International Journal of Computational Intelligence Research
ISSN 0973-1873 Volume 12, Number 1 (2016), pp. 63-68
© Research India Publications
http://www.ripublication.com

# Comprehensive Tool for Generation and Compatibility Management of Subtitles for English Language Videos

**Ketan Kulkarni, Alka Londhe, Bhushan Mahajan, Chinmay Inamdar, Akshay Jakhotiya**

*Department of Computer Engineering,*
*Pimpri Chinchwad College of Engineering, Pune, India,*
*E-mail: ketankulkarni59@gmail.com, alka.londhe.pccoe@gmail.com*
*bhushanmahajan94@gmail.com, inamdarchinmay26894@gmail.com*
*akshayj001@gmail.com*

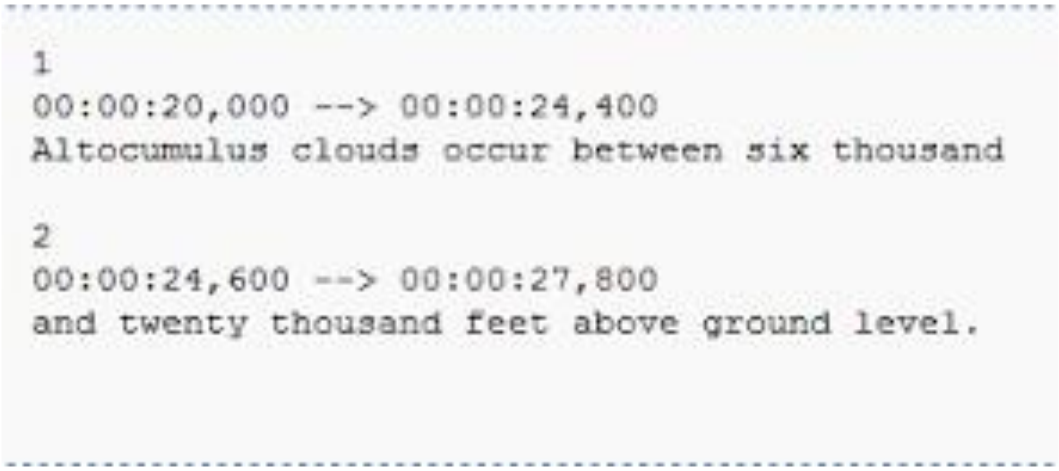
## Abstract

Video is the most powerful medium in the propagation of information and the use of subtitles in representing the textual transcript of dialogues of a video is a customary routine. Though it being a very effective means of communication, has still some leftover challenges. The deaf and hearing impaired people & the non-native language speakers make use of subtitles to read and understand the dialogue. There are various types of software designed to generate the subtitles manually, but the dearth of those automating the process, is often felt. The intended software system will produce automatic subtitles via a three staged process: Audio extraction, Speech Recognition and Synchronization of subtitles.

The realm of this paper is to portray the stage wise mechanism of subtitle generation and describing of various other features provided by the software.

## I. INTRODUCTION

The subtitles are the text translation appearing on the screen of a video displayed in real time during its playback. The users of the subtitle are the people unfamiliar with the language, deaf and those trying to improve reading skills of the language. Depending as per the need, the language of the subtitle could be same as that of the video, or the another one suitable to the user. A subtitle file facilitates the display of subtitles during the playback of video. Below is the structure of a subtitle file in a commonly used format named srt[3].



```
1
00:00:20,000 --> 00:00:24,400
Altocumulus clouds occur between six thousand

2
00:00:24,600 --> 00:00:27,800
and twenty thousand feet above ground level.
```

**Fig 1:** Structure of a Subtitle File

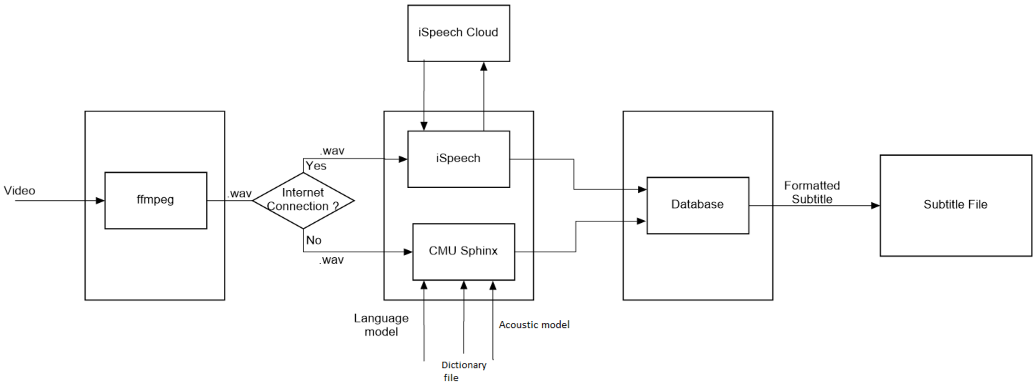
1. Subtitle number-A number to uniquely identify the subtitle in the sequence.
2. Subtitle display time slots –The time instant for which a subtitle appears on the screen. After this time slot the subtitle will disappear from the screen.
3. Subtitle text –This field represents the subtitle.
4. A blank line separating two subtitles and indicates the start of new subtitle [1].

