Journal of International Technology and Information Management

Volume 20
Issue 1 *Double Issue 1/2*

Article 4

2011

Speech Corpus Generation from DVDs of Movies and TV Series

Veton Z. Kepuska
Florida Institute of Technology

Pattarapong Rojanasthien
Florida Institute of Technology

Recommended Citation

Kepuska, Veton Z. and Rojanasthien, Pattarapong (2011) "Speech Corpus Generation from DVDs of Movies and TV Series," *Journal of International Technology and Information Management*: Vol. 20: Iss. 1, Article 4.
DOI: https://doi.org/10.58729/1941-6679.1100
Available at: https://scholarworks.lib.csusb.edu/jitim/vol20/iss1/4

Speech Corpus Generation from DVDs of Movies and TV Series

Veton Z. Këpuska
Pattarapong Rojanasthien
Florida Institute of Technology
USA

ABSTRACT

Speech corpus is a database of audio files containing spoken words/sentences and text transcriptions. In this work we present a data collection system for creating speech corpora from movies and TV series DVDs. Corpus generation from these DVDs is significantly lower-cost solution comparing to conventional way of obtaining a speech corpus. In addition, it also takes a shorter amount of time to collect the data and processes it into a corpus. In order to be able to perform this operation the ***Data Collection Toolkit*** *is introduced. This toolkit is an application developed using C# .Net Framework 3.5 in Visual Studio 2008. Throughout the presented work, this toolkit is included to show how it can be utilized to simplify the process of creating a corpus.*

INTRODUCTION

A term speech corpus refers to a database of speech data including audio files and corresponding text transcriptions. The idea of corpus generation from DVDs of movies and TV series is inspired from the study of prosodic analysis of Wake-Up-Word technology (Këpuska & Klein, 2009; Këpuska & Shih, 2010; Këpuska, Gurbuz, Rodriguez, Fiore, Carstens, Converse, & Metcalf, 2008). Their studies showed that the prosodic features extracted from pitch of WUW-II speech corpus did not yield the expected result. That was due to the nature of the corpus they used; it was created by having people read transcripts and recorded via telephone (landline, speaker-phone, mobile, etc.), which made the corpus less natural than actual speech obtained from conversations between people (Këpuska & Shih, 2010).

Obtaining speech corpora by hiring a professional company or buying (e.g., linguistic consortium: http://www.ldc.upenn.edu/) would cost a significant amount of money as shown in Figure 1 (LDC, University of Pennsylvania, 2009). Creating a corpus from recording a conversation (Aiken, 2009) and then writing corresponding text transcriptions would also be very time consuming and are dependent on the quality of recordings (Musa, 2010). These two methods are typically not feasible for those who have a low research budget and limited amount of time to work on the data collection. That is why using corpus generation from DVDs of movies and TV series is a better option in obtaining speech corpora. Not only the utterances from the corpora are natural speech, it also costs nothing to create a large set of speech corpora assuming the DVDs have already been bought.

Although most of the DVDs of movies and TV series are likely to be protected by copyright-law (Lippert, 2007), this research should not violate this law. That is, 1) there is no public distribution of the viewing contents from the DVDs, and 2) the corpora generated from the DVDs will be used as an "in-house" research for the speech recognition system only (Hemming & Lassi, 2010). The proposed approach should not affect the copyright holders to suffer a loss

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

of profit from selling their DVDs. On the contrary, this corpus generation should increase the sales since more DVDs will be bought as there is a need in creating corpora from them (see Table 1).

Table 1: List of top ten corpora from linguistic data consortium and their costs.

Corpora	Cost
TIMIT Acoustic-Phonetic Continuous Speech Corpus	$250.00
Web 1T 5-gram Version 1	$150.00
CELEX2	$300.00
TIDIGITS	$500.00
ECI Multilingual Text	$75.00
Treebank-3	$3150.00
TIPSTER Complete	$500.00
NTIMIT	$500.00
YOHO Speaker Verification	$1000.00
Message Understanding Conference (MUC) 7	$500.00

The goal of this work is to explore the potential of the concept of data collection from the DVDs of movies and TV series, and to develop the application that utilizes this concept. Each component of the data collection system is described in starting from extracting the data from a DVD all the way to generating a corpus into a proper structure. Data Collection Toolkit is a C# .NET application developed to carry out this task is appropriately described in this section.

