Audio, Video Indexing & Retrieval

Project report submitted to
Indian Institute of Technology Kharagpur
in partial fulfilment for the award of the degree
of

Masters of Technology
in
Information and Communication Technology

By
Chandra Shekhar Sengupta
(Roll No:- 11IT61K14)

Under the guidance of
Dr. K. S. Rao

School of Information Technology
INDIAN INSTITUTE OF TECHNOLOGY
KHARAGPUR

April 2015

Indian Institute of Technology Kharagpur
भारतीय प्रौद्योगिकी संस्थान खड़गपुर

CERTIFICATE

This is to certify that the thesis '**Audio, Video Indexing & Retrieval**', submitted by **Chandra Shekhar Sengupta** (**11IT61K14**) in partial fulfillment of the requirements for the degree of Master of Technology in Information and Communication Technology, is a bonafide record of the work done by him in the School of Information Technology, Indian Institute of Technology Kharagpur, under my supervision and guidance.

Dr. K. S. Rao	Signature ______________________
Associate Professor	
School of Information Technology	
Indian Institute of Technology Kharagpur,	Date ______________________
India.	

DECLARATION

I certify that

a. The work contained in the report is original and has been done by myself under the guidance of my supervisor.

b. I have followed the guidelines provided by the Institute in writing the Report.

c. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

d. Whenever I have used materials (data, theoretical analysis, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references.
.

Date:-

Chandra Shekhar Sengupta

Roll No:-11IT61K14

ACKNOWLEDGEMENT

It gives me immense pleasure to express my sincere gratitude to **Dr. K. S. Rao**, under whose supervision and guidance, this work has been carried out. Without his advice and constant encouragement throughout, it would have been impossible to carry out this project work with confidence. It's really a privilege to work under his guidance.

I would like to express my sincere gratitude to **Dr. Indranil Sengupta** for encouraging me to develop an App and Web based Service for Audio, Video Indexing.

I would also like to thank all faculty members of the School of Information Technology who have been extremely kind and helpful to me and motivated me from time to time to perform my level best.

Thanks to my family, friends and colleagues for listening to my ideas about this project and encouraging me.

Thanks a lot to Researchers, Open source world and bloggers who have provided so much information related to this project.

Chandra Shekhar Sengupta
School of Information Technology
Indian Institute of Technology
Kharagpur, India

Abstract

Searching is the most performed activity in web as well as in local systems nowadays. We have so much of data around us which requires efficient searching to retrieve the most relevant results. Textual searching is becoming more advanced with the advent of latest technologies developed by Google, Microsoft. But huge amount of data available are in audio video format which are not efficiently searchable using existing search mechanism. We need better mechanism for indexing and searching audio, video data.

According to the 2011 update of Cisco's VNI IP traffic forecast, by 2015, Internet video will account for 61% of total Internet data.[1] Surprisingly in 2010, Google had only indexed .004% of the data on the internet. These statistics help us understanding the need for more effective audio, video indexing & retrieval mechanism.

This report depicts a new approach for indexing & searching audio, video files and this mechanism will be well integrated with existing search engine Google and video-sharing website YouTube so that we can utilize existing ecosystems.

CONTENTS

List of Figures

Chapter 1:

Introduction

With the advent of essentially unlimited data storage capabilities and with the rapid use of the Internet, it should be possible to access any of the stored information with a few keystrokes or voice commands. Since much of this data will be in the form of speech and video from various sources, it becomes important to develop technologies necessary for indexing and browsing such audio, video data.

Human beings have amazing ability to distinguish different types of audio, video. Given any audio/video piece, we can instantly tell the type of audio (e.g., human voice, music or noise), type of video (e.g., content, ambience or noise), speed (fast or slow), the mood (happy, sad, relaxing etc.), and determine its similarity to another piece of audio, video. .[3] However, a computer sees a piece of audio/video as a sequence of sample values. At the moment, the most common method of accessing audio, video pieces is based on their titles or file names. Due to the incompleteness and appropriateness of the file name and text description, it may be hard to find audio, video pieces satisfying the particular requirements of applications.

With more and more audio, video being captured and stored, there is a growing need for automatic audio, video indexing and retrieval techniques that can retrieve relevant audio, video pieces quickly on demand.

Thesis Organization

The thesis has been organized in the following way:-

Chapter 1 focuses on introduction, need, objective of a new audio, video indexing & retrieval system.

Chapter 2 discusses the related work in Audio, Video Indexing & Retrieval and the differentiating factor of our approach from the existing ones.

Chapter 3 Analyses its relevance by providing few case studies and proposes the Architecture & Algorithms for Audio, Video Indexing & Retrieval.

Chapter 4 contains the implementation work done so far, work in progress & future scope of work and pending areas.

Finally it concludes along with Reference resources.

Motivation (Big data & mobile technology)

The Economist reports in its 2012 Outlook that the quantity of global digital data expanded from 130 Exabyte in 2005 to 1,227 in 2010, and is predicted to rise to 7,910 Exabyte in 2015.

An Exabyte is quintillion bytes. If you find that hard to visualize, consider this: someone has calculated that if you loaded an Exabyte of data on to DVDs in slim line jewel cases, and then loaded them into Boeing 747 aircraft; it would take 13,513 planes to transport one Exabyte of data. Using DVDs to move the data collected globally in 2010 would require a fleet of more than 16 million jumbo jets. .[4]

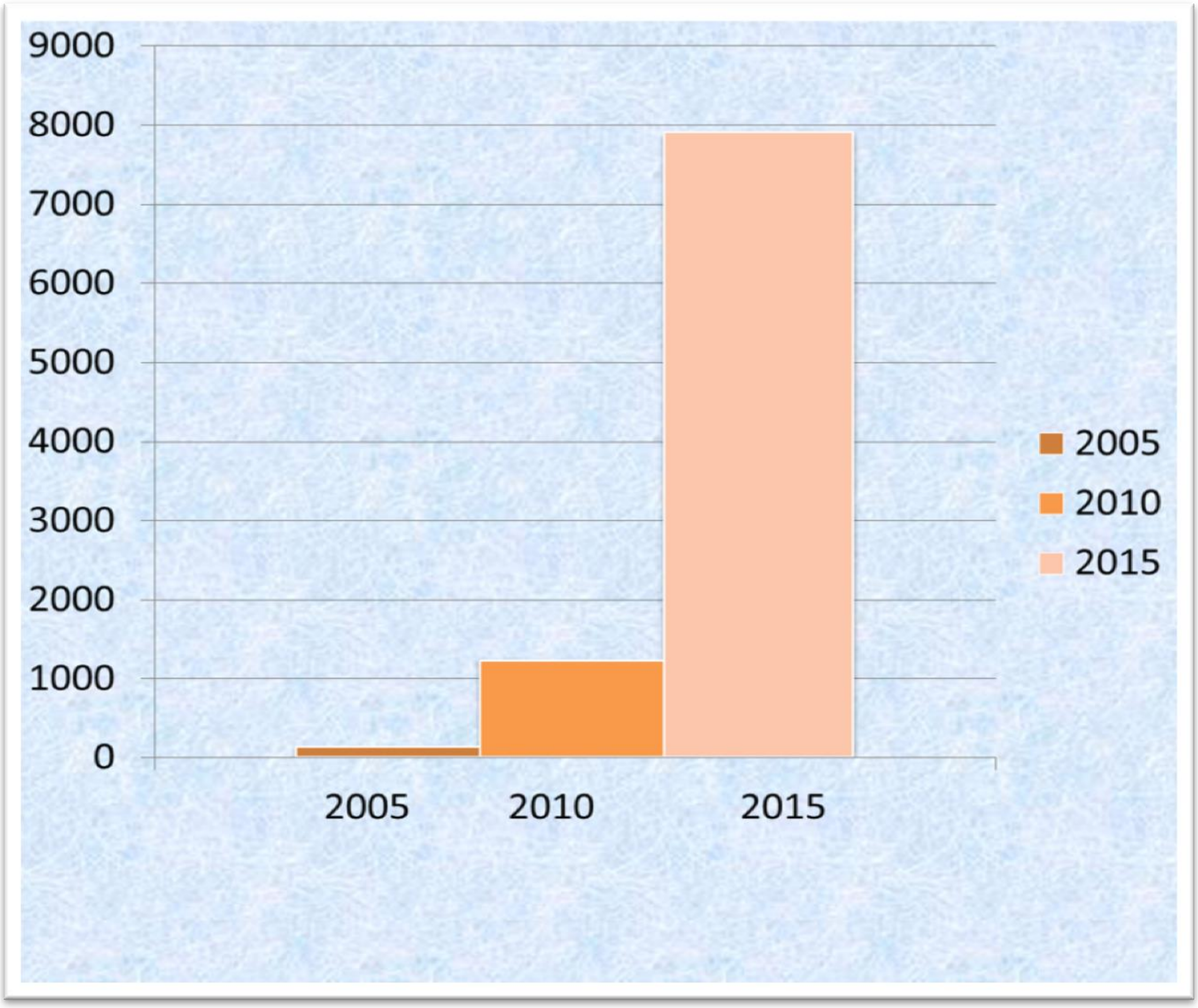


Figure 1: Amount of Global Data in last 15 years

And Exabyte are rapidly becoming passé. The volume of stored information in the world is growing so fast that scientists have had to create new terms, including zettabyte and yottabyte, to describe the flood of data. The importance of big data is not just a result of its size or how fast it is growing (about 60 per cent a year), but also the reality that the data come from an amazing array of sources. The Internet captures lots of data. Facebook alone has more than 800 million active users, more than half of whom log in every day, where they generate more than 900 million web pages and upload more than 250 million photos every day.

In 2010, a lifetime ago in Internet time, Google sites were used by more than 1 billion unique visitors every month who spent a collective 200 billion minutes on its sites. Google-owned YouTube passed 1 trillion video playbacks in 2011. Email, IM, VOIP calls, and other communications generate tens of trillions of recorded messages every year.

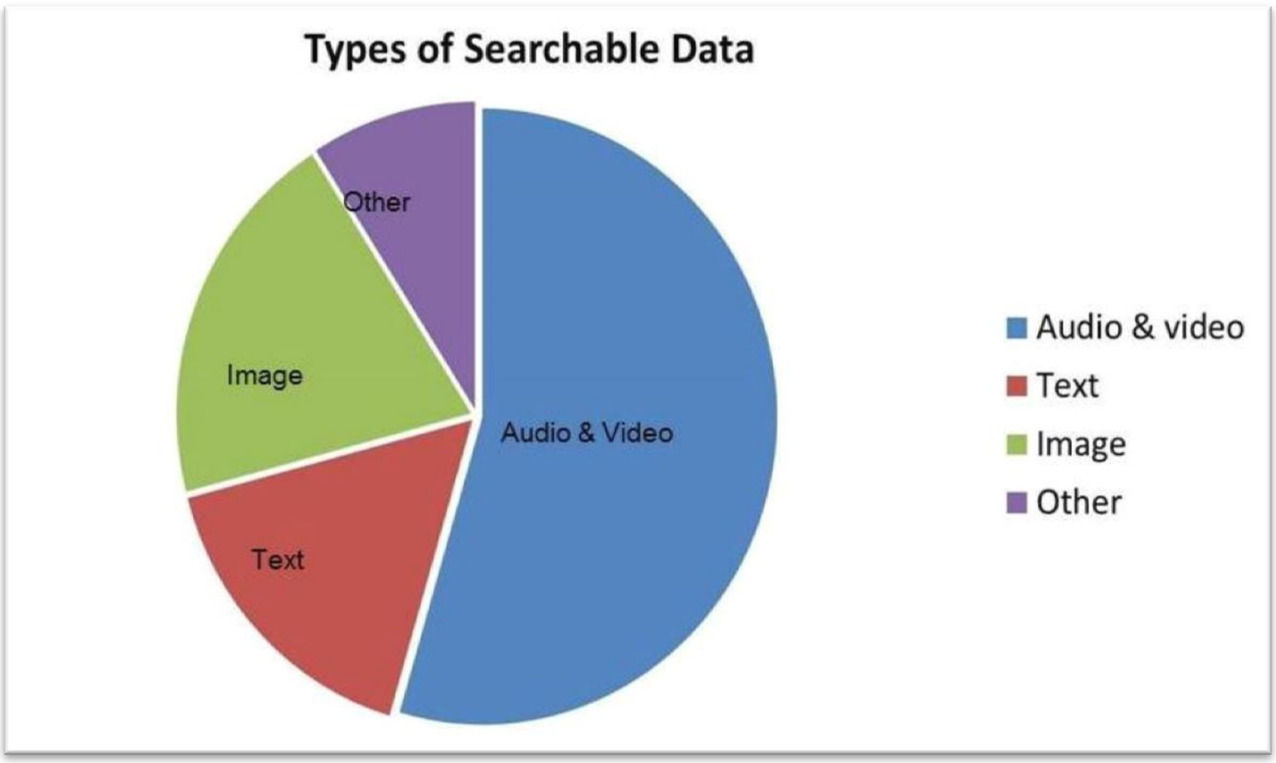


Figure 2: Types of Searchable Data in Internet and their ratio

People tend to visit the same sites frequently, so they don’t realize how truly ginormous the Internet really is. It is so big, that even with the millions of sites that Google has indexed, as of 2010; Google had only indexed **.004%** of the data on the Internet. Google had indexed 200 terabytes of the 5 million terabytes of information that existed on the Internet.

Netflix’s video-streaming service continues to be the most bandwidth-hungry application on the Internet, now accounting for a whopping 34.9% of all downstream traffic during peak periods on North American broadband networks, according to a new study.
The No. 1 subscription-video service outstripped all other services in terms of bandwidth consumption, as measured over a one-month period this fall by bandwidth-management vendor Sandvine. That’s up from Netflix’s 34.2% share in the first half of 2014 and 31.6% in the fall of 2013, according to the study. Netflix represented more than twice the

bandwidth-usage of YouTube, which accounted for 14% of all peak-period downstream traffic in North America, compared with 18.9% in the second half of 2013, according to the report.

The Cisco® Visual Networking Index (VNI) says that, Internet video & Audio will account for 61% of total Internet data by 2015.With the ever increasing amount of information being stored in audio and video formats, it is necessary to develop efficient methods for accurately extracting relevant information from these media with little or no manual intervention. [5]

Here are few statistics related to internet videos.

- **4 billion** – Number of hours of video we watched on YouTube per month.
- **60 million** – Number of global viewers monthly on Ustream.
- **14 million** – Number of Vimeo users.
- **200 petabytes** – Amount of video played on Vimeo during 2012.
- **150,648,303** – Number of unique visitors for video to Google Sites, the number one video property (September).
- **1 billion** – PSY's Gangnam Style video became the first online video to reach 1 billion views (currently just over 1.1 billion) and it achieved it in just 5 months.
- **2.7 billion** – Number of views of videos uploaded to YouTube tagged Obama or Romney during the 2012 U.S. election cycle
- **2.5 million** – Number of hours of news-related video that was uploaded to YouTube.
- **8 million** – The number of concurrent viewers of the lifestream of Felix Baumgartner's jump from the edge of space, the most ever on YouTube.
- **16.8 million** – Number of total viewers in a 24 hour period for a video on Ustream, the most ever.
- **181.7 million** – Number of total unique viewers of online video in the U.S. during December.

Few more statistics about Images

- **7 petabytes** – How much photo content Facebook added every month.
- **300 million** – Number of new photos added every day to Facebook.
- **5 billion** – The total number of photos uploaded to Instagram since its start, reached in September 2012.
- **58** – Number of photos uploaded every second to Instagram.
- **1** – Apple iPhone 4S was the most popular camera on Flickr.

From the above discussion we can figure out following points:

1. Audio-Video is the most consumed data in the internet. So we have to improve the way user consume audio video data to cop up with **growing amount of data** available.
2. Users would like to get more relevant results when they search for audio, video data. So we need better **Audio video Indexing mechanism**.
3. Most of the time we spend so much time playing a video and looking for specific information. If we can tag video content with timestamp and provide a searchable interface so that you can search and play the video from desired timestamp. So we need better **Audio video Retrieval mechanism**.

