Real Time Audio-Based Search in Media Files Using Machine Learning

[1] Swati Krishnan, [2] Sahil Raina, [3] Neha Aher

[1, 2, 3,] MIT College of Engineering, Pune, Maharashtra 411038, India

Abstract - This paper explores the various audio processing and matching methodologies available to extrapolate an algorithm which can be applied in real-time for effective audio extraction from audio-visual files and then searching for certain user defined audio patterns in said media file. With the exponential rise in multimedia content, the need to search and find information contained in these assets is a must. We propose to build tool which will enable the user to search across the spoken content of any audiovisual file chosen locally on his/her machine.

***Keywords* - Phonetic Search, Audio Indexing, Audio Retrieval, Machine Learning, Audio Acquisition, Fourier Transforms, Mel-frequency cepstral coefficients, Hidden Markov Model.**

1. Introduction

With a tidal increase in cheaper alternatives of storage space, faster processing speeds and better networking; there has been a marked increase in the usage and availability of video and audio content. This has, in turn, led to an increase in research towards analyzing the data that is stored and transferred in this form. In this paper, we are going to look at the different ways of processing and analyzing the audio content in videos so that the user can jump to the audio content that interests him/her. Being able to search by the title or a few disconnected tags, poses an insufficient solution and a major bottleneck. Currently, the best way to understand what's in a video is by examining the accompanying metadata, such as titles and captions, but that's often much narrower than what's spoken.

Most of the currently existing softwares we have surveyed search across video repositories which have already been indexed and stored. We are looking to eliminate this need for searching across already stored and transcripted data and push it to real time wherein we can search for certain keywords across any video; not just those that have been stored and indexed. But we will be looking at the methods that have been employed to convert audio into patterns that can be matched. Using existing tools like full-text search engines, natural language query or speech recognition, you'd have to transcribe the audio into a text file, then index it with a lexicon of terms that included the term or catchphrase that we want to find. Such an undertaking would be labor-intensive, time-consuming, and error-prone. The key to expediting the process would be eliminating the need for transcription or indexing or both. The main large platform software we have seen uptil now mostly facilitate searching for keywords from a certain repository of homogenous videos across the net like Mavis, which just searches across indexed video content across 10,000 informative videos.

We propose to make the need for storing videos with transcribed content unnecessary by porting various audio searching algorithms into an open source application that analyses and searches the video content in real time. So the user will have increased control about the category of videos he/she needs to mine information from. This will be done by extrapolating two mainstream audio mining methods: text based analytical approach, phonetic audio retrieval. In the first method audio to text conversion algorithms are used. A speech signal will be taken into consideration and the probability that it matches a certain set of predefined sounds will be determined. Based on this probability a word that it has the closest affiliation to will be chosen as its textual counterpart. As we can see, the boundaries of language play a very limiting role in this algorithm since only the words that are known will be included for comparison with the set of sounds made by the audio signal.

We are looking to bypass this with the inclusion of the phonetic search method which skips the audio to text conversion part and simply makes a comparison based on the phonetic string that the user wants to find and the phonemes in the audio file that is being searched. It will extract the phonemes out of the audio and create an index based on this information. Then a search will be undertaken for a match of sounds.This search will be refined by machine learning techniques especially the

Hidden Markov Model. A sound, rather than a word transcription will also lead to speeding up the matching process.

2. Present Techniques

2.1 LVCSR

This is also known as Text Based Indexing, and involves two steps. The extracted audio file of the media is processed for its speech content, using a vocabulary recognizer. This is the first step, which helps in generating a large index file containing information about the sequence of words spoken in the audio/video data. In the second step, we define the data which is to be searched in the index file, and the subsequent matching of the data (a word or a phrase). The results are then shown graphically as “Search Hits”.

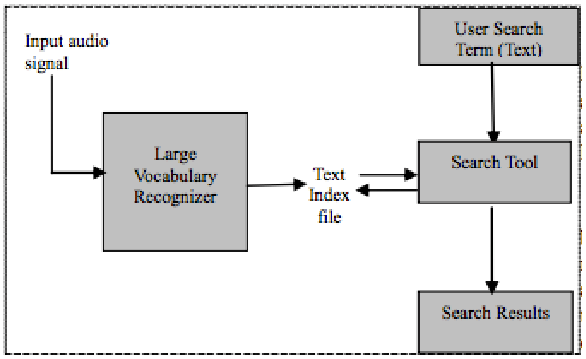


Figure 1 LVCSR

Since the basic unit is a set of words, this procedure needs to have hundreds and thousands of words to match the audio against. However, the output will be a stream of readable words, which makes it richer to work with. It can surface new business issues, the queries are much faster, and the accuracy is higher than the phonetic approach. Most importantly because the complete semantic context is in the index it is possible to find and focus on business issues very rapidly. However, this form of search takes a toll on the time required for matching the audio.

