VOICE ACTIVATED SYSTEMS

Chinmay Bake
InstrumentationEngineeringDepartment
VishwakarmaInstituteofTechnology
Pune,India
Chinmay.bake13@vit.edu

Viren Bhale
InstrumentationEngineeringDepartment
VishwakarmaInstituteofTechnology
Pune,India
viren.bhale13@vit.edu

Agasti Dukare
InstrumentationEngineeringDepartment
VishwakarmaInstituteofTechnology
Pune,India
Agsti.dukare13@vit.edu

Dr. Sanika Patankar
InstrumentationEngineeringDepartment
VishwakarmaInstituteofTechnology
Pune,India
Sanika.patankar@vit.edu

Abstract: Voice commands can be given to drive or actuate a system. Speech recognition is the primary requirement for the said task. Speech recognition is ability of a machine or program to identify words and phrases in spoken language and convert them into a machine readable format .The recognition process involves processing of the input command and the library commands and then by matching them by using a suitable matching algorithm. Suitable software and hardware tools are used for development of a voice command activation system for applications in Home automation.

KEYWORDS: Speech recognition, matching algorithm, Home automation

I. INTRODUCTION

Speech Recognition which is also known as automatic speech recognition (ASR) and voice recognition recognizes the spoken words and phrases and converts them to a machine-readable format. By converting spoken audio into text, speech recognition technology let users to control digital devices by speaking instead of using conventional tools such as keystrokes, buttons, keyboards etc.

Speech recognition (SR) is the inter-disciplinary sub-field of computational linguistics that develops methodologies and technologies that enables the recognition and translation of spoken language into text by computers. It is also known as "automatic speech recognition" (ASR), "computer speech recognition", or just "speech to text" (STT). It incorporates knowledge and research in the linguistics, computer science, and electrical engineering fields.

Some SR systems use "training" (also called "enrollment") where an individual speaker reads text or isolated vocabulary into the system. The system analyzes the person's specific voice and uses it to fine-tune the recognition of that person's speech, resulting in increased accuracy. Systems that do not use training are called "speaker independent" systems. Systems that use training are called "speaker dependent".

Speech recognition applications include voice user interfaces such as voice dialing (e.g. "Call home"), call routing (e.g. "I would like to make a collect call"), domotic appliance control, search (e.g. find a podcast where particular words were spoken), simple data entry (e.g., entering a credit card number), preparation of structured documents (e.g. a radiology report), speech-to-text processing (e.g., word processors or emails), and aircraft (usually termed Direct Voice Input).

The term voice recognition or speaker identification refers to identifying the speaker, rather than what they are saying. Recognizing the speaker can simplify the task of translating speech in systems that have been trained on a specific person's voice or it can be used to authenticate or verify the identity of a speaker as part of a security process.

From the technology perspective, speech recognition has a long history with several waves of major innovations. Most recently, the field has benefited from advances in deep learning and big data. The advances are evidenced not only by the surge of academic papers published in the field, but more importantly by the worldwide industry adoption of a variety of deep learning methods in designing and deploying speech recognition systems. These speech industry players include Google, Microsoft, IBM, Baidu, Apple, Amazon, Nuance, SoundHound, IflyTek, CDAC many of which have publicized the core technology in their speech recognition systems as being based on deep learning.

Terminologies used in Speech Recognition

The following definitions are the basics needed for understanding speech recognition technology:

Utterance

An utterance is the vocalization (speaking) of a word or words that represent a single meaning to the computer. Utterances can be a single word, a few words, a sentence, or even multiple sentences.

Speaker Dependence

Speaker dependent systems are designed around a specific speaker. They generally are more accurate for the correct speaker, but much less accurate for other speakers. They assume the speaker will speak in a consistent voice and tempo. Speaker independent systems are designed for a variety of speakers. Adaptive systems usually start as speaker independent systems and utilize training techniques to adapt to the speaker to increase their recognition accuracy.

Vocabularies

Vocabularies (or dictionaries) are lists of words or utterances that can be recognized by the SR system. Generally, smaller vocabularies are easier for a computer to recognize, while larger vocabularies aremore difficult. Unlike normal dictionaries, each entry doesn't have to be a single word. They can be as long as a sentence or two. Smaller vocabularies can have as few as 1 or 2 recognized utterances (e.g."Wake Up"), while very large vocabularies can have a hundred thousand or more!

Accuracy

The ability of a recognizer can be examined by measuring its accuracy - or how well it recognizes utterances. This includes not only correctly identifying an utterance but also identifying if the spoken utterance is not in its vocabulary. Good ASR systems have an accuracy of 98% or more! The acceptable accuracy of a system really depends on the application.