Proposed approach

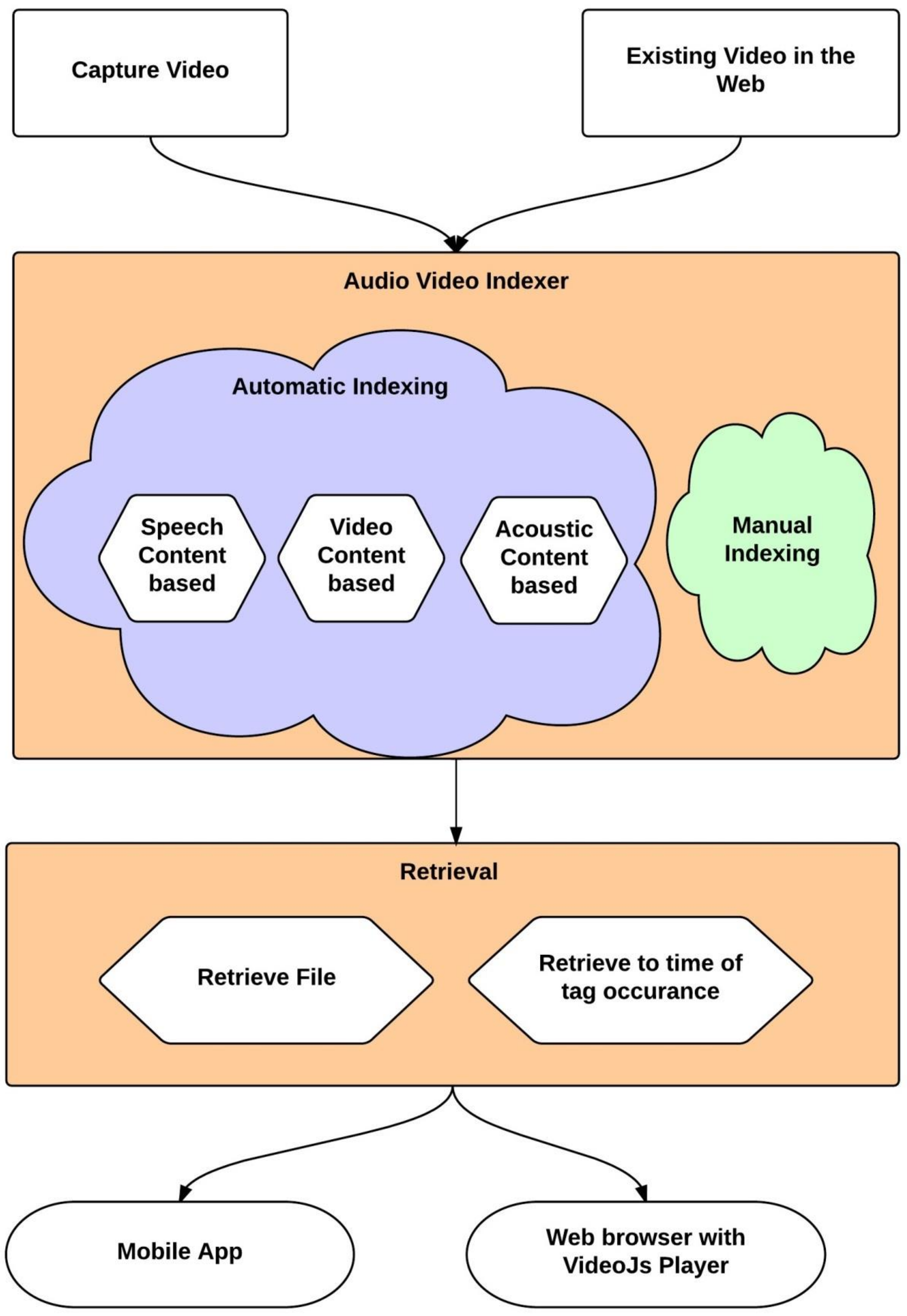


Figure 3 : Proposed Approach for Audio. Video Indexing & Retrieval

In this approach, we are extracting information about the video and audio content of the file from various aspects such as:

- Speech Content Based
    a. Speech to Text transform
    b. Existing Sub-Titles files
    c. Existing Captions
- Video Content based
    a. OCR from Video Frames
    b. Object detection from Video Frames
    c. Face detection from Video Frames
- Acoustic Content Based
    a. Background Noise Information
    b. Background Music Information
- And new approaches can be added in future as this will be a plug-in supported framework

Then we are determining most suitable tags and attaching them with file so that we can search & retrieve this file based on those tags.

We are also developing a video player which can associate tags with the video frames and retrieve to the desired video frame depending on user requirement.

Objective

To summarize the motivation & objective of this project, following points can be mentioned:

- ❑ Current web search engines generally do not enable searches into audio, video files. Informative metadata/tags would allow searches into audio, video files, but producing such metadata is a tedious manual task. So we need to come up efficient automatic indexing algorithm for audio and video files.
- ❑ There are some very good indexing & retrieval technologies available for indexing music files. But not much for speech files.
- ❑ The main objective is to come up with an audio, video **indexing technique** which incorporates unique & searchable index keys for each audio, video file along with the manually entered metadata. Index keys will be determined from various aspects of the input file and most relevant tags will be chosen to index the file.
- ❑ Another objective is to develop a suitable **retrieval technique** of the exact video/audio content with respect to specific tag.
- ❑ To enable indexing & retrieving of existing videos / audios available in the web by developing a **platform that can co-exist & support existing ones**.

Chapter 2:

Survey of Existing Works

Audio, video related indexing & searching technology can be broadly divided into following major categories.

- Speech Search
- Music Search
- Video Frame Search

Notable researches done in these fields are listed below:

Audio Video Search

Research Projects:

MUVIS

MUVIS is a framework for management (indexing, browsing, querying, summarization, etc.) of the multimedia collections such as audio/video clips and still images.

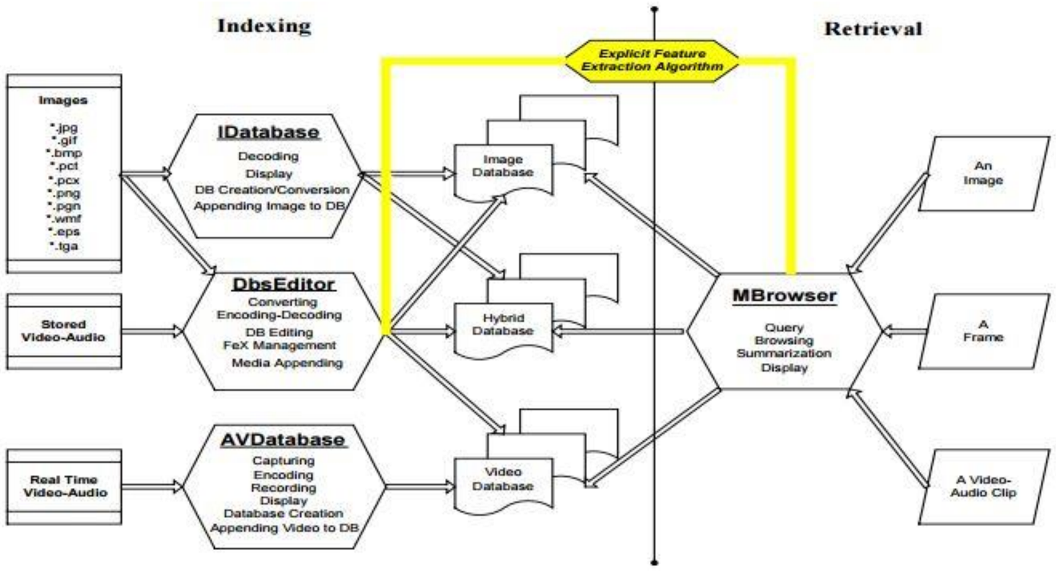


Figure 4 : General structure of MUVIS framework

MUVIS is a collaborative framework that supports indexing, browsing and querying of various multimedia types such as audio, video, audio/video interlaced in several formats. It allows real-time audio and video capturing, encoding by last generation codecs such as MPEG-4, H.263+, MP3 and AAC. MUVIS also supports several audio/video file format such as AVI, MP4, MP3 and AAC. MUVIS achieves a global and unified solution for content-based indexing and retrieval problem and provides user-friendly applications and a generic framework especially for third parties to develop their feature extraction modules.

Rough ‘n’ Ready

Rough ’n’ Ready indexes speech data, creates a structural summarization, and provides tools for browsing the stored data. The Rough ’n’ Ready system has focused entirely on the linguistic content contained in the audio signal and, thereby, derives all of its information from the speech signal. This is a conscious choice designed to channel all development effort toward effective extraction, summarization, and display of information from audio. This gives Rough ’n’ Ready a unique capability when speech is the only knowledge source.

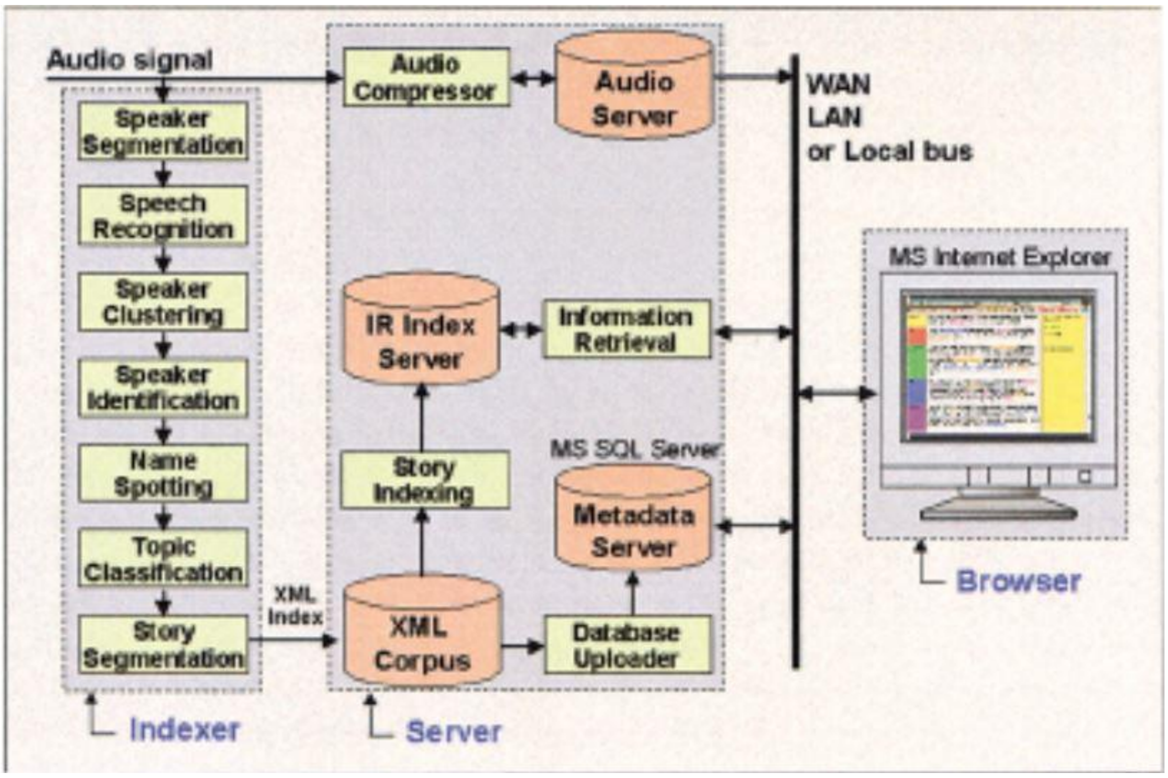


Figure 5 : General Architecture of Rough n Ready framework

Another salient feature of this system is that all of the speech and language technologies employed share a common statistical modeling paradigm that facilitates the integration of various knowledge sources.

Talk Miner

A new online video search tool launched recently makes it easier to search the content of video lectures by automatically transcribing words used in the lecturer's visual aids. TalkMiner was created by researchers at Fuji Xerox Palo Alto Laboratory (FXPAL), in California, to help students and professionals search the ever-expanding online archives of video lectures and presentations.

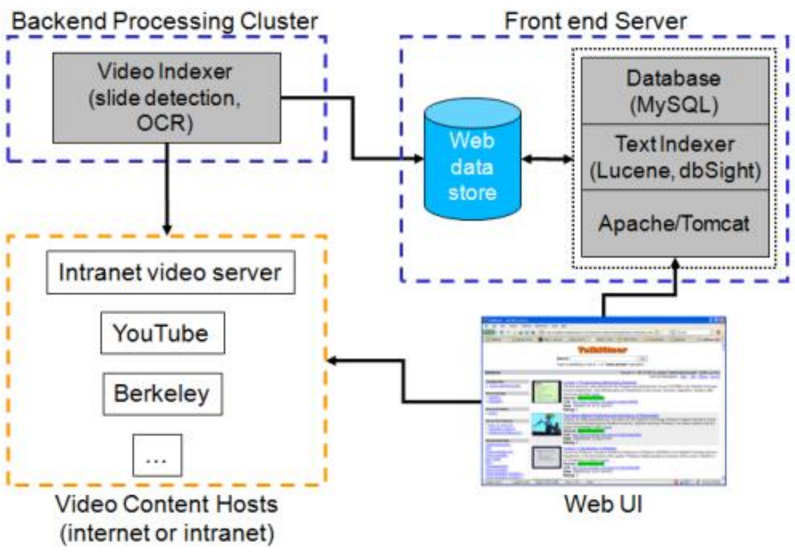

Backend Processing Cluster
Video Indexer (slide detection, OCR)
Front end Server
Web data store
Database (MySQL)
Text Indexer (Lucene, dbSight)
Apache/Tomcat
Intranet video server
YouTube
Berkeley
...
Video Content Hosts (internet or intranet)
Web UI

Figure 6 : Architecture of TalkMiner

TalkMiner builds a search index from words on the presentation slides in the source video. The system analyzes the video to identify unique slide images.

TalkMiner does not maintain a copy of the original video media files. The files are processed to produce metadata about the talk including the video frames containing slides and their time codes, and a search index constructed from the text recovered from those frames. When a user plays a lecture, the video is played from the original website on which the lecture webcast is hosted. As a result, storage requirements for the system are modest. The current prototype has indexed over 11,000 talks but it only requires approximately 11 GB of storage (i.e., 1 MB per talk).

SpeechBot

SpeechBot is a public search system (SpeechBot, 1999) similar to AltaVista which indexes audio from public Web sites such as Broadcast.com, NPR.org, Pseudo.com, and InternetNews.com. The index is updated daily as new shows are archived in their Web sites.

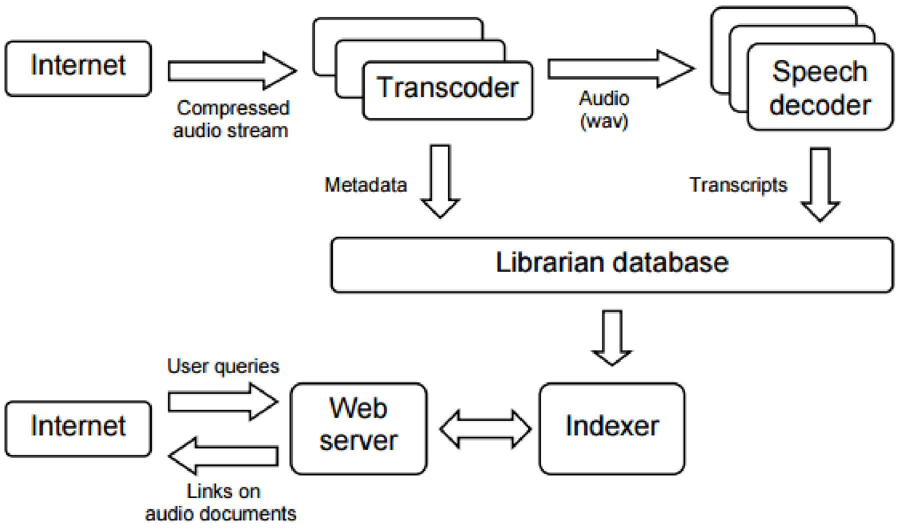


Figure 7 : Architecture of SpeechBot

SpeechBot is an audio search engine incorporating speech recognition technology. This allows indexing of spoken documents from the World Wide Web when no transcription is available. This site indexes several talk and news radio shows covering a wide range of topics and speaking styles from a selection of public Web sites with multimedia archives. This Web site is similar in spirit to normal Web search sites; it contains an index, not the actual multimedia content. The audio from these shows suffers in acoustic quality due to bandwidth limitations, coding, compression, and poor acoustic conditions. The shows are typically sampled at 8 kHz and transmitted, RealAudio compressed, at 6.5 kbps. Word-error rate results using appropriately trained acoustic models show remarkable resilience to the high compression, though many factors combine to increase the average word-error rates over standard broadcast news benchmarks. Even if the transcription is inaccurate, we can still achieve good retrieval performance for typical user queries (69%). Because the archive is large - over 5000 hours of content (and growing at a rate of 100 hours per week), totaling 47 million words and growing rapidly - we measure performance in terms of the precision of the top-ranked matches returned to the user.

Commercial Products:

Google Audio Indexing

In the 2008 presidential election race in the United States, the prospective candidates made extensive use of YouTube to post video material. Google developed a scalable system that transcribes this material and makes the content searchable (by indexing the meta-data and transcripts of the videos) and allows the user to navigate through the video material based on content. The system is available as an iGoogle gadget as well as a Labs product (labs.google.com/Gaudi). Given the large exposure, special emphasis was put on the scalability and reliability of the system.
Google's efforts to improve video search by using speech recognition technology started to become visible in July, when Google launched a gadget for searching inside the political speeches from YouTube. The gadget has been expanded into a new service called GAudi (Google Audio Indexing), which is now available at Google Labs.

Although Gaudi is removed from iGoogle as well as Google Labs, it was a great and innovative approach towards Audio Indexing and retrieval.

HP Autonomy

The main technology, 'Intelligent Data Operating Layer' (IDOL), allows search and processing of text taken from database, audio, video or text files or streams. The processing of such information by IDOL is referred to by Autonomy as Meaning-Based Computing.
Autonomy's technology attempts to "understand" any form of unstructured information, including text, voice, and video, and based on that understanding perform automatic operations, for example inferring what the user wants and on that basis finding other information that may be of interest.