2.2 Phonetic Audio Mining

Unlike LVSCR, this doesn't convert speech to text, but works only with, as the name suggests, Phonemes.

Phonemes are the smallest units of speech in a language, and a word is formed when different phonemes are spoken together. For e.g., the utterance of long “a”, and short “a”, can be differentiated with the help of the phonemes used to identify both of them.

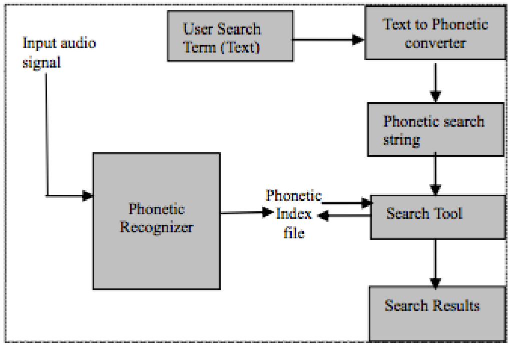


Figure 2 Phonetic Audio Mining

Phonetic Audio Mining phonetizes the entire query term, which will then be matched with the existing output phonetic string. This is the fastest approach for processing, mostly because the size of the grammar is very small, and the basic recognition unit is a phoneme. There are only few tens of unique phonemes in most languages, and the output of this recognition is a stream (text) of phonemes, which can then be searched.

3. Proposed Model for Audio Matching

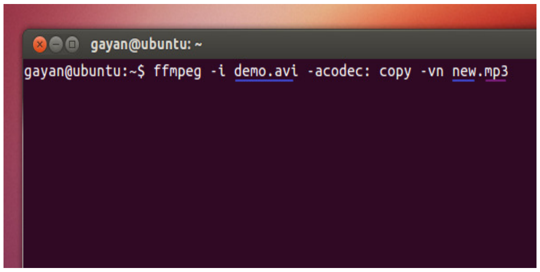
The method we propose for our research is divided into three modules:

1.Extraction and Recording
2.Conversion of Audio into Data Set.
3.Audio Matching Using Machine Learning.

Once these modules are done with, the results of the matched/unmatched audio along with their timestamps are shown.

3.1 Extraction and Recording

Extraction of the audio from the video file will be implemented using the FFMPEG2 software. The reason for choosing this software is because of the cross-platform opportunities it provides, as well as the fact that it supports the most commonly used video file formats, which includes mp4, avi, mpeg etc.



```
gayan@ubuntu:~$ ffmpeg -i demo.avi -acodec: copy -vn new.mp3
```

Fig. 3 Extraction of Audio from a Video File in Ubuntu

Recording the audio to be searched will involve making use of the inbuilt capabilities of the OS. For e.g., in Ubuntu Linux, a simple terminal command "arecord test.wav" will be able to record the spoken input. Similarly, audio recording will vary based on the operating system used.

3.2 Conversion of Audio into Data Set

In this section, we convert the raw input data into a sequence of vectors, also known as observational or feature vectors. This is the front end and is like a function that maps the input data into a sequence of feature vectors. The sequence of vectors is obtained by first splitting the data into fragments and then by applying a feature extraction algorithm to each fragment. In the case of speech, the data is a sound wave, that is a signal s[n] containing physical measurements obtained at regular time steps. The fragments are obtained by isolating short signal windows (typically 30ms long) shifted by 10-20 ms with respect to each other. The description of the speech front end will focus on the Melfrequencycepstrum coefficients extraction, maybe the most common front end in speech recognition systems.