II. PROPOSED METHODOLOGY

2.3 DYNAMIC TIME WARPING

One of the earliest approaches to isolated word speech recognition was to store a prototypical version of each word (called a template) in the vocabulary and compare incoming speech with each word, taking the closest match. This presents two problems: what form do the templates take and how are they compared to incoming signals.

The simplest form for a template is a sequence of feature vectors -- that is the same form as the incoming speech. We will assume this kind of template for the remainder of this discussion. The template is a single utterance of the word selected to be typical by some process; for example, by choosing the template which best matches a cohort of training utterances.

Comparing the template with incoming speech might be achieved via a pairwise comparison of the feature vectors in each. The total distance between the sequences would be the sum or the mean of the individual distances between feature vectors. The problem with this approach is that if a constant window spacing is used, the lengths of the input and stored sequences is unlikely to be the same. Moreover, within a word, there will be variation in the length of individual phonemes: Cassidy might be uttered with a long /A/ and short final /i/ or with a short /A/ and long /i/. The matching process needs to compensate for length differences and take account of the non-linear nature of the length differences within the words.

The Dynamic Time Warping algorithm achieves this goal; it finds an optimal match between two sequences of feature vectors which allows for stretched and compressed sections of the sequence. The paper by Sakoe and Chiba (Dynamic Programming Algorithm Optimisation for Spoken Word Recognition) gives a detailed description of the algorithm; I will summarise it here.

2.3.1 The DTW Grid

We can arrange the two sequences of observations on the sides of a grid with the unknown sequence on the bottom (six observations in the example) and the stored template up the left hand side (eight observations). Both sequences start on the bottom left of the grid. Inside each cell we can place a distance measure comparing the corresponding elements of the two sequences.

To find the best match between these two sequences we can find a path through the grid which minimises the total distance between them. The path shown in blue in gives an example. Here, the first and second elements of each sequence match together while the third element of the input also matches best against the second element of the stored pattern. This corresponds to a section of the stored pattern being stretched in the input. Similarly, the fourth element of the input matches both the second and third elements of the stored sequence: here a section of the stored sequence has been compressed in the input sequence. Once an overall best path has been found the total distance between the two sequences can be calculated for this stored template.

The procedure for computing this overall distance measure is to find all possible routes through the grid and for each one of these compute the overall distance. The overall distance is given in Sakoe and Chiba, Equation 5, as the minimum of the sum of the distances between individual elements on the path divided by the sum of the warping function (which will be discussed later). The division is to make paths of different lengths comparable.

It should be apparent that for any reasonably sized sequences, the number of possible paths through the grid will be very large. In addition, many of the distance measures could be avoided since the first element of the input is unlikely to match the last element of the template for example. The DTW algorithm is designed to exploit some observations about the likely solution to make the comparison between sequences more efficient.

III. Description of Project

2.3.4 Practical DP Matching

The Dynamic Programming algorithm is a well known optimised search algorithm which finds the least cost path in a grid. It works by first evaluating the least cost distance to reach every square in the grid and then tracing back the path which corresponds to the overall least cost distance.

The DP match algorithm presented in the paper for the simple case of no slope constraint (P=0) can be stated as follows:

- *Initialise an array g to hold the lowest cost path to each grid square.*
- *Set g(1,1) = 2d(1,1) -- ie the lowest cost to the start square is twice the distance between the two first elements (the 2 is w(1) for the symmetric form of w(k)).*
- *For each row i and column j, calculate g(i,j) as the minimum of the costs of the three possible ways to get to (i,j) plus w(k)*d(i,j). Don't consider points where |i-j| >= r -- ie outside the adjustment window.*
- *The overall distance is 1/N * g(I,J) where I and J are the lengths of the input and stored patterns respectively.*

This algorithm involves one simple calculation per grid square and relies on the observation that there are three ways to get to any grid square (horizontally, vertically or diagonally). In the end we have a measure of the cost of the best path but we don't know the path that gave rise to it. In most situations we're not really interested in the path but if we do want it it can be reconstructed by keeping track at each grid square which of the three directions gave the minimum cost path. The path can then be traced back from the endpoint.

2.4 FEATURE EXTRACTION: MFCC VECTORS

The first step in any automatic speech recognition system is to extract features i.e. identify the components of the audio signal that are good for identifying the linguistic content and discarding all the other stuff which carries information like background noise, emotion etc.