Microsoft Audio Video Indexing Service (MAVIS)

The Microsoft Audio Video Indexing Service (MAVIS) is a Windows Azure application which uses state of the art speech recognition technology developed at Microsoft Research to enable searching of digitized spoken content, whether they are from meetings, conference calls, voice mails, presentations, online lectures, or even Internet video. A side benefit of MAVIS is the ability to generate automatic closed captions and keywords which can increase accessibility and discoverability of audio and video files with speech content. At this time MAVIS supports English speech content. MAVIS is now available as a commercial service through a subscription to **Greenbutton inCus**.

Music Search

Acoustic Fingerprinting

An **acoustic fingerprint** is a condensed digital summary, deterministically generated from an audio signal, which can be used to identify an audio sample or quickly locate similar items in an audio database.[1]

Practical uses of acoustic fingerprinting include identifying songs, melodies, tunes, or advertisements; sound effect library management; and video file identification. Media identification using acoustic fingerprints can be used to monitor the use of specific musical works and performances on radio broadcast, records, CDs and peer-to-peer networks. This identification has been used in copyright compliance, licensing, and other monetization schemes.

Shazam

Shazam have developed and commercially deployed a flexible audio search engine. The algorithm is noise and distortion resistant, computationally efficient, and massively scalable, capable of quickly identifying a short segment of music captured through a cellphone microphone in the presence of foreground voices and other dominant noise, and through voice codec compression, out of a database of over a million tracks. The algorithm uses a combinatorial hashed time-frequency constellation analysis of the audio, yielding unusual properties such as transparency, in which multiple tracks mixed together may each be identified.

Query By Humming

Query by humming (**QbH**) is a music retrieval system that branches off the original classification systems of title, artist, composer, and genre. It normally applies to songs or other music with a distinct single theme or melody. The system involves taking a user-hummed melody (inputquery) and comparing it to an existing database. The system then returns a ranked list of music closest to the input query.

One example of this would be a system involving a portable media player with a built-in microphone that allows for faster searching through media files.

The MPEG-7 standard includes provisions for QbH music searches.

SoundHound

SoundHound is revolutionizing the way people interact with mobile devices by delivering innovative technologies and compelling user experiences in sound recognition. SoundHound's breakthrough Sound2Sound technology searches sound against sound, bypassing traditional sound to text conversion techniques even when searching text databases. Sound2Sound has resulted in numerous breakthroughs including the world's fastest music recognition, the world's only viable singing and humming search, and instant-response large scale speech recognition systems.

Sound2Sound (S2S) Search Science

Sound2Sound (S2S) Search Science is a revolutionary technology, created by SoundHound, capable of recognizing various sound inputs including music and speech. It offers a breakthrough combination of speed and accuracy unattainable through traditional approaches to sound recognition.S2S performs recognition by extracting features from the input signal and converting them to a compact and flexible Crystal representation. This Input Crystal is then matched against a database of Target Crystals which have been derived from searchable content.

Tunebot

Tunebot is a music search engine developed by the Interactive Audio Lab at Northwestern University. Users can search the database by humming or singing a melody into a microphone, playing the melody on a virtual keyboard, or by typing some of the lyrics. This allows users to finally identify that song that was stuck in their head.

Searching Techniques

Tunebot is a Query by humming system. It compares a sung query to a database of musical themes by using the intervals between each note. This allows a user to sing in a different key than the target recording and still produce a match. The intervals are also unquantized to allow for other tunings besides the standard A=440Hz, since not many people in the world have perfect pitch.

Query by Melody

Music Retrieval based on Melodic Similarity, which is one type of content-based retrieval, is proposed. Every fragment of a melody is represented as an appropriate N-dimensional vector, called a feature vector. A feature vector consists of values corresponding to intervals and rhythm of a melody. Depending on the distance between every fragment of a melody in the music database to the entered melody, k-nearest melodies can be obtained quickly using the SS-tree.

Musipedia

Musipedia offers three ways of searching:

- Based on the melodic contour,
- based on pitches and onset times, or
- based on the rhythm alone.

The melodic contour search uses an editing distance. Because of this, the search engine finds not only entries with exactly the contour that is entered as a query, but also the most similar ones among the contours that are not identical. Similarity is measured by determining the editing steps (inserting, deleting, or replacing a character) that are needed for converting the query contour into that of the search result. Since only the melodic contour is relevant, one can find melodies even if the key, rhythm, or the exact intervals are unknown.

The pitch and onset time-based search takes more properties of the melody into account.

The "query by tapping" method that only takes the rhythm into account uses the same algorithm as the pitch and onset time method, but assumes all pitches to be the same. As a result, the algorithm can be used for tapped queries that only contain onset times.

Reverse Image Search

Reverse image search is content-based image retrieval (CBIR) query technique that involves providing the CBIR system with a sample image that it will then base its search upon; in terms of information retrieval, the sample image is what formulates a search query. In particular, reverse image search is characterized by a lack of search terms. This effectively removes the need for a user to guess at keywords or terms that may or may not return a correct result. Reverse image search also allows users to discover content that is related to a specific sample image, popularity of an image, and discover manipulated versions and derivative works.

TinEye

TinEye is a reverse image search engine. You can submit an image to TinEye to find out where it came from, how it is being used, if modified versions of the image exist, or to find higher resolution versions. TinEye is the first image search engine on the web to use image identification technology rather than keywords, metadata or watermarks. It is free to use for non-commercial searching.

TinEye regularly crawls the web for new images, and we also accept contributions of complete online image collections. To date, TinEye has indexed 10,974,401,835 images from the web to help you find what you're looking for.

Google Search By Image

Google Images has a Search by Image feature for performing reverse image searches. Unlike traditional image retrieval, this feature removes the need to type in keywords and terms into the Google search box. Instead, users search by submitting an image as their query. Results may include similar images, web results, pages with the image, and different resolutions of the image. For example, if you search using a picture of a beach, you might get results including similar beaches, information about the same beach, and websites that use the same beach picture.

The precision of Search by Image's results is higher if the search image is more popular. Additionally, Google Search by Image will offer a "best guess for this image" based on the descriptive metadata of the results.

Bing Image Match

Called Image Match, Bing’s reverse image search offers a more streamlined version of what Google has been offering; only displaying the information you absolutely need, as well as a multiple sizes of the image you’re looking up. On top of all that, the Image Match feature is incredibly easy to use. Bing has made it as easy as possible to find the Image Match tool. All you need to do is go to “bing.com/images” here and click on “Image Match,” which will be displayed to the right of the search field at the top of the screen. After adding an image’s URL or uploading an image, Bing will display the results it finds on a search results page.

With Bing, site descriptions are omitted, giving a much cleaner look. The various sizes that are available are also listed at the top of the screen.

Yandex Image Search

Russian Search Engine giant Yandex provides a reverse image search service as well. You can find similar images or group of images easily and quickly using Yandex image search engine.

Differentiating & Unique factors in our Approach

Our approach has following differentiating & unique factors which makes this worth implementing:

1. Integration of index keys available from three major components of a video/audio file
2. Index key determination algorithm(tag cloud) based on relevancy
3. Categorization of Index keys based on Acoustic information
4. Index keys with time frame
5. New file format for storing index keys for audio/video files as plug-n-play
6. Most of the existing research projects for audio video indexing do not integrate properly with existing audio video websites like Youtube, Vimeo, Netflix, Ustream etc. We will enable integration with existing platforms.
7. Integration with YouTube upload and existing contents of Youtube
8. Most of the existing research projects are not optimized for audio video indexing in mobile devices. But most of the audio/videos are being captured by mobile devices nowadays. We are optimizing this platform for providing audio/video indexing capabilities while capturing via mobile device.
9. Retrieval from mobile device
10. Retrieval from web
11. Retrieval by time frame
12. New video player extension for supporting this audio/video indexing approach

Chapter 3:

Architecture

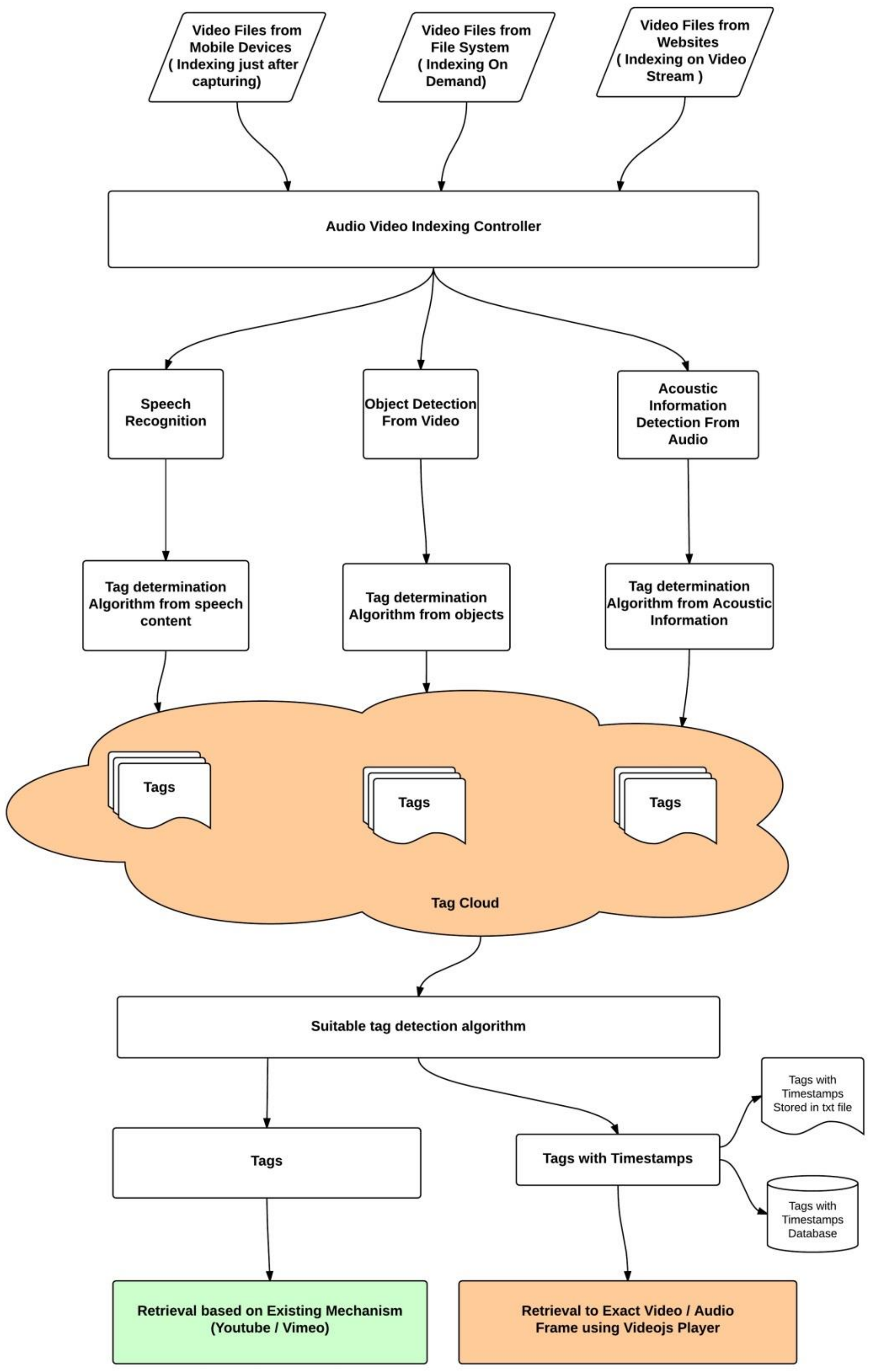

Video Files from Mobile Devices (Indexing just after capturing)
Video Files from File System (Indexing On Demand)
Video Files from Websites (Indexing on Video Stream)
Audio Video Indexing Controller
Speech Recognition
Object Detection From Video
Acoustic Information Detection From Audio
Tag determination Algorithm from speech content
Tag determination Algorithm from objects
Tag determination Algorithm from Acoustic Information
Tags
Tags
Tags
Tag Cloud
Suitable tag detection algorithm
Tags
Tags with Timestamps
Tags with Timestamps Stored in txt file
Tags with Timestamps Database
Retrieval based on Existing Mechanism (Youtube / Vimeo)
Retrieval to Exact Video / Audio Frame using Videojs Player

Figure 8: Overall Architecture

In this approach Audio Video indexing controller will extract following information from input file and route them corresponding processors.

- Speech Content → Speech Content Processor
- Image Object → Image Object Processor
- Acoustic Information → Acoustic Information processor

These processors will segregate the specific contents and pass through corresponding **Tag determination Module**.

Now tags from all sources will form a **Tag cloud** for the input file. Now we will run **Relevant Tags Detection Algorithm** to choose most suitable tags for the input file. Now finalized tags will be associated with the input file. Timestamps associated with tags will be stored in Timestamp database.

While audio retrieval there are two options available:

1. Retrieve based on tags from existing websites like YouTube , Vimeo
2. Retrieve based timestamp using our **tag with timestamp database.**

Algorithm

Speech Recognition from Audio Video

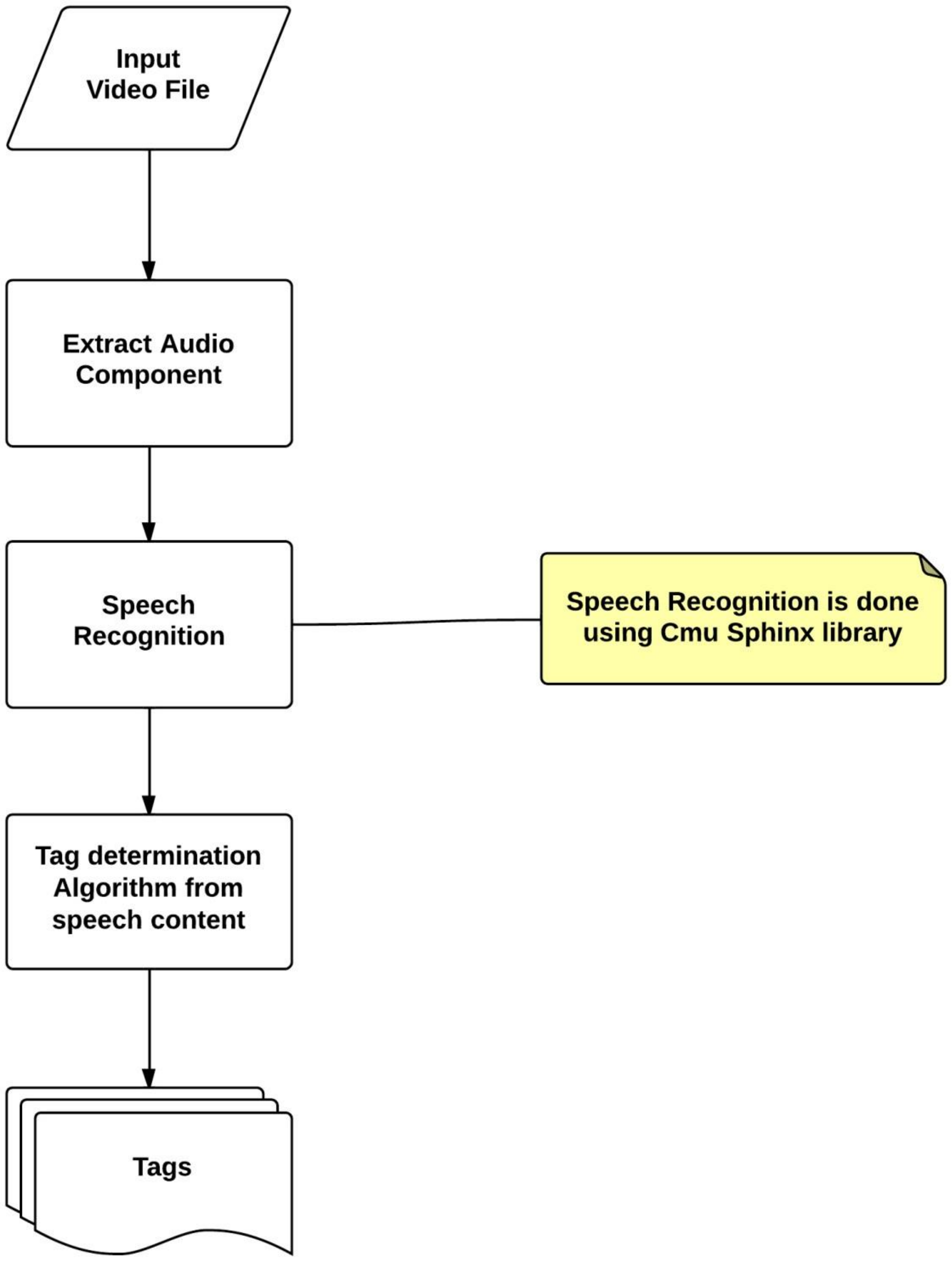

Input Video File
Extract Audio Component
Speech Recognition
Speech Recognition is done using Cmu Sphinx library
Tag determination Algorithm from speech content
Tags