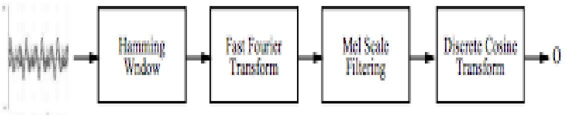


Figure. 4 MFCC Extraction block diagram

In the block diagram of the MFCC extraction, the first step is the application of the Hamming Window. The shift between two consecutive window position is around 10 ms. Such values give a good balance between the need of being short enough to detect phonemes boundaries and the need of being long enough to avoid local fluctuations. Each window isolates a segment of the raw signal which is then used for the following steps of the processing.

The second step of the MFCC extraction is the application of the Fourier Transform to each segment. The result is that the spectrum of the signal, i.e. the distribution of the energy at different frequencies is available at each window position. The graphical representation of such information is called spectrogram where the horizontal axis corresponds to the time, while the vertical one corresponds to the frequencies. The stability intervals in the spectrogram roughly correspond to the different phonemes, i.e. to the different articulator configurations used in the voicing process.

At this point of the extraction process, the original signal is converted into a sequence of 20-22 dimensional vectors where each component accounts for the energy in a critical band. The last operation is the application of a discrete cosine transform to such vectors with the goal ofde-correlating the data, i.e. of transforming the data so that the covariance between different components is null. Only the first coefficients of the DCT(in general, 12) are retained and the resulting vector is the result of the front end process. This is how we would obtain a data vector set from the raw audio. These data points will be further processed to find relevant phonemes using machine learning techniques.

3.3 Audio Matching using Machine Learning

Audio obtained from extracting the media file, and the input audio given by the user will be matched, so as to find the occurrence of the spoken input in the media file. This will essentially use the previously explained method of Phonetic Audio Mining, which will be enhanced upon by a Machine Learning method (HMM Training).

To process the extracted audio and recorded data for results, we shall use pure phonetic search. This search technique will first convert the extracted audio into a phonetic representation, rather than written words. The recorded audio will also be then converted into phoneme (sound-based) sequences, and will then be matched to the recognized sound recordings. The matching will be done by a technique, commonly referred to as Fuzzy Matching, which is basically a process of elimination of phoneme-based data, when the output is imprecise.

Thus, by eliminating the need for text-based audio search, as well as audio search specifically in pre-processed media files, the speed with which the search is conducted will increase exponentially. By our estimate, using a single core of a typical Intel processor, we would be able to search eight hours of data in just over a second. Phonetic search engines use a fraction of the hardware required by traditional solutions to deliver greater depth of audio search. It has the flexibility to use multiple search items, leading to greater accuracy and relevancy of results. And exact numbers of audio files relating to a specific topic can be easily determined, even across extremely large data-sets.

This algorithm will then be improved upon using machine learning techniques, specifically, Hidden Markov Model Training (HMM Training) for speech recognition. This training technique will require us to have two elements.
The first element necessary for training is the Training Set, i.e., the set of spoken utterances for which the transcript is available. The second element needed for training is the lexicon, which is the list of unique words that the

recognizer can give as output with their corresponding codification in terms of sub-units. As shown in the figure, for speech recognition, the word is coded with the sequence of phonemes /m /eh /r /iy corresponding to the sounds produced when uttering Mary. This will enable us to model any word using a small set of HMMs (letters in latin alphabet are 26 and the phoneme sets contain around forty elements).

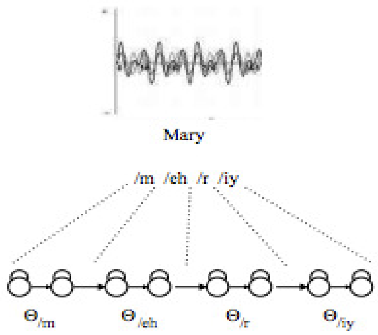


Figure 5 Lexicons for the word "Mary"

Once these two elements have been obtained, the training actually starts. This training involves three prominent steps:

1. The model corresponding to each training sample will be aligned with the sequence of vectors using the Viterbi algorithm. This will lead to a segmentation that associates each data segment to a specific subunit model.
2. Each training sample shall be processed in order to count the number of times a feature vector corresponds to a given state in a given model, the number of times a transition between two given states in each model takes place, compute the means and the variances of the feature vectors corresponding to each state in each model.
3. Counts, means and variances hence estimated will then be used to obtain estimates of initial state probabilities, transitional probabilities, means and variances of Gaussians in the mixtures, coefficients of the Gaussians in the mixtures.

The process can be stopped when a certain number of iterations have been reached or when the variation between two following estimates falls below a predefined threshold.

