represented in the dataset.

REFERENCES

[1] S. Furui, “Recent advances in robust speech recognition,” in Proc. ESCA-NATO Tutorial and Research Workshop on Robust Speech Recognition for Unknown Communication Channels, Pont-a-Mousson, France, Apr. 1997, pp.11–20.

[2] Gupta, Anshul; Patel, Nileshkumar; Khan, Shabana (2014). *[IEEE 2014 International Conference on Science Engineering and Management Research (ICSEMR) - Chennai, India (2014.11.27-2014.11.29)] 2014 International Conference on Science Engineering and Management Research (ICSEMR) - Automatic speech recognition technique for voice command.*

[3] Y.-F. Gong, “Speech recognition in noisy environments: A survey,” Speech Commun., vol. 16, pp. 261–291, 1995.

[4] B.-H. Juang, “Speech recognition in adverse environments,” Comput. Speech Lang., vol. 5, pp. 275–294, 1991.

[5] S. Young and N. Ressell, “Token passing: a simple conceptual model for connected speech recognition systems,” Cambridge University Engineering Department, 1989.

[6] Batlouni, Salim N.; Karaki, Hala S.; Zaraket, Fadi A.; Karameh, Fadi N. (2011). *[IEEE 2011 18th IEEE International Conference on Electronics, Circuits and Systems - (ICECS 2011) - Beirut, Lebanon (2011.12.11-2011.12.14)] 2011 18th IEEE International Conference on Electronics, Circuits, and Systems - Mathifier - Speech recognition of math equations.*