Figure 9: Speech Recognition from Audio, Video

Tag Determination Algorithm for texts from recognized speech

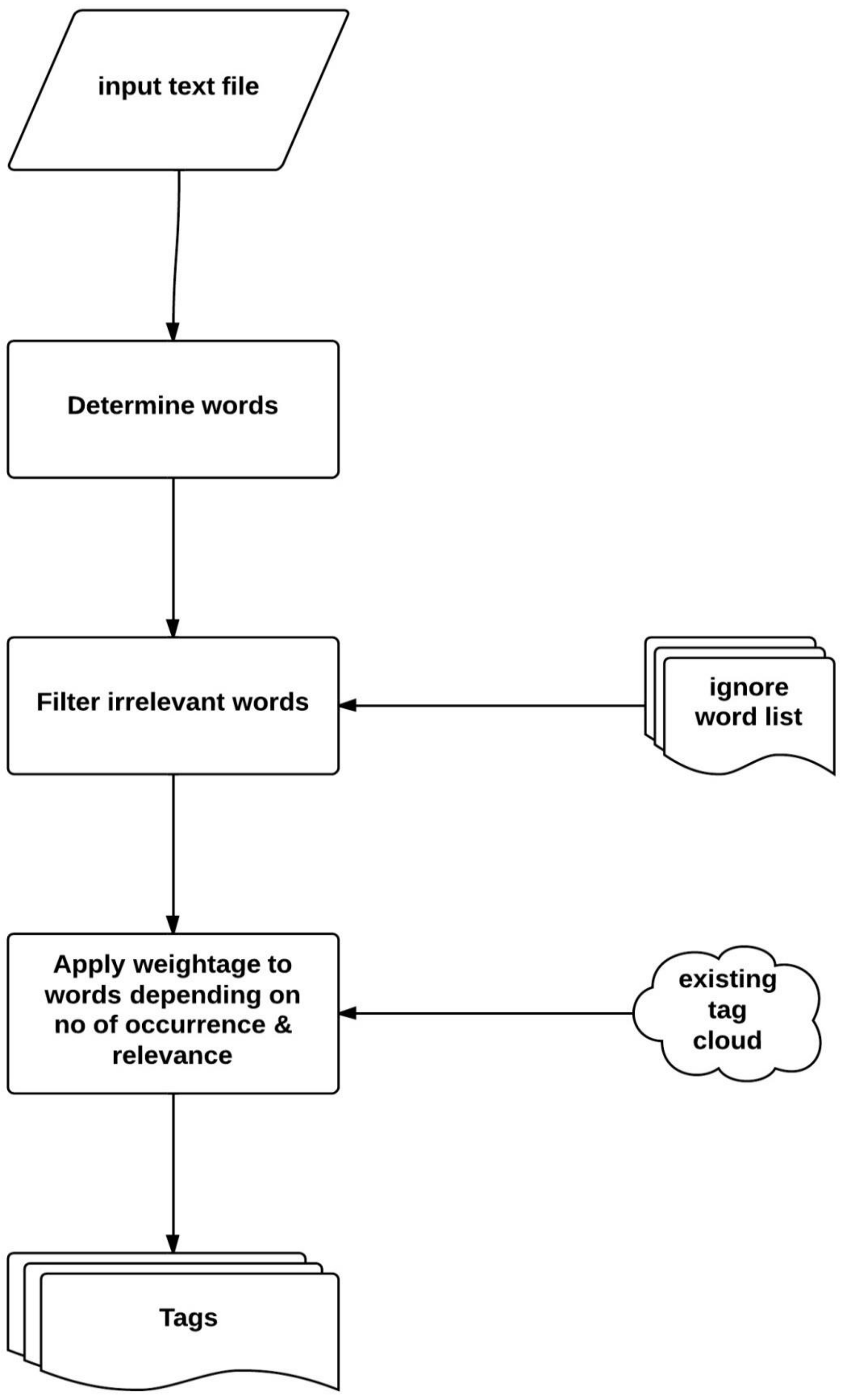
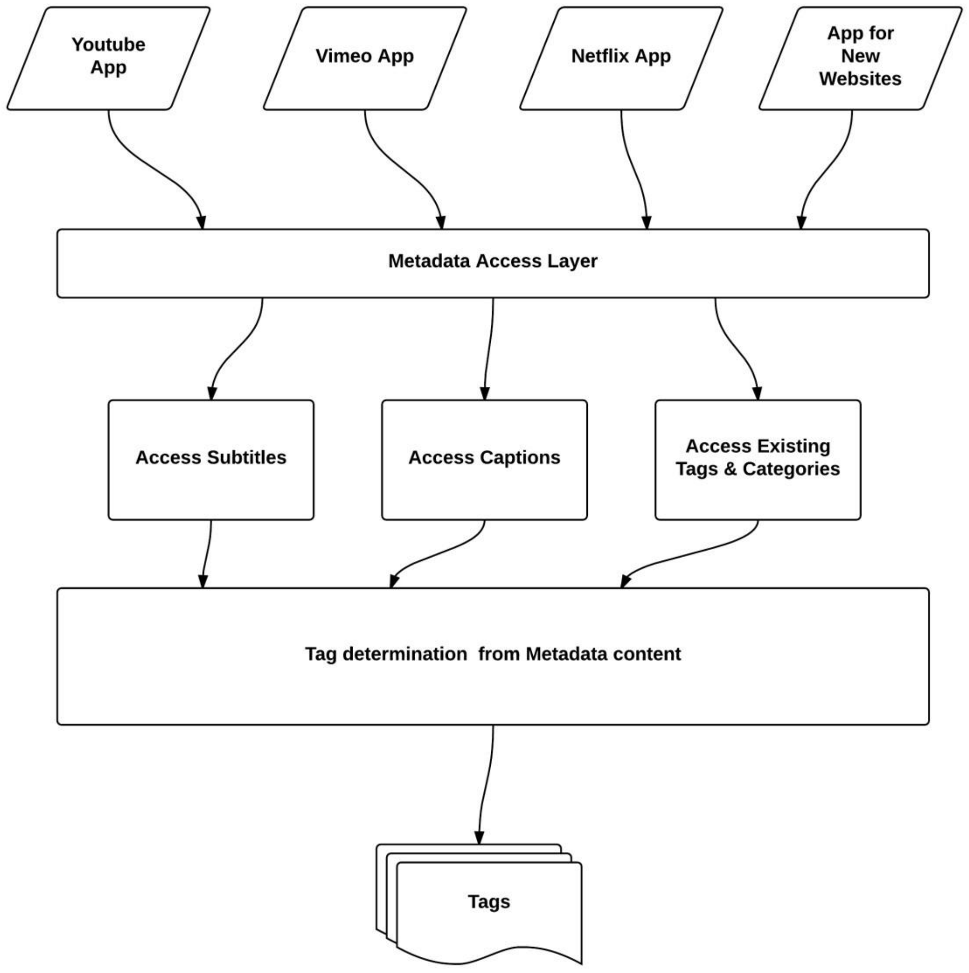


Figure 10: Tag detection from recognized speech

Tag Detection from Video Metadata Available from Websites



Object Detection from Video

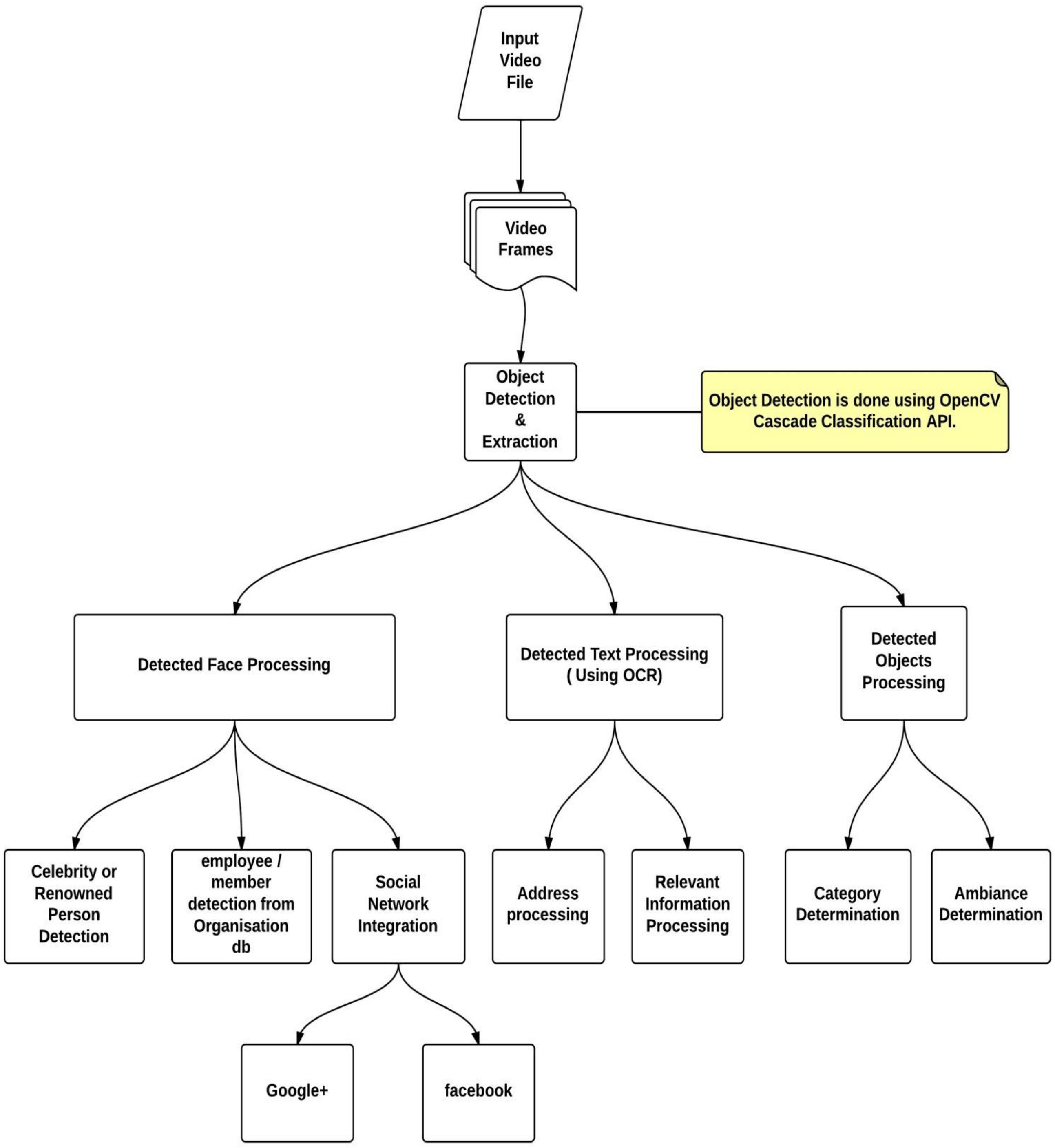

Input Video File
Video Frames
Object Detection & Extraction
Object Detection is done using OpenCV Cascade Classification API.
Detected Face Processing
Detected Text Processing (Using OCR)
Detected Objects Processing
Celebrity or Renowned Person Detection
employee / member detection from Organisation db
Social Network Integration
Address processing
Relevant Information Processing
Category Determination
Ambiance Determination
Google+
facebook

Figure 11: Object Detection from Video

Acoustic Information Detection from Audio

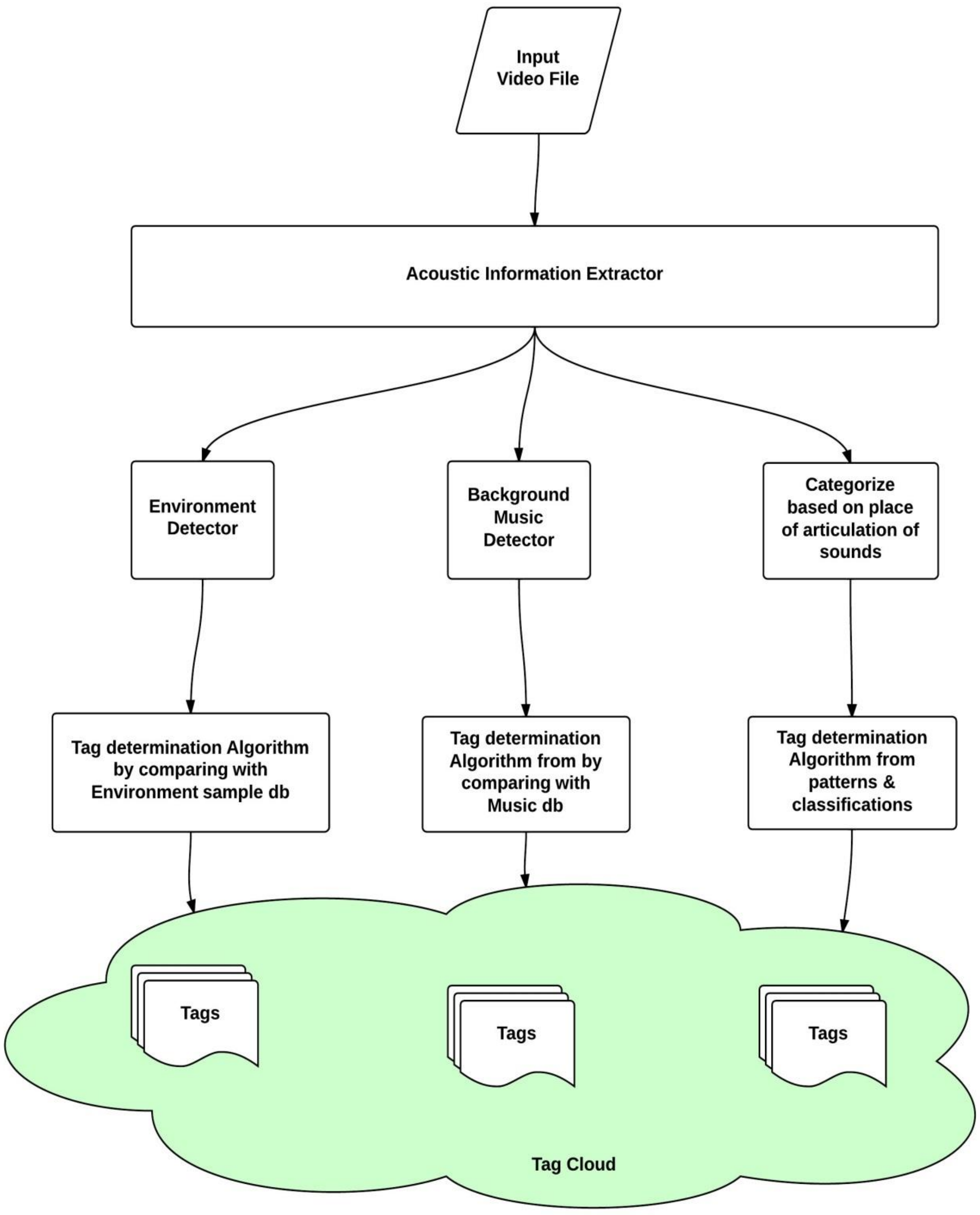
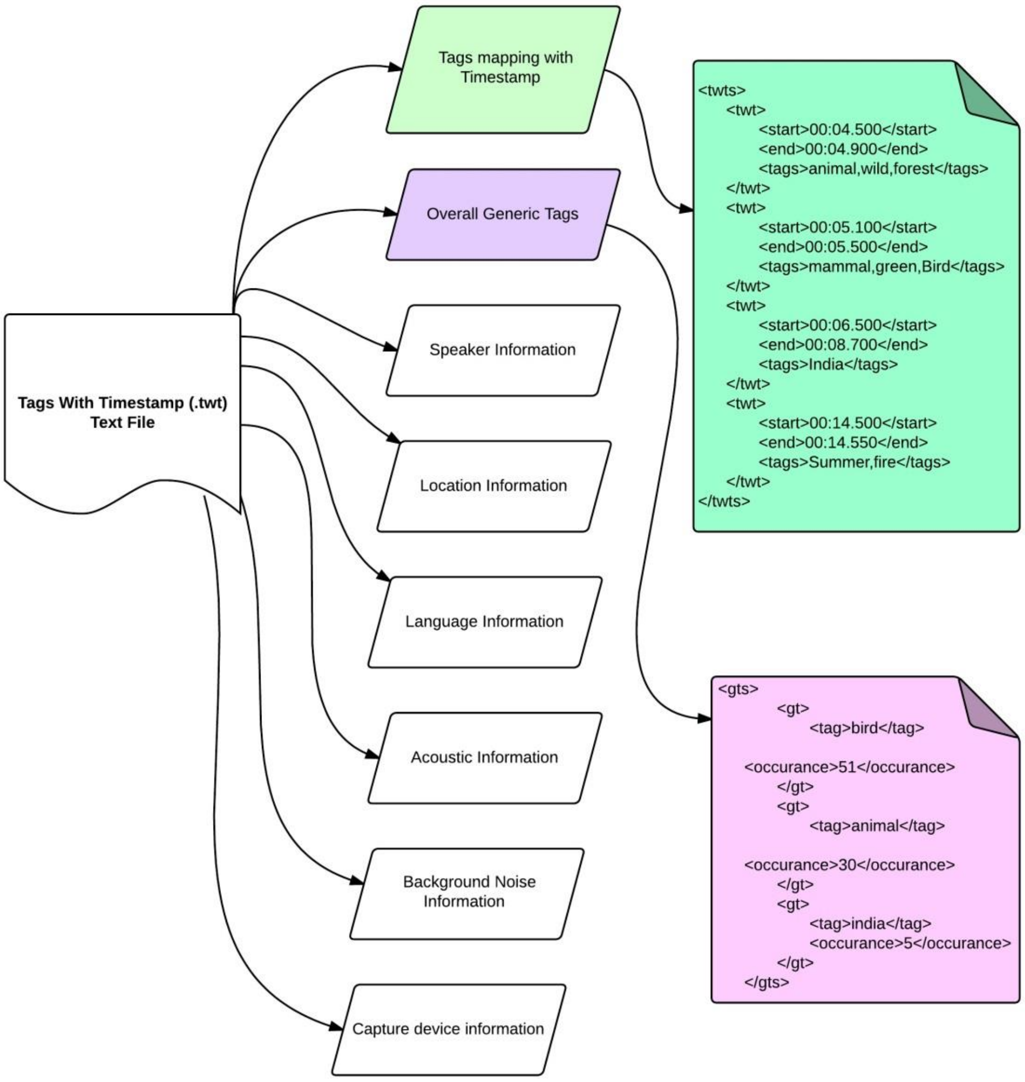