Some examples are provided of how to Data Collection Toolkit and speech corpora generated from this system can be used; namely: TIMIT corpus browsing for the study of TIMIT corpus, creating features for prosodic analysis of Wake-Up-Word's context detection, and training of CMU Sphinx's acoustic model.

To verify the quality of speech corpora generated from this system, TIMIT corpus is used for evaluating time markers of the words (where they start and end) from text transcriptions generated by force alignment process. The utterances from TIMIT corpus are also used for evaluating the error rates of the acoustic models trained by corpora generated from DVDs of movies and TV series

COMPONENTS OF DATA COLLECTION SYSTEM

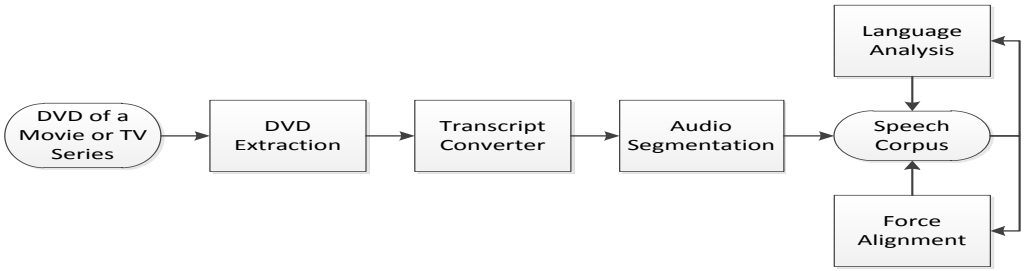
The data collection system consists of five components. The first step of the data collection system begins from selecting a DVD of a movie or TV series. The audio will then be extracted from the video of the DVD. The text from the subtitles will be converted into text transcriptions. The subtitles will also be used for cutting the extracted audio file into small utterances (audio segmentation). The text transcriptions and utterances are then combined into a basic speech corpus.

There are two optional components in the data collection system that can be used with the corpora generated from the DVDs for more specific tasks. First is the language analysis: this

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

component takes a text from a transcription and finds the relationship of each word. This analysis can be used for the study of Wake-Up-Word's context detection. Another component is the force alignment: which can be used for generating time markers of each individual word in the selected utterance. The force alignment can also be used for filtering and/or trimming utterances to ensure that they match with the corresponding text transcriptions. Figure shows an overview of the data collection system.

Figure 0: Overview of data collection system.

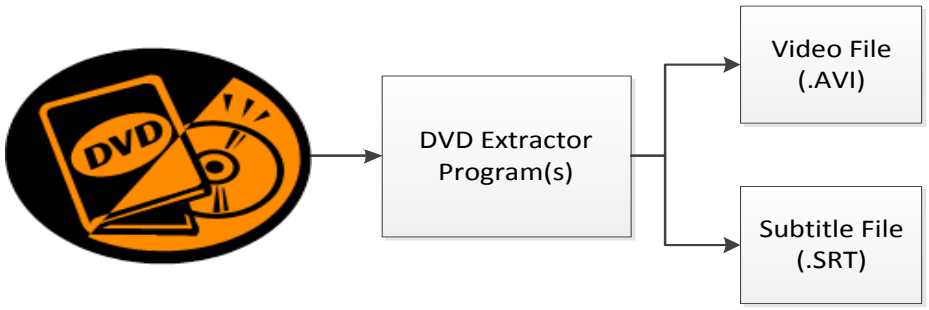


DVD Extraction

After the DVD of a movie or TV series has been chosen, the process of the data collection system starts with an extraction of the video and its subtitles (e.g., closed captioning) as illustrated in. First, software capable of extracting a video from a DVD and then storing it on a hard drive needs to be obtained. Any software that is free and able to extract the video into a file in AVI format would be desirable since the AVI file is commonly used and easy to extract the audio channel from it.

Subtitles, which are the textual representations of the utterances from the video, can be obtained by either downloading from a website or using a program such as SubRip, which can perform an Optical Character Recognition (OCR). The program will attempt to recognize the subtitles embedded on the video and save them into a text file. If there is any word that it cannot recognize, it will prompt a user to enter the word individually (Zuggy, 2009). Note that although these subtitles can be obtained easily, there is no guarantee that they are accurate in terms of how well the texts and the time markers from the subtitles are matched to the actual speech audio. Authors of subtitles may not follow the speech dialogue literally while writing subtitles of a particular movie or TV series. One of the mitigations for this issue is to use the force alignment, which is one of the data collection system's components.

Figure 1: Diagram of DVD extraction process.