3.4 Results

As soon as the matching of the user-input and the extracted audio is done, the user is provided with time-stamps, each showing those points in the media where the spoken input and the extracted output are matched. The user can go through every time stamp until he/she finds the part of the media they were hoping to seek.

4. Applications

1. Use the "phonetic search" capability to jump to specific lines in scenes based on text selected from the shooting script. – Search of spoken word content in audio and video would benefit the professional video editing community. E.g.: For documentary post-production.
2. Transcriptionists at law courts where videos have to be searched for relevant testimony.
3. Television and radio networks have thousands of hours of programming but no fast way to index and negotiate them.
4. Call Centers, to help improve the quality of their customer service by searching the recorded conversations and examining them for customer needs and better customer satisfaction.

There are, of course, many other applications which will only be explored by increased usage.

5. Future Scope

1. With the help of our proposed tool, users will be enabled to share only those relevant segments of the video clip that interests them and be saved the possibility of going through the entire video.
2. Real-time audio search in media/audiovisual files, in the future, can be helpful in creating user-profiles based on the media content watched by the users, so as to sell targeted ads. Media watched by a user can be analyzed and tagged based on the words being spoken in it, and thereby targeting ads based on the content.
3. The audiovisual search could be linked to the internet so that other videos with similar catchphrases and content can be found.

6. Conclusion

In this study, we have explored the hitherto unexplored area of audio matching on a *local machine, in real time*. Our method will seek the media based on the spoken input from the user. For a general user, this study will thus help move media-seeking to its next logical conclusion. Also, with the increase in its usage, the proposed method will get better at correctly matching the recorded and the extracted audio, using machine learning, thereby increasing its accuracy in cases where background noise might hamper the results.

Acknowledgments

We would like to thank all those who assisted and guided us in every step of our project. The research required for this paper would not have been possible without their support. We are thankful to the Computer Department at MIT College of Engineering, for providing us with all the necessary resources required at various stages throughout the development of our project.

References

[1] J.L.Gauvain, A. Messaoudi and H. Schwenk. “Languagerecognition using phoneme lattices,” Proc. Int’l Conf. On Spoken Language Processing (ICSLP 2004), 2004, pp. 1283-1286.

[2] Mporas, T.Ganchev,P. ZervasandN. Fakotakis,“Recognition of Greek Phonemes using Support Vector Machines,” LNCS 3955, Springer, pp. 290-300, 2006.

[3] N.Leavitt, “Let's Hear It for Audio Mining,” IEEE Computer, Vol.35, pp. 23-25, Oct.2002.

[4] S.Shetty, and K.K. Achary, “Audio Data Mining Using Multi-perceptron Artificial Neural Network,” International Journal of Computer Science and Network Security, vol.8, pp.224-229, Oct. 2008.

[5] V.Jain and L.K. Saul, “Exploratory analysis and visualization of speech and music by locally linear embedding,” Proc. Int’l Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2004), pp. 984-987, 2004.

[6] N. Cristianini and J. Shawe-Taylor, An Introduction to Support Vector Machines. Cambridge, U.K.: Cambridge University Press, 2000.

[7] J.L.Gauvain, A. Messaoudi and H. Schwenk.“Language recognition using phoneme lattices,” Proc. Int’l Conf. on Spoken Language Processing (ICSLP 2004), 2004, pp.1283-1286

[8] W. Byrne, D. Doermann, M. Franz, S. Gustman, J. Hajic, D.Oard, M. Picheny,J. Psutka, B. Ramabhadran, D.Soergel,T.Wardand Wei-Jing Zhu. Automatic recognition of spontaneous speech for access to multilingual ora history archives IEEE Transactions on Speech and Audio Processing 12(4):420–435, 2004.

[9] A robust high accuracy speech recognition system for mobileapplications. IEEE ransactions on Speech an Audio Processing, 10(8):551–561, 2002.

Swati Krishnan Currently pursuing B.E. from the Computer Science Department of Maharashtra Institute of Technology College of Engineering, Pune (2014-2015 batch)

Sahil Raina Currently pursuing B.E. from the Computer Science Department of Maharashtra Institute of Technology College of Engineering, Pune (2014-2015 batch)

Neha Aher Currently pursuing B.E. from the Computer Science Department of Maharashtra Institute of Technology College of Engineering, Pune (2014-2015 batch)