Input Video File
Acoustic Information Extractor
Environment Detector
Background Music Detector
Categorize based on place of articulation of sounds
Tag determination Algorithm by comparing with Environment sample db
Tag determination Algorithm from by comparing with Music db
Tag determination Algorithm from patterns & classifications
Tags
Tags
Tags
Tag Cloud

Figure 12: Acoustic Information Detection from Audio

Tag With Timestamps File Format


Tags With Timestamp (.twt) Text File
Tags mapping with Timestamp
Overall Generic Tags
Speaker Information
Location Information
Language Information
Acoustic Information
Background Noise Information
Capture device information

```
<twts>
    <twt>
        <start>00:04.500</start>
        <end>00:04.900</end>
        <tags>animal,wild,forest</tags>
    </twt>
    <twt>
        <start>00:05.100</start>
        <end>00:05.500</end>
        <tags>mammal,green,Bird</tags>
    </twt>
    <twt>
        <start>00:06.500</start>
        <end>00:08.700</end>
        <tags>India</tags>
    </twt>
    <twt>
        <start>00:14.500</start>
        <end>00:14.550</end>
        <tags>Summer,fire</tags>
    </twt>
</twts>
```

```
<gts>
        <gt>
            <tag>bird</tag>

    <occurance>51</occurance>
        </gt>
        <gt>
            <tag>animal</tag>

    <occurance>30</occurance>
        </gt>
        <gt>
            <tag>india</tag>
            <occurance>5</occurance>
        </gt>
    </gts>
```

Figure 13: Tags with Timestamps File Format

Tags with Timestamps file format can be represented in XML or JSON. In the scope this project we will work on following two elements

- Tags With Timestamps (twts)
  - Start time (start)
  - End time (end)
  - Tags (tags)
- Generic Tags (gts)
  - Tag Name (tag)
  - Number of Occurrence (occurrence)

The root element of all kinds of tags is <**avir**> (Audio Video Indexing & Retrieval).

Tag incorporation Algorithm

We are going to follow a generic approach for incorporating tags with indexed Video/Audio Files so that we can support almost all the videos available online and offline.

Video files can be accessed from following sources –

- File system (offline)
- Web (online)

Corresponding Tag/index information of video/audio file can reside in following ways

- a file available in file system or hosted in web
- a centralized database
- stored as EXIF information of the video file
- Pushed to Host Website's (Youtube , Vimeo) tags
- Integrate with Operating System Search Index

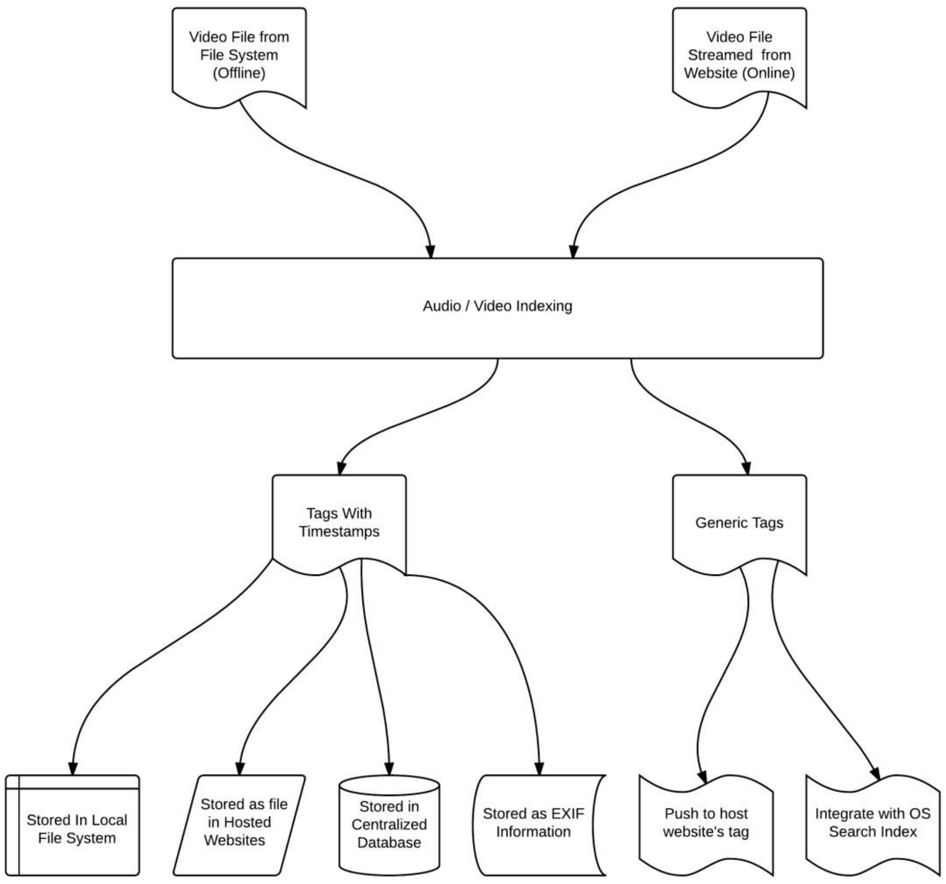

Video File from File System (Offline)
Video File Streamed from Website (Online)
Audio / Video Indexing
Tags With Timestamps
Generic Tags
Stored In Local File System
Stored as file in Hosted Websites
Stored in Centralized Database
Stored as EXIF Information
Push to host website's tag
Integrate with OS Search Index

Figure 14: Tag Incorporation Mechanism

Indexing & Retrieval Environment

The scope of indexing and retrieval of audio/video files depend on the environment as well.

Individual Tag File based

Our Audio Video Indexing Mechanism will generate tag information and store it as a separate text file. User can use this tags file to browse through the corresponding video file. User can choose not to integrate this information with a centralized database and Operating System's Search Indexes.

Centralized Tags Database based

Centralized Tags Database can enable searching through all the video files with indexed keys along with timestamp based retrieval as well. This architecture can be deployed in local system as well as in websites.

Integrated with Operating System Search Index

We can integrate indexes derived from this approach with Windows operating systems federated search index repositories. This will enable efficient and faster search and retrieval of the indexed audio/video files.

Video Retrieval Algorithm

Our indexing approach can integrate index keys as tags in various ways as mentioned earlier. So there can be following two types of retrieval mechanism:

Video with Generic Tags

Video with generic tags should not require any special mechanism for retrieval other than the existing ones. Normal search queries with the desired tags should fetch the video file and generic video player should be available to play the video.

And Video websites also should fetch the video with desired search keywords as tags are already pushed to website's indexing platform.

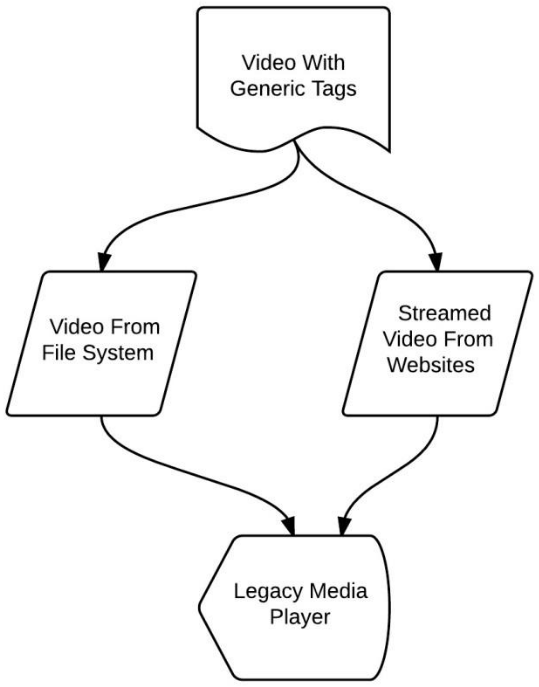


Figure 15: Video With Generic Tags Retrieval Mechanism

Video Frame by Tags with Timestamp

We can retrieve video frames by searching for associated tags using enhanced VideoJs Player. Our indexing mechanism associates tags with timestamp of the video frame and saves into the **Tags With Timestamps (TWT)** File. Depending on the location of Video and TWT file there are six combinations as described in the below diagram. We have enhanced VideoJs Player to support all six combination of retrieval.

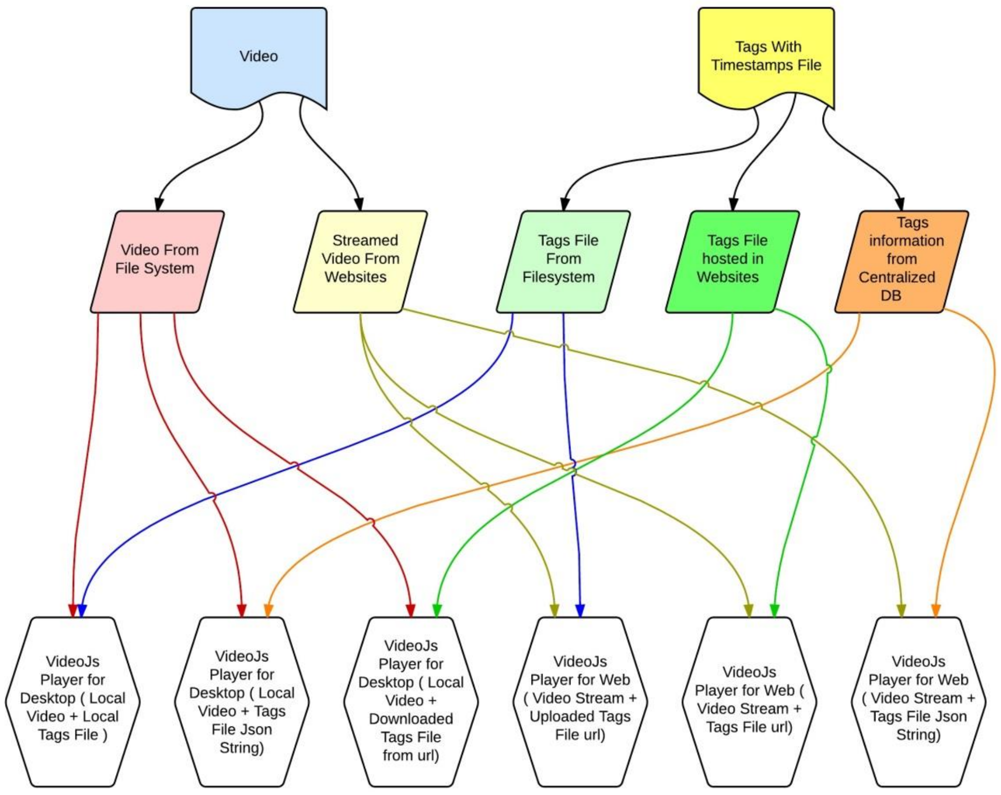

Video
Tags With Timestamps File
Video From File System
Streamed Video From Websites
Tags File From Filesystem
Tags File hosted in Websites
Tags information from Centralized DB
VideoJs Player for Desktop (Local Video + Local Tags File)
VideoJs Player for Desktop (Local Video + Tags File Json String)
VideoJs Player for Desktop (Local Video + Downloaded Tags File from url)
VideoJs Player for Web (Video Stream + Uploaded Tags File url)
VideoJs Player for Web (Video Stream + Tags File url)
VideoJs Player for Web (Video Stream + Tags File Json String)

Figure 16: Video Frame Retrieval By Tags With Timestamp

Applications & Tools to be developed for this Approach

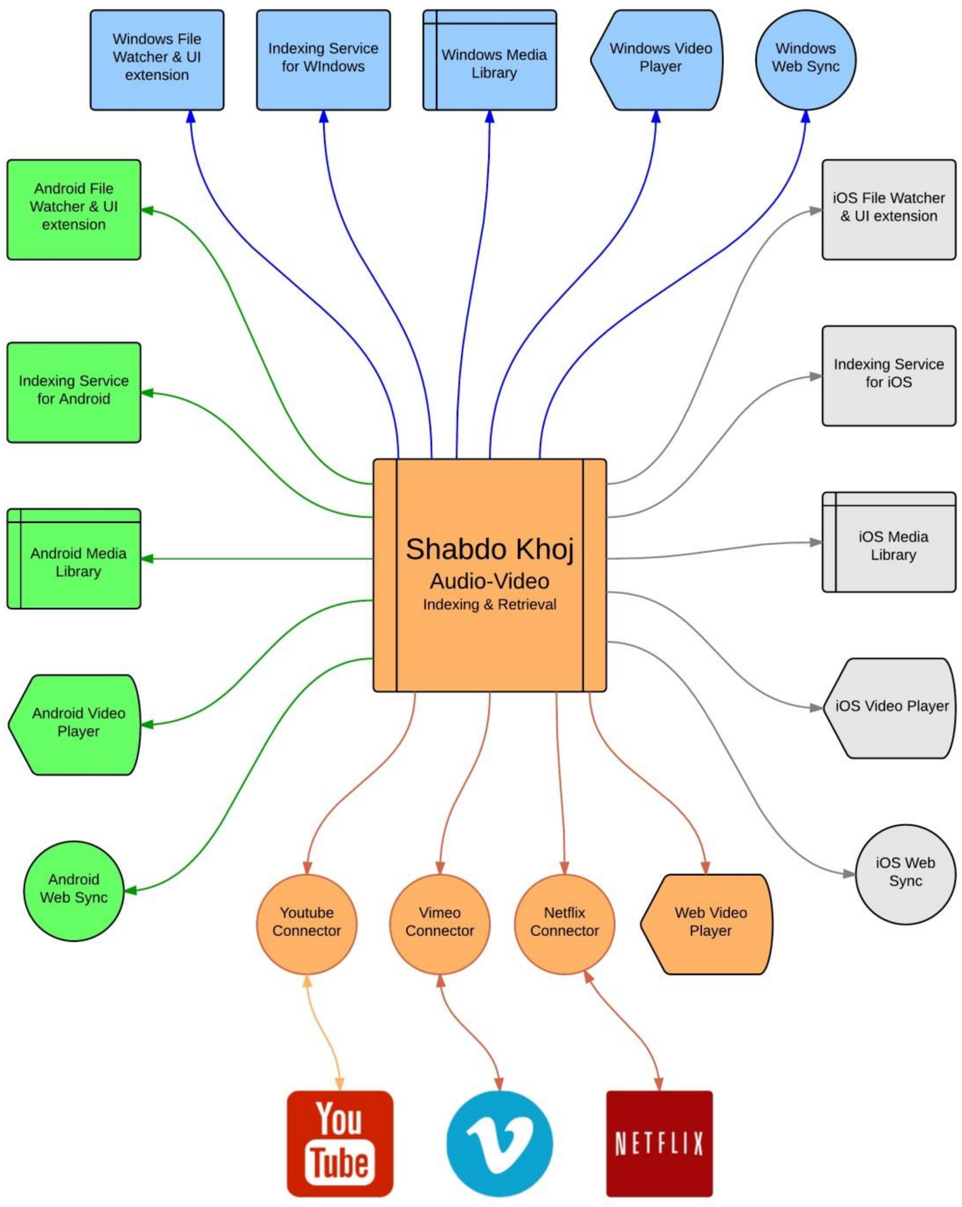


Figure 17: ShabdoKhoj Applications & Tools

Case Studies

Single Video File from File System indexed for future retrieval

Problem

1. 35 min long video file present in the file system.
2. Video consists of sports news update of EPL & La Liga football teams.
3. User is interested to watch news portion related to EPL club Manchester United and La Liga Club Barcelona.
4. User does not want to spend more than 2 min of time watching the relevant content.

Solution

Indexing

1. User can open up VideoJs Player With **ShabdoKhoj** Extension
2. User queues the video for indexing.
3. Index the video file from OCR text, Recognized Speech, Metadata etc and record corresponding time stamp with tags.
4. Create a tag cloud; determine suitable tags with timestamp for the file.
5. **Tags with Timestamp** for the video is saved in the file system
6. Let's say, Barcelona, Manchester United, Man U tags appear in the 5th min , 19th Min and 27th Min.