The system being developed would automate the process of subtitle generation by making use of speech recognition engine. It uses two speech recognition engines namely iSpeech and CMU Sphinx, where the former being an online service would offer better accuracy and performance over to the latter which is an offline engine. The feature of the system is to recognize automatically, the availability of network and preferably to work online, as the effective output in terms of accuracy and the time efficiency are the primary objectives. The other major feature of the software which is worth mentioning is, its ability to burn the subtitles into a video file, so that they become part of a video rather than being fetched from an external file by the video player. To make the tool more convenient to the user, the feature of inter-convertibility among various subtitle file formats as per the requirement is incorporated. This system also provides, the scope and the accessibility for the developer to edit the defective portion of the automatically generated subtitles, manually.

Thus the system being developed is advantageous over the earlier generation non automated system, which uses subtitle editing software like ‘Gaopol’ or ‘GNOME’[4][5][6]. In a non-automated system for a given subtitle entry, the start time instant and end time instant are selected and the subtitle is typed manually. The word ‘subtitle entry’ here denotes individual units containing one subtitle number, subtitle display slot and subtitle text.

## II. MECHANISM AND USER INTERFACE

### Architecture Diagram



### A. *Subtitle Creation*

The mechanism of automatic subtitle generation is the process comprising of three stages:

i) Audio Extraction
ii) Speech Recognition
iii) Subtitle Synchronization

### *i) Audio Extraction:*

The input of speech recognition engine is a.wav file format. ffmpeg is used to convert the video into.wav file. It is a complete, cross-platform tool used for inter-conversion of audio and video [7]. It extracts an audio slice from the video after start and end time of the slot is specified.

The video for which the subtitles are to be created is played into video panel. The preview of the portion of video for which subtitles are to be generated is shown by this panel. The panel provides all types of video control functions.

Start and Stop buttons facilitates the user to specify the time slot for a subtitle entry. The user will click on pause button to stop the video and click on start button to record the starting instance of a subtitle. The same procedure is followed to obtain the end time of subtitle. After the specification of time slot, user will click on OK button to get the speech slice converted to wav format.

### *ii) Speech Recognition*

The output of audio extraction process is a wav file, which is forwarded further for speech recognition. The process of speech recognition requires a considerable amount of computational power, and to achieve it on a mediocre type of computer machine is very difficult. So, the system will be incorporated with two speech recognition engines namely iSpeech and CMU sphinx[2].

iSpeech-This is a cloud-based, service providing online speech recognition engine, which gets input from a local computer using internet, to transmit audio file. The

textual transcript of the transmitted audio file is the output of this engine. The textual transcript in turn forms a part of subtitle entry to finally become the desired subtitle.
CMU Sphinx-This is an offline service providing speech recognition engine. All its components are present locally. It gets input in the form of a audio file. The textual transcript of the audio file is the output of CMU Sphinx. CMU sphinx project is developed under the umbrella of Carnegie Mellon University, Sun Microsystems Inc. and Mitsubishi Electric Research Laboratories. Sphinx-4 is a Java wrapper, which interfaces with all the components of CMU Sphinx[9]. The three major dependencies of CMU Sphinx are:

1) Acoustic Model
2) Language Model
3) Dictionary file

### 1) Acoustic Model

Linguistic units like phonemes, which is the output of the acoustic model, makes up the speech. The acoustic model is used for representing the relation between audio signals and phonemes. To train the acoustic model, it is provided with audio recordings and its corresponding transcript [10].

### 2) Language Model

The words and phrases some times sound similar, but mean entirely different, the context is provided by the language model to distinguish. The grammatical accuracy of the recognized sentences is dependent upon the language model. It matches audio with word sequences, by deciding upon the words which could follow the previously recognized words, and restricts the matching process, making use of probability distribution assigned to the words [10].