The main point to understand about speech is that the sounds generated by a human are filtered by the shape of the vocal tract including tongue, teeth etc. This shape determines what sound comes out. If we can determine the shape accurately, this should give us an accurate representation of the phoneme being produced. The shape of the vocal tract manifests itself in the envelope of the short time power spectrum, and the job of MFCCs is to accurately represent this envelope. This page will provide a short tutorial on MFCCs.

Mel Frequency Cepstral Coefficents (MFCCs) are a feature widely used in automatic speech and speaker recognition. They were introduced by Davis and Mermelstein in the 1980's, and have been state-of-the- art ever since. Prior to the introduction of MFCCs, *Linear Prediction Coefficients (LPCs) and Linear Prediction Cepstral Coefficients (LPCCs) (click here for a tutorial on cepstrum and LPCCs) and were the main feature type for automatic speech recognition (ASR), especially with HMM classifiers. This page will goover the main aspects of MFCCs, why they make a good feature for ASR, and how to implement them.*

2.4.1 Steps at a Glance:

We will give a high level intro to the implementation steps, then go in depth why we do the things we do. Towards the end we will go into a more detailed description of how to calculate MFCCs.

1. *Frame the signal into short frames.*
2. *For each frame calculate the periodogram estimate of the power spectrum.*
3. *Apply the mel filterbank to the power spectra, sum the energy in each filter.*
4. *Take the logarithm of all filterbank energies.*
5. *Take the DCT of the log filterbank energies.*
6. *Keep DCT coefficients 2-13, discard the rest.*

There are a few more things commonly done, sometimes the frame energy is appended to each feature vector. Delta and Delta-Delta features are usually also appended. Liftering is also commonly applied to the final features.

4.1 Software Analysis

Software prototypes were developed in MATLAB and in Python. The final software prototype for implementing the algorithm was developed in Python. Following are the major steps involved in the algorithm.

- *Recording of the voice command as test file*
- *Recording and reading of library files into the system*
- *Computing MFCC of the library and the voice command*
- *Using the Dynamic time warping algorithm for finding Euclidean distance between the MFCCS of voice command and the library files*
- *Minimum distance is determined*
- *The library file corresponding to the minimum distance is identified and the function associated with that respective voice command is performed*

4.2 List of modules used in Python

1. *SCIPY.IO.WAVFILE – SCIPY – It is used in importing audio library files from the system into Python*
2. *NUMPY – This module is used in computing DTW and can also be used in signal resizing(if correlation is used)*
3. *DTW - This module is used in computing DTW of the input voice command and the library files.*
4. *PYTHON_SPEECH_FEATURES – This module is used in computing MFCC of the library and voice commands*
5. *PYAUDIO (OPTIONAL) - For recording real time audio*

4.3 Algorithm in detail

STEP 1: Import all required modules in Python

STEP2: Read all the library files created. Create library files using same medium which will be used for recording input files.

STEP3: Compute the mfcc coefficients of all the library files using the same audio medium which will be used for the input voice command.

STEP4: Perform step-3 on the input voice command as well

CASE 1

STEP5: Find the distance between the MFCC assigned variables of all libraries with the input voice command one by one.

STEP6: Assign all the values with different variables

STEP7: Create an array of all those variables.

STEP8: Using if-elseif condition, if the distance is least for a particular word with the input voice commant then the task associated with that word is performed inside that loop.

STEP9: If no, then next condition is checked.

IV. Conclusion And Observations

Primary observations made out are as follows:

- *MFCC features can be used in differentiating between two audio signals.*
- *DTW and Correlation can both be used in further matching of the signals.*
- *DTW as observed is more accurate than Correlation. There is a substantial data loss in the process of finding correlation because of resizing requirement.*
- *If DTW and Correlation both are used in parallel, then the system becomes more rigid and less sensitive.*
- *If DTW is used alone then the system is supposedly more accurate.*
- *The system does not support discontinuities in any phase of the audio while comparing.*
- *Any discontinuities, excessive stretching or excessive compressing might result in False acceptance by the system.*

VIII. References

1. *Baum, L. E., Petrie, T., Soules, G. & Weiss, N. (1970), 'A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains', Ann.Math. Sastist.*
2. *Blum, T. L., Keislar, D. F., Wheaton, J. A. & Wold, E. H. (1999), Method and article of manufacture for content-based analysis, storage, retrieval, and segmentation of audio information, U.S Patent 5, 918, 223*
3. *Foote, J. T. (1997), Content-based retrieval of music and audio, in 'SPIE', pp. 138 – 147*
4. *Logan, B. T. & Chu, S. (2000), Music summarization using key phrases, in 'Proceedings IEEE Internal Conference on Acoustics, Speech, and Signal Processings*