Searching & Retrieval

1. **Tags With Timestamp** file for the Video file is linked to VideoJs Player UI after completion of Indexing.
2. Now user can search the tags to see when 'barca' tags are present (e.g.: 5th min). User can fast forward to 5th min and watch the relevant content
3. User can add this video file and its indexing information to **ShabdoKhoj** Library for future use

Multiple Videos File from Website indexed for future retrieval

Problem

1. User has 3 YouTube video URL and 2 Vimeo video URL which describes about devastating Nepal Earthquake 2015.
2. Videos consist of news update & social media updates of earth quake with total running time of **4 hours**.
3. User has to prepare a report about activity of Indian Government's **National Disaster Response Force**.
4. User does not want to spend more than **30 min** of time watching the relevant content.

Solution

Indexing

1. User can open up VideoJs Player With **ShabdoKhoj** Extension
2. User creates a **IndexingQueue** by adding all 5 URL and queue those for indexing.
3. Index the video files from OCR text, Recognized Speech, Metadata etc. and record corresponding time stamp.
4. Create a tag cloud; determine suitable tags with timestamp for the file.
5. Tags with Timestamp files for videos are saved in the file system
6. Let's say, **National Disaster Response Force**, **Indian Government**, and NDRF tags appear in the 15th min, 29th Min and 47th Min of 1st video and so on.

Searching & Retrieval

1. **Tags With Timestamp** file for the Video files are linked in the VideoJs Player UI
2. Now user can search the tags to see when 'NDRF tags are present (e.g.: 15th min). User can fast forward to 15th min and watch the relevant content. User can repeat the same process for remaining 4 videos as well.
3. User can add these video files and their indexing information to **ShabdoKhoj** Library for future use

Video Search within Organization

Problem

1. 2 days remaining for End Semester examination. Total time available for preparation is Maximum 30 hours.
2. 30 x 3 = 90 video lectures available, which will take 90 hours to play.
3. Video titles are corrupted somehow. So it is not possible to identify the important lectures quickly.

Solution

Indexing

1. User can open up VideoJs Player With **ShabdoKhoj** Extension
2. User creates a **IndexingQueue** by adding all 90 videos and queue those for indexing.
3. Index the video files according to Speaker, Creation Time, Capture Location, Recognized Speech, Metadata etc.
4. Index videos by Image object , Image text, Background noise & Acoustic information
5. Create a tag cloud, determine suitable tags with file.
6. Tags with Timestamp files for videos are saved in the file system

Searching & Retrieval

1. **Tags With Timestamp** file for the Video files are linked in the VideoJs Player UI
2. Now User can Open the Tag Browser for videos added in playlist
3. Retrieve According to Speaker : Ksrao, Rsc, Sc
4. Retrieve According to Creation Time & Sort.
5. Retrieve According to Capture Location
6. Retrieve According to Recognized Speech content
7. Retrieve According to User defined metadata / tag
8. Retrieve According to Acoustic information

Complete Web based Video File indexing and retrieval

Problem

1. User has to study around 100 YouTube videos (Running time 50 hours) from Berklee College of Music which describes about Musical activities happening in that college.
2. Videos consist of Musical Collaboration, Fusion music, Jamming sessions & social media updates.
3. User has to analyze Indian fusion Music activity happening in Berklee College of Music so that user can opt to apply in that college.
4. User does not want to spend more than **3 hours** of time watching the relevant content.

Solution

Indexing

1. User can open up VideoJs Player with **ShabdoKhoj** Extension for Web.
2. User creates a **IndexingQueue** by adding Youtube Channel URL and queue it for indexing.
3. Index the video files from OCR text, Recognized Speech, Metadata etc and record corresponding time stamp.
4. Create a tag cloud; determine suitable tags with timestamp for the file.
5. Tags with Timestamp files for videos are saved in the **Centralized Database** hosted in Web
6. Let's say, **Indian Music**, **Indian Classical**, **Fusion** and **A R Rahman** tags appear in 5 videos and so on.

Searching & Retrieval

1. **Tags With Timestamp** file for the Video files are linked in the VideoJs Player UI
2. Now user can search the tags to see when (e.g.: 15th min) and in which video (e.g.: 11th video) '**Indian**' tags are present. User can fast forward to 15th min and watch the relevant content. User can repeat the same process for matching videos as well.
3. User can add these video files and their indexing information to **ShabdoKhoj** Library for future use

Chapter 4:

Implementation Details

Video file access:

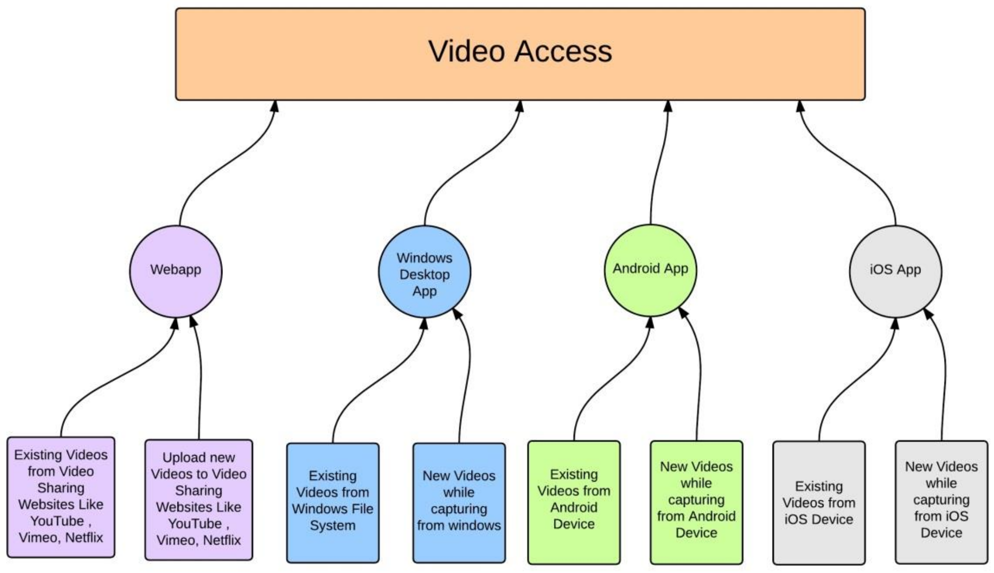

Video Access
Webapp
Windows Desktop App
Android App
iOS App
Existing Videos from Video Sharing Websites Like YouTube , Vimeo, Netflix
Upload new Videos to Video Sharing Websites Like YouTube , Vimeo, Netflix
Existing Videos from Windows File System
New Videos while capturing from windows
Existing Videos from Android Device
New Videos while capturing from Android Device
Existing Videos from iOS Device
New Videos while capturing from iOS Device

Figure 18: ShabdoKhoj Video Access Options

Video Access is the most important aspect for us. We should be able to access existing videos as well as videos while capturing. Most of the videos are captured from mobile devices or accessed from online video sharing websites. I have written apps in four different platforms to enable video indexing better.

Windows App for Video Access

Windows App for Video Access will have 4 ways of adding a new file for indexing. Such as:

- **File Browser** : User can browse the file system to add desired file for indexing
- **File system Watcher:** User can add folders to watch list. Whenever a new file added in the Watch Listed Folders, it will be automatically queued for indexing.
- **File Explorer Menu Integration** : User can right click on a video file and add the file for indexing by clicking on menu item – "Index With ShabdoKhoj"
- **New Video File Capture**: User will get an option for indexing the video file while capturing video with Webcam or attached camera device.

Place Holder for Screenshot of Windows App for Video Access

Android App for Video Access

Video files from android device can be accessed in three ways. Such as:

- Browse existing videos and select for indexing
- Share existing videos and Add to queue for indexing
- New video can be added to queue for indexing while capturing

Place Holder for Screenshot of Android App for Video Access

iOS App for Video Access

Video files from iOS device can be accessed in following ways. Such as:

- Browse existing videos and select for indexing
- New video can be added to queue for indexing while capturing

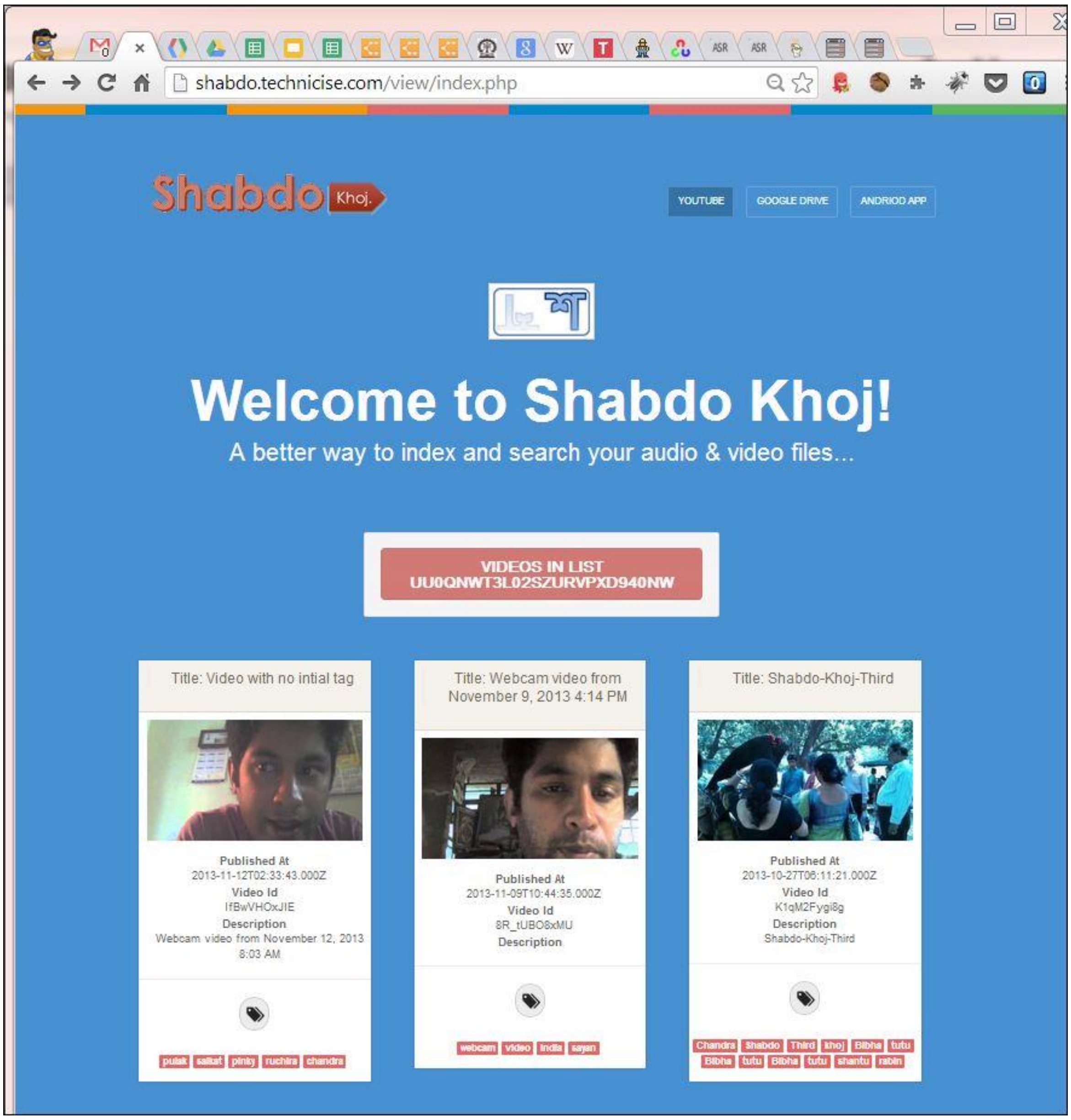
Place Holder for Screenshot of iOS app for Video Access

Web App for Video Access

Video files can be accessed from existing websites like:

1. YouTube
2. Vimeo (Not Implemented)
3. Netflix (Not Implemented)
4. Google Drive (Not Implemented)
5. Drop Box (Not Implemented)

But most of these websites do not allow accessing raw video file using APIs. So we are accessing other relevant metadata. For Youtube Connector, we are accessing sub-titles / captions of the video file along with the timestamp. We can extract Tags with Timestamps from metadata extracted from video files.



Welcome to Shabdo Khoj!

A better way to index and search your audio & video files…

VIDEOS IN LIST
UU0QNWT3L02SZURVPXD940NW

Title: Video with no intial tag

Published At
2013-11-12T02:33:43.000Z
Video Id
IfBwVHOxJIE
Description
Webcam video from November 12, 2013 8:03 AM

pulak saikat pinky ruchira chandra

Title: Webcam video from November 9, 2013 4:14 PM

Published At
2013-11-09T10:44:35.000Z
Video Id
8R_tUBO8xMU
Description

webcam video india sayan

Title: Shabdo-Khoj-Third

Published At
2013-10-27T06:11:21.000Z
Video Id
K1qM2Fygi8g
Description
Shabdo-Khoj-Third

Chandra Shabdo Third khoj Bibha tutu Bibha tutu Bibha tutu shantu rabin

Figure 19: Web App for Accessing YouTube Videos

Web app will enable another option to upload video files from our device specific Uploader. Our device specific apps will perform the audio/video indexing process before uploading to the websites and attach the index keys.

Queue for Indexing

Audio / Video Indexing is a time consuming process. Users can keep on adding files for indexing. So we need to implement queue for files waiting to be indexed.

Indexing Queue for Windows

Indexing Queue for Windows can be implemented using .NET Task Schedulers and ConcurrentQueue. It will be a FIFO queue which lets users to add videos for indexing.

Place Holder for Screenshot of Windows App for Video Indexing Queue

Indexing Queue for Android

Indexing Queue for Android can be implemented using customized Job-Queue classes. It will be a collection of android async tasks performing indexing on video files.

Place Holder for Screenshot of Android App for Video Indexing Queue

Indexing Queue for iOS

Indexing Queue for iOS can be implemented using Grand Central Dispatch (GCD)'s Task Queue. GCD provides and manages FIFO queues to which your application can submit tasks in the form of block objects. We will use Concurrent queues so that tasks are dequeued in FIFO order, but run concurrently and can finish in any order.

Place Holder for Screenshot of iOS App for Video Indexing Queue

Indexing Queue for Web

Indexing Queue for Web can be implemented as node.js based job queue using Kue. *This feature is not implemented yet*.

Audio/Video Indexing Process - Incorporation of Searchable keys

I came up with algorithm & workflow for incorporating Unique & Searchable keys for each audio/Video file which can be a combination of below mentioned metadata.

Audio/Video Indexing metadata content

Audio/Video data is more complex than text data. So we need to rely on metadata for more relevant search results. Here are list of metadata can be considered.

a. Recognized Speech content (*Partially Implemented*)

b. Optical Character recognition (*Implemented*)

c. Timestamp (*Implemented*)

d. Image Object Information(*Partially Implemented*)

e. Existing Video Metadata(Implemented)

f. Background Music & Noise Information

g. Capture device information

h. Capture type information(sampling rates, no of bits)

i. Location information

j. Speaker Information
k. Acoustic information
l. Language information
m. Image information
n. User defined metadata / tag

Indexing using Recognized Speech content

Audio data is extracted from Video files using ffmpeg library in windows, android and iOS platform ports. Here is the generic command for audio extraction.

```
D:\ChandraPersonal\Mtech\shabdokhoj\FFmpeg>
ffmpeg.exe -i source-video.mp4 -ar 16000 -vn retrieved-audio.wav
```

We had to adjust input parameters according to source video file parameters.

Then CMU Pocket Sphinx library is used to determine speech content from the audio file in windows, android and iOS. Pocket Sphinx library is depended on Sphinx Base library. I have configured CMU Sphinx with English language model. I need to configure this with Indian English as well to get better result.

Here is the generic command for Speech to Text conversion.

```
D:\ChandraPersonal\Mtech\shabdokhoj\PocketSphinx>
bin/Release/pocketsphinx_continuous.exe -infile  retrieved-audio.wav -hmm model/en-
us/en-us -lm model/en-us/en-us.lm.dmp -dict model/en-us/cmudict-en-us.dict >
recognized-text.txt
```

Then I am running Tag Detection Algorithm on recognized speech content to determine Tags with Timestamp.

Indexing using Optical Character Recognized content

Video Frames are extracted from Video files using ffmpeg library in windows, android and iOS platform ports. Here is the generic command for Video Frame extraction as image.

```
D:\ChandraPersonal\Mtech\shabdokhoj\FFmpeg>
ffmpeg -i source-video.mp4 -r 1 -s 720x480 -f image2 foo-%03d.jpeg
```

I had to adjust image resolution input parameter depending on the video resolution. Now I have video frame as image for each second of video duration.

Then I am using Tesseract OCR library (windows, android, iOS) to recognize text from

each image and save them as text file.
Here is the generic command for OCR text extraction from image.

```
D:\ChandraPersonal\Mtech\shabdokhoj\tesseract>
"D:\Program Files (x86)\Tesseract-OCR\tesseract.exe" t.JPG out
```

Now we have recognized text files for every video frames. Then we are using PERL script to parse & merge all text files in to single Tags with Timestamp file

Indexing using Recognized Object content

Video Frames are extracted from Video files using ffmpeg library in windows, android and iOS platform ports. Here is the generic command for Video Frame extraction as image.

```
D:\ChandraPersonal\Mtech\shabdokhoj\FFmpeg>
ffmpeg -i source-video.mp4 -r 1 -s 720x480 -f image2 foo-%03d.jpeg
```

I had to adjust image resolution input parameter depending on the video resolution. Now I have video frame as image for each second of video duration.