### 3) Dictionary file

The dictionary file contains a list of words in a language and their corresponding phonemes. The phonetic information generated by the acoustic model, as its output, is mapped with phonemes in the dictionary, and the appropriate word matching to the phoneme is recognized and provided out.
The speech recognition engine outputs its result into the text-area. As the accuracy of the speech recognition engine cannot be relied upon, it facilitates the user to modify the text output if there are any errors. The text area will also allow the user to format the subtitles. The formatting involves the modification of text size, color and font type [3]. The software allows user to export the subtitles into 15 different file formats, where each has its own formatting style. So, user has to specify the intended output file format before starting the formatting of subtitles. The formatting helps emphasizing the emotional variations in the dialogue. After, the completion of formatting, user clicks OK button to add the subtitle to the table. There is a possibility that user wish to discontinue for some time and to keep saved project intermediately and resume it later. So, the object of table is serialized and saved in a file. The saved file can then be opened to resume the incomplete project. The file containing the serialized object of table is encrypted with the DES algorithm.

To allow user to modify or delete the previously generated subtitles, the GUI is also incorporated with the history panel to ease the navigation.

***iii) Subtitle Synchronization***

This is the ultimate phase wherein the subtitle file for the recognized subtitles is generated. The user specifies the intended output format of the subtitle file. The subtitle would then be written as per the style of destination file format, after being fetched from the table, containing the generated subtitles, in a generalized format.

**Table I.** format of subtitle table

| **START TIME IN MILLISEC** | **END TIME IN MILLISEC** | **START TIME** | **END TIME** | **SUBTITLE** |
|---|---|---|---|---|
| 40417 | 41980 | 00:00:40, 417 | 00:00:41, 980 | Is it true?? |
| 48746 | 49676 | 00:00:48, 746 | 00:00:49, 676 | It can be disastrous for us. |
| 51744 | 53639 | 00:00:51, 744 | 00:00:53, 639 | When is it going to end? |
| 63443 | 65170 | 00:01:03, 443 | 00:01:05, 170 | I will try to get out of it, as early as possible. |

### *B. Subtitle file conversion*

The compatibility of a subtitle file on different video players has always been a very subtle issue, and it is often observed that a subtitle file compatible on one video player, is not supported by another. The incorporation of this feature aims at providing inter-convertibility among various subtitle file formats. The user specifies the source format, output format and inputs the file to be converted. The file is then converted to output format, after the user clicks on the convert button. A generalized table, with a structure similar to the one mentioned above, acts as a mediator, in facilitating the conversion. The subtitles and their associated information is parsed from the input file, and stored into the table. As every file format has different style of representing the parameters of a subtitle entry, the various Java methods, each dedicated to a particular file format have been designed, which are responsible for parsing the information from input file to the table, and printing the output file using the information present in the table.

### C. Subtitle hardcoding into video file

The conventional method for displaying the subtitles during the video playback is through an external subtitle file**,** placed alongside the video file. The video player performs the real time fetching of subtitles from the file. However, video player on various devices, other than traditional computers, does not support this feature, thus minimizes its compatibility. So, to overcome this shortcoming, the software has been subsumed, with the feature of burning the subtitles into a video file. The subtitles are hard scribed into the frames of video, so that they become the part of video file itself. ffmpeg [7][8] is used to accomplish this feature. ffmpeg takes video and subtitle file as input and fabricates the new output video file with subtitles burned in it. However, the subtitle file formats supported by the ffmpeg are.srt and.ass.

So, by making use of format conversion, the inputted subtitle file of any format can be converted to srt format, before passing it onto ffmpeg.

## CONCLUSION

We have developed this system to automate the process of subtitle generation and tried to encourage its use through the introduction of various innovative features like subtitle file conversion and hardcoding of subtitles, leading to increase in compatibility. However, improving the accuracy of the speech recognition engine to reach near to the same of manual typing is a major challenge, along with the other challenges like difficulties in getting punctuation marks in the text, also still does persist.

## REFERENCES

[1] J.O. Djan and R. Shipsey, "E-Subtitles: Emotional Subtitles as a Technology to assist the Deaf and Hearing-Impaired when Learning from Television and Film", in Sixth International Conference on Advanced Learning Technologies, pp.464-466, 2006

[2] A. Mathur, T. Saxena, R. Krishnamurthi, "Generating Subtitles Automatically using Audio Extraction and Speech Recognition", 2015 IEEE International Conference on Computational Intelligence & Communication Technology, pp.621-626

[3] http://www.matroska.org/technical/specs/subtitles/srt.html, accessed April 2016

[4] http://gnome-subtitles.sourceforge.net, accessed April 2016

[5] http://home.gna.org/subtitleeditor

[6] http://home.gna.org/gaupol, accessed April 2016

[7] https://www.ffmpeg.org, accessed April 2016

[8] https://trac.ffmpeg.org/wiki/Projects, accessed April 2016

[9] http://cmusphinx.sourceforge.net/doc/sphinx4

[10] http://sourceforge.net/projects/cmusphinx/files/Acoustic%20and%20Language%20Models/, accessed April 2016