Now I am using OpenCV library to recognize face, objects from the image files in windows, android and iOS. All the relevant information along with the timestamp is stored in to single Tags with Timestamp file

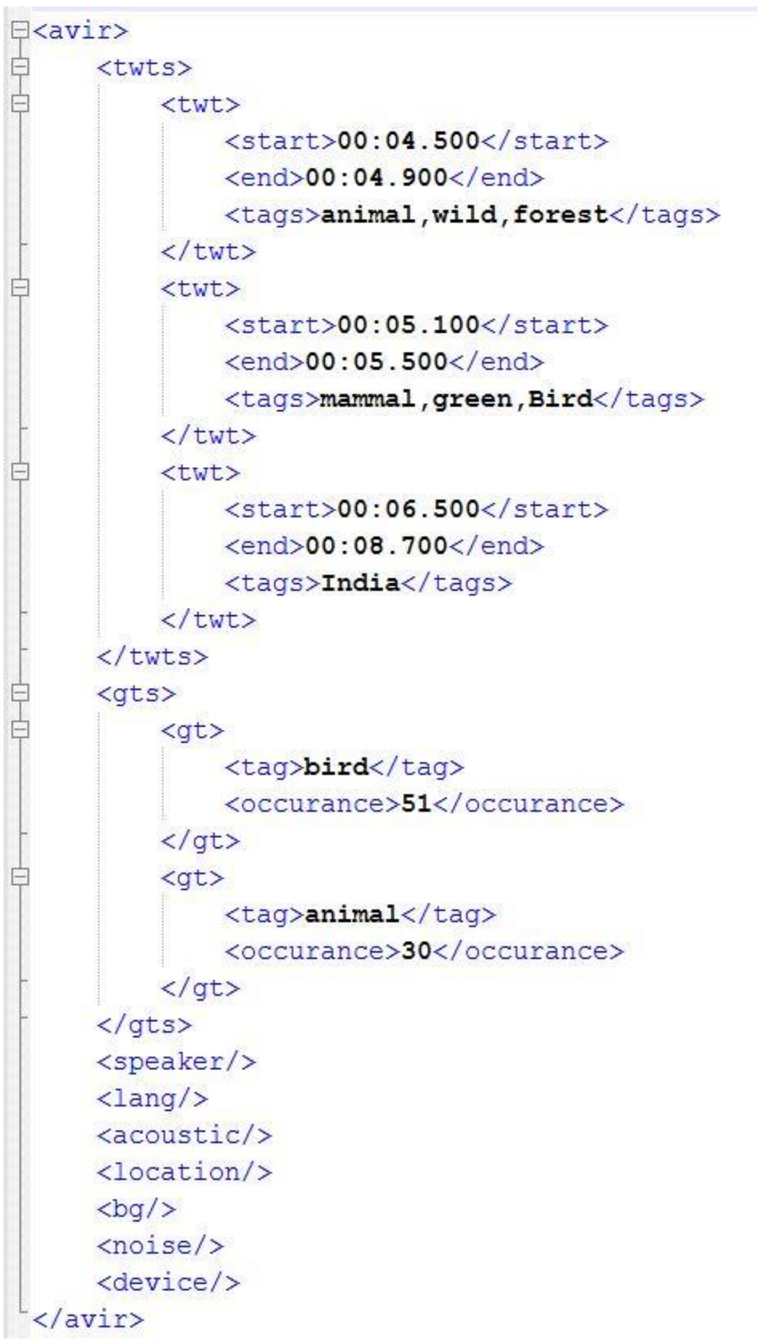
Indexing keys / Tags Storage

Derived index keys can be tied to individual files or stored as a collection.

Individual File Specific

Tags with Timestamp as text file for each video/audio file

Below is an example of Tags with Timestamp file. It is a text file can be represented as xml or json format.



```xml
<avir>
    <twts>
        <twt>
            <start>00:04.500</start>
            <end>00:04.900</end>
            <tags>animal,wild,forest</tags>
        </twt>
        <twt>
            <start>00:05.100</start>
            <end>00:05.500</end>
            <tags>mammal,green,Bird</tags>
        </twt>
        <twt>
            <start>00:06.500</start>
            <end>00:08.700</end>
            <tags>India</tags>
        </twt>
    </twts>
    <gts>
        <gt>
            <tag>bird</tag>
            <occurance>51</occurance>
        </gt>
        <gt>
            <tag>animal</tag>
            <occurance>30</occurance>
        </gt>
    </gts>
    <speaker/>
    <lang/>
    <acoustic/>
    <location/>
    <bg/>
    <noise/>
    <device/>
</avir>
```

Pushed to Host Website's (Youtube) tags

Our Web app can insert tags in to existing Youtube Video. It can be manually indexed tags or automatically derived tags. This is implemented using Youtube developers V3 API.

Collection of Files

Centralized DB for Tags with Timestamp

Centralized DB for tags can be treated as user's media library. We are allowing user to add video/audio into his Indexing Queue. After completion of indexing, Tags with timestamps for each file are stored in database. We are using following database tables.

- FileInfo
  - File_id
  - File_Name
  - File_Path

- IndexInfo
  - File_id (foreign key)
  - Tags_with_timestamp_xml

Audio Retrieval

Audio Retrieval can be done following ways. This includes legacy way of audio/video retrieval as well as new timestamp based retrieval.

Legacy retrieval with generic index keys

Our indexing mechanism can come up most relevant & appropriate tags for indexed audio/video file. We are inserting these new tags/index keys in the video files for enabling better retrieval chances. User can use operating system search or Website search with the desired keys to find out the videos.

Tag Browser of Centralized DB for tags

Centralized DB or Media Library for Tags with Timestamps can hold enough information about the indexed video files to represent them in a formatted way. Current implementation of tag browser displays videos along with its tags in a Grid view or list view.

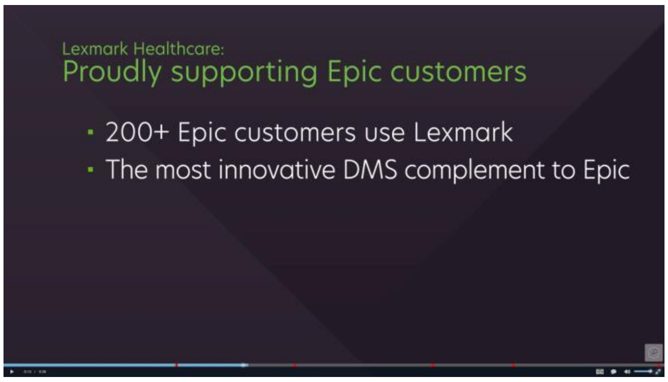
Place Holder for Screenshot of Tag Browser

Timestamp based video frame retrieval

Tags with Timestamps for a video enable us better retrieval mechanism for video frames. But Traditional video player cannot utilize this feature. So I have extended videojs (The open source HTML5 video player) to support Tags with Timestamps for a video. ShaboKhoj Extension for VideoJs has following new features:

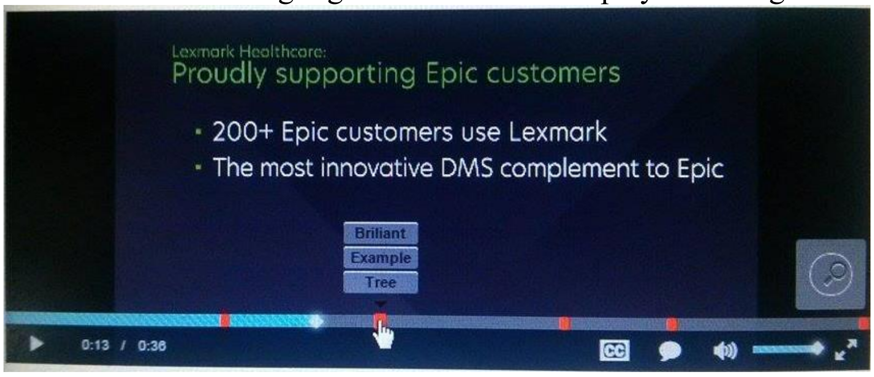
- Parse Tags with Timestamps for video and associate tags in the video timeline.

- Timestamp which has corresponding tags will marked in highlighted color in video progress bar.



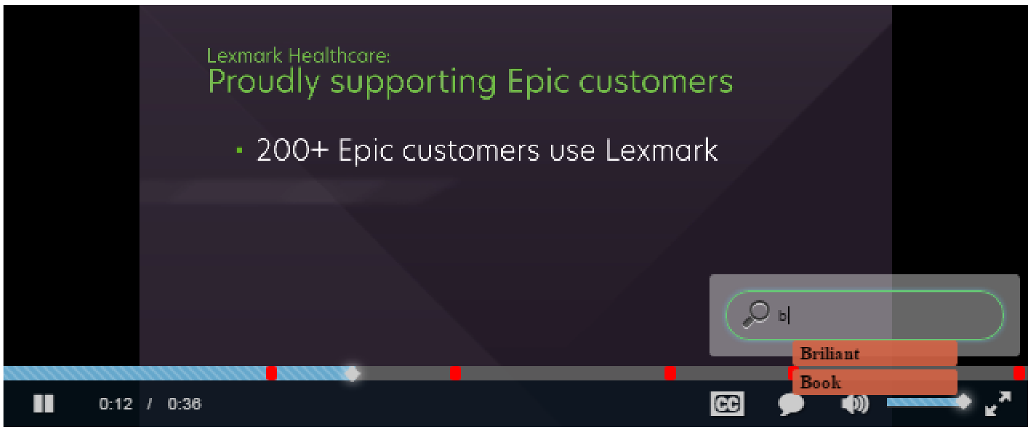
Lexmark Healthcare:
Proudly supporting Epic customers
- 200+ Epic customers use Lexmark
- The most innovative DMS complement to Epic

- On hover action on the highlighted marker will display list of tags.



Lexmark Healthcare:
Proudly supporting Epic customers
- 200+ Epic customers use Lexmark
- The most innovative DMS complement to Epic

- Search tags and navigate to selected tag's video frame



Lexmark Healthcare:
Proudly supporting Epic customers
- 200+ Epic customers use Lexmark

End to End Flows Implemented

Web Video Access & Manual Indexing Prototype (Flow 1)

I have created a working demo of "Audio Indexing and Retrieval project" in http://shabdo.technicise.com/php/ . It is a Google App for YouTube. You can login using your Google id. It will list down all YouTube videos uploaded by you and will display relevant information about the videos. You can add tags depending on the audio contents.

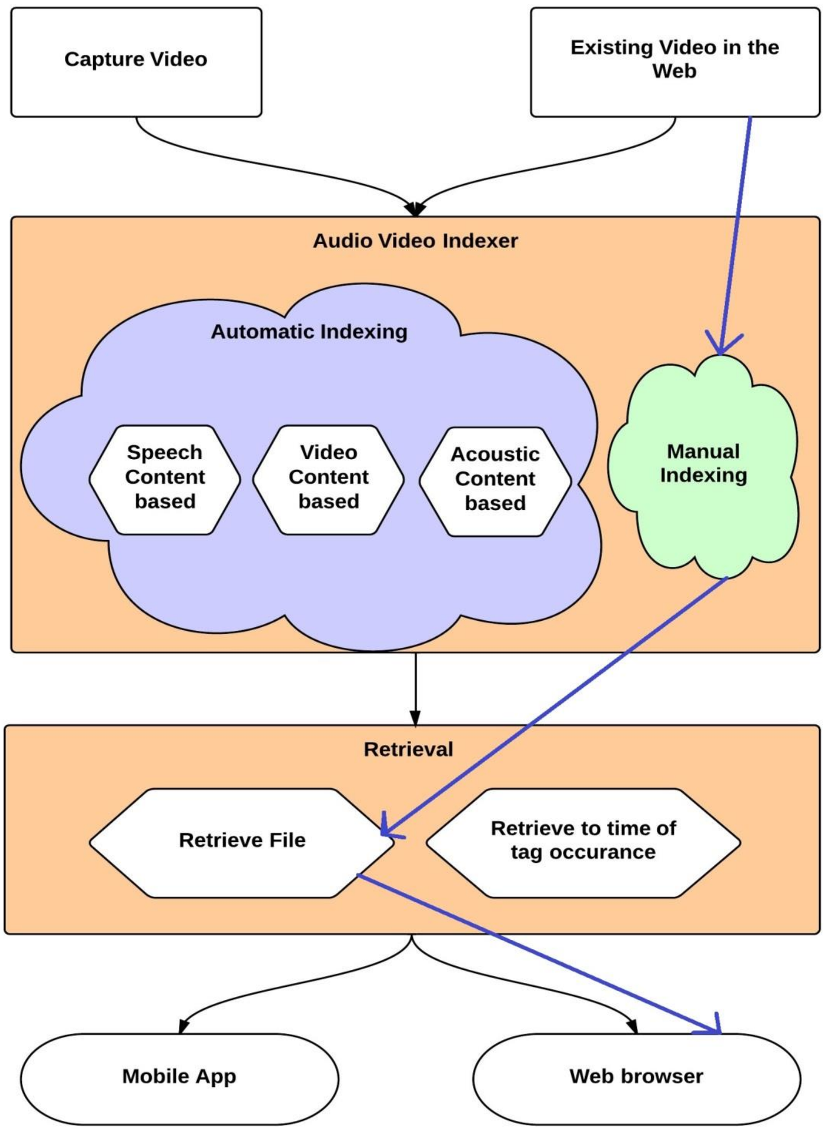


Figure 20: End to end flow implemented in 5th Sem

Implemented flow is marked by violet arrows.

Web Video Access & Automatic Indexing Prototype (Flow 2)

Developed a YouTube App → Authenticate using Google Open Id → Access List of videos available in user's YouTube Account → Access Video information like Tags, Video Id, Thumbnail, Upload time, Title, Description, Captions, Sub-Titles → Parse Caption/Sub-title file to find out relevant tags →Automatically Add tags to the existing YouTube video.

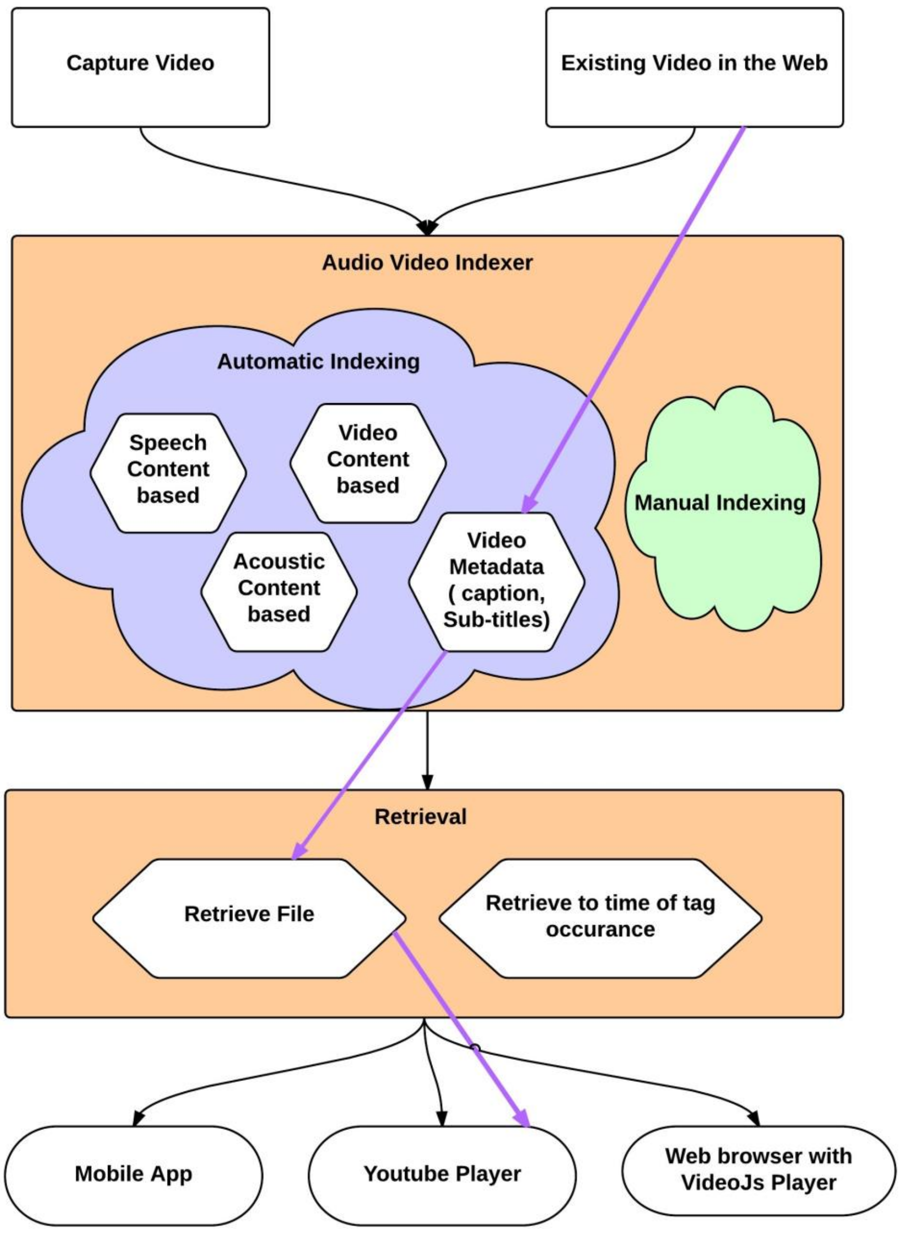


Figure 21: End to end flow implemented with Automatic Indexing of Web videos

Implemented flow is marked by violet arrows.

Windows File System Video & Automatic Indexing of Speech Content (Flow 3)

Access video File from Windows File System → Extract Audio Data using ffmpeg → Convert Speech To text using CMU Pocket Sphinx → Parse Recognized text to find out relevant tags with timestamp→Add tags to the TWT file of the video → Play Using VideoJs Player to retrieve to Timestamp for a specific tag.

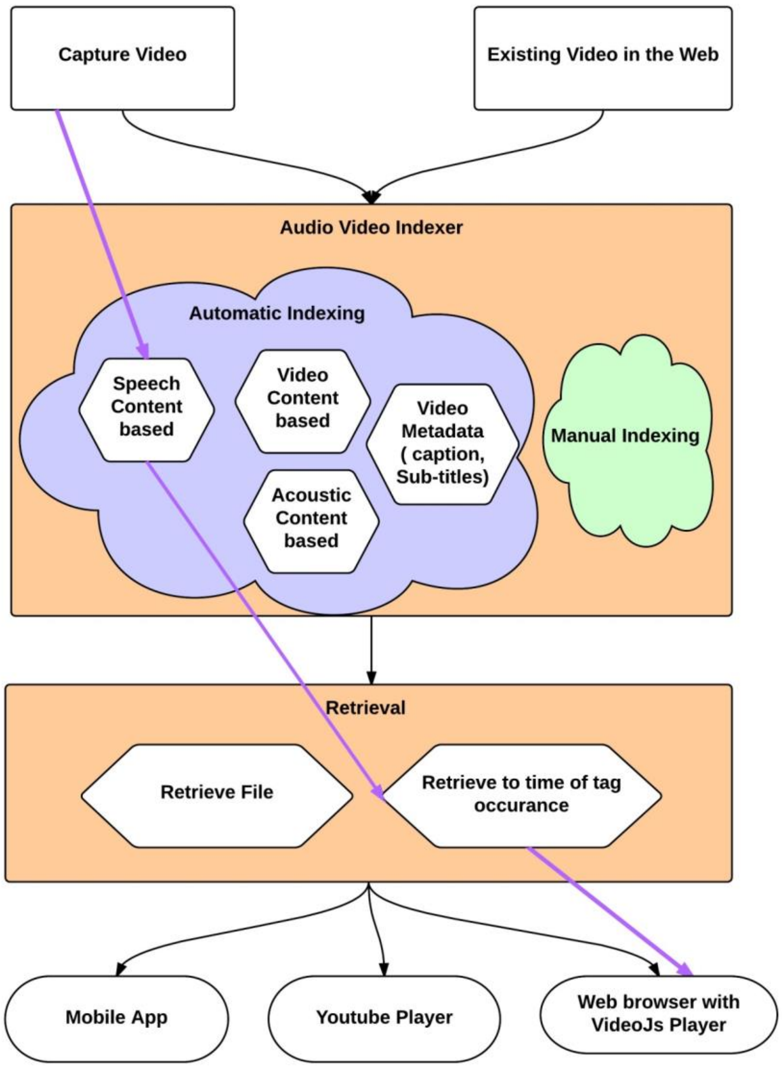


Figure 22: End to end flow implemented with Automatic Indexing of Web videos

Implemented flow is marked by violet arrows.

Windows File System Video & Automatic Indexing of OCR Content (Flow 4)

Access video File from Windows File System → Extract Video Frames using ffmpeg → Extract OCR text using Tesseract → Parse Recognized text to find out relevant tags with timestamp→Add tags to the TWT file of the video → Play Using VideoJs Player to retrieve to Timestamp for a specific tag.

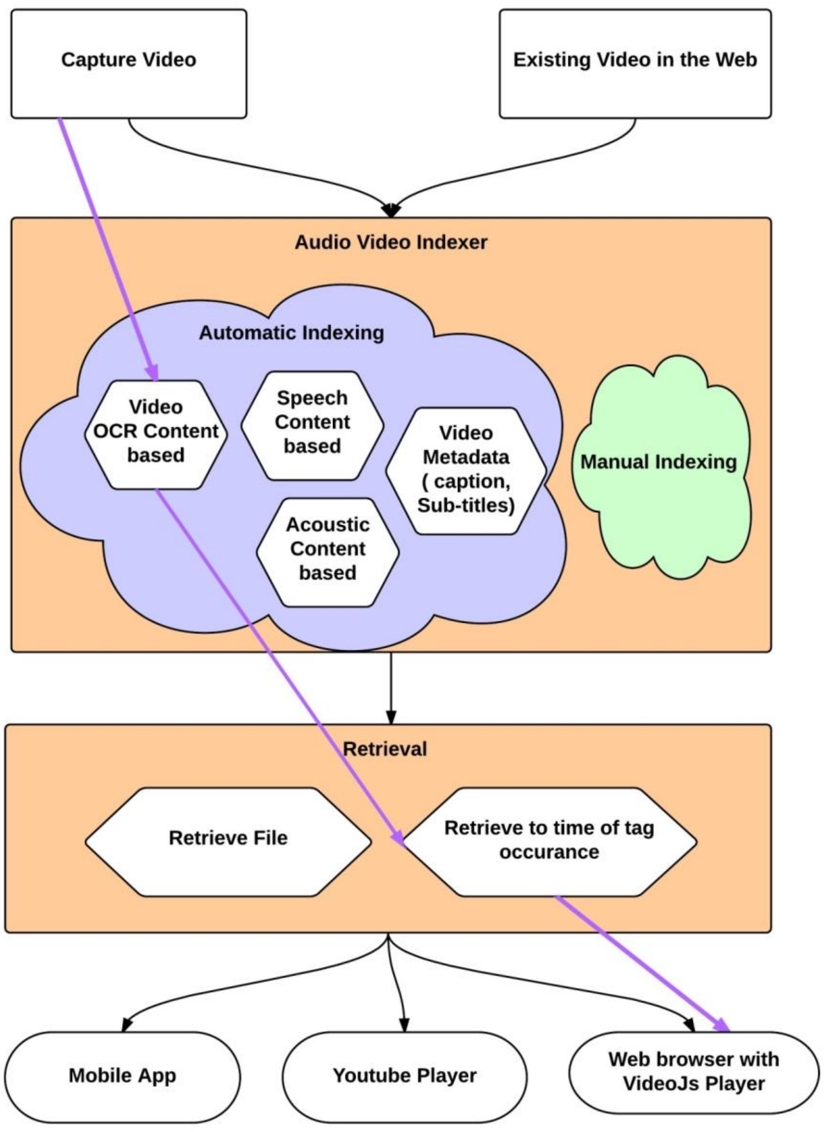


Figure 23: End to end flow implemented with Automatic Indexing of Web videos

Implemented flow is marked by violet arrows.

Windows Filesystem Video&Automatic Indexing based on Video Object(Flow 5)

Access video File from Windows File System → Extract Video Frames using ffmpeg → Detect Objects / Faces using OpenCV→ Parse Recognized objects to find out relevant tags with timestamp→Add tags to the TWT file of the video → Play Using VideoJs Player to retrieve to Timestamp for a specific tag.

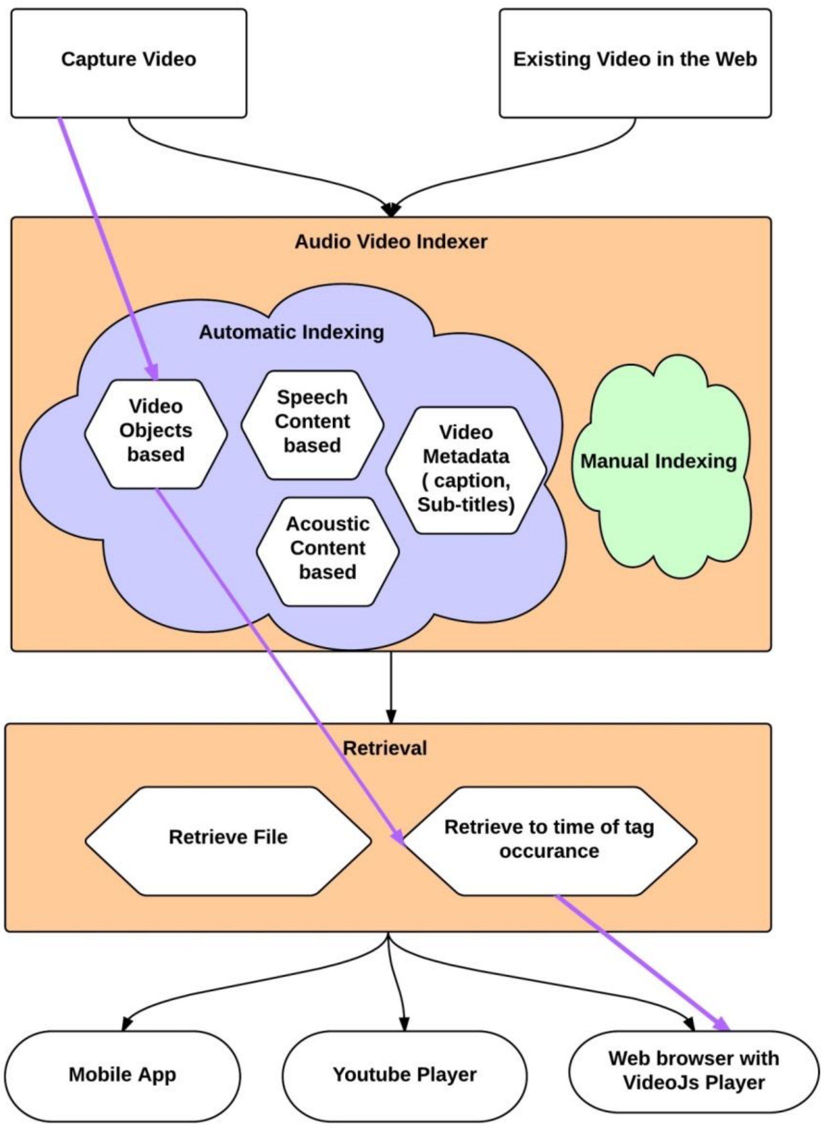


Figure 24: End to end flow implemented with Automatic Indexing of Web videos

Implemented flow is marked by violet arrows.

Performance & Accuracy Matrix

Indexing & Retrieval Mechanism	Accuracy	Performance (Processing Time)	Comments
OCR Text Based Indexing	Accurate	1/50th of running time	
Video metadata (Captions , Subtitles, EXIF) Based Indexing	Accurate	1/100th of running time	
Speech Content based Indexing	40 % accurate	1/30th of running time	Need to model for Indian English and other languages.
Video Object Content based Indexing	30 % accurate	1/40th of running time	Add more Training Data

Future Scope

1. Integration With Vimeo & other video sharing website (Netflix , ustream, box.com, Dropbox , Google Drive)
2. Gamification of Video Indexing while video capture by mobile devices
3. Search Engine Optimization of Indexed Video/Audio files
4. Exclusive Indexing and tagging of Music Files
5. Integration of Tags file with EXIF MIE metadata
6. Integration of Tags file with Operating System Search Indexes

Conclusions

In this approach, we are extracting information about the video and audio content of the file from various aspects. Then we are determining most suitable tags and attaching them with file so that we can search & retrieve this file based on those tags. This initiative will provide better searching options for video and audio file. Implementation of this approach will provide an extendible framework for Audio, Video Indexing which can be enhanced & customized by developers and users respectively.

References

[1] http://allthingsd.com/20110601/cisco-the-internet-is-like-really-big-and-getting-bigger/

[2] http://www.omg-facts.com/Technology/In-2010-Google-Had-Only-Indexed-004-Of-T/52586

[3] GUOJUN LU , “Indexing and Retrieval of Audio: A Survey”

[4] Christopher Kuner, Fred H. Cate, Christopher Millard and Dan Jerker B. Svantesson , “The challenge of ‘big data’ for data protection”, http://idpl.oxfordjournals.org/content/2/2/47.extract

[5]http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html

[6] Websites:

- http://en.wikipedia.org/wiki/Query_by_humming
- http://www.midomi.com/
- http://www.soundhound.com/
- http://en.wikipedia.org/wiki/Tunebot
- http://en.wikipedia.org/wiki/Musipedia
- http://www.kecl.ntt.co.jp/csl/sirg/people/yasushi/SoundCompass.pdf
- http://en.wikipedia.org/wiki/Parsons_code
- http://music.cs.northwestern.edu/index.php
- http://tunebot.cs.northwestern.edu/index.php
- http://www.musipedia.org/
- http://en.wikipedia.org/wiki/Acoustic_fingerprint
- http://en.wikipedia.org/wiki/Shazam_(service)
- http://www.speech.kth.se/wavesurfer/
- http://audacity.sourceforge.net/
- http://www.hongkiat.com/blog/25-free-digital-audio-editors/
- https://incus.greenbutton.com/
- http://maart.sourceforge.net/
- https://sourceforge.net/projects/camel-framework/
- https://sourceforge.net/projects/jaudio/
- http://www.speech.kth.se/snack/
- http://research.microsoft.com/en-us/projects/mavis/
- http://googlesystem.blogspot.in/2008/09/google-audio-indexing.html

- http://en.wikipedia.org/wiki/Google_Audio_Indexing
- http://googleblog.blogspot.in/2008/09/google-audio-indexing-now-on-google.html
- http://www.autonomy.com/content/Technology/evolution/evolution-of-search/index.en.html
- http://techcrunch.com/2008/09/17/google-launches-audio-indexing/
- http://en.wikipedia.org/wiki/Multimedia_search
- http://en.wikipedia.org/wiki/Audio_search_engine
- http://www.playaudiovideo.com/pav_start.htm
- http://www.ramp.com/
- http://en.wikipedia.org/wiki/Google_Voice_Search
- http://music.cs.northwestern.edu/index.php
- http://www.aurix.com/pages/3417/Technology.htm
- http://www.apple.com/ios/siri/
- http://en.wikipedia.org/wiki/Siri_(software)
- http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html
- http://mashable.com/2012/03/06/one-day-internet-data-traffic/
- http://hadoop-karma.blogspot.in/2010/03/how-much-data-is-generated-on-internet.html
- http://www.scientificpsychic.com/workbook/chapter2.htm
- http://www.omg-facts.com/Technology/In-2010-Google-Had-Only-Indexed-004-Of-T/52586
- http://googleblog.blogspot.in/2008/07/we-knew-web-was-big.html
- http://idpl.oxfordjournals.org/content/2/2/47.extract
- http://www.internetworldstats.com/stats7.htm
- http://www.fiercewireless.com/story/idc-future-mobile-device-user-input/2010-05-03
- http://www.audiblemagic.com/
- http://www.neurostechnology.com/
- http://www.linkedin.com/company/musiwave
- http://www.crunchbase.com/company/musiwave
- http://mashable.com/2007/11/15/musiwave-microsoft-deal/
- http://www.microsoft.com/en-us/news/press/2007/nov07/11-12ProjectTunesPR.aspx

- http://www.lumenvox.com/resources/tips/historyOfSpeechRecognition.aspx
- http://www.pcworld.com/article/243060/speech_recognition_through_the_decades_how_we_ended_up_with_siri.html?page=2
- https://www.dropbox.com/developers/chooser
- http://www.dropboxwiki.com/Main_Page
- http://support.google.com/drive/bin/answer.py?hl=en&answer=2500820
- http://infolab.stanford.edu/~backrub/google.html
- http://www.google.co.in/intl/en/insidesearch/howsearchworks/thestory/
- http://www.google.co.in/intl/en/insidesearch/howsearchworks/algorithms.html
- http://www.wired.com/insights/2013/10/google-hummingbird-where-no-search-has-gone-before/
- http://royal.pingdom.com/2013/01/16/internet-2012-in-numbers/
- http://variety.com/2014/digital/news/netflix-streaming-eats-up-35-of-downstream-internet-bandwidth-usage-study-1201360914/
- http://www.cs.tut.fi/~moncef/publications/c-b-multimedia-indexing-paris.pdf
- http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.208.1129&rep=rep1&type=pdf
- https://www.fxpal.com/publications/talkminer-a-lecture-webcast-search-engine.pdf
- ftp://ftp.hpl.hp.com/pub/dec/CRL/publications/jmvt/speechbotRIAO2000.pdf
- https://www.tineye.com
- https://www.bing.com/images/
- https://images.google.com/
- http://www.technorms.com/39812/bings-image-match-feature-lets-reverse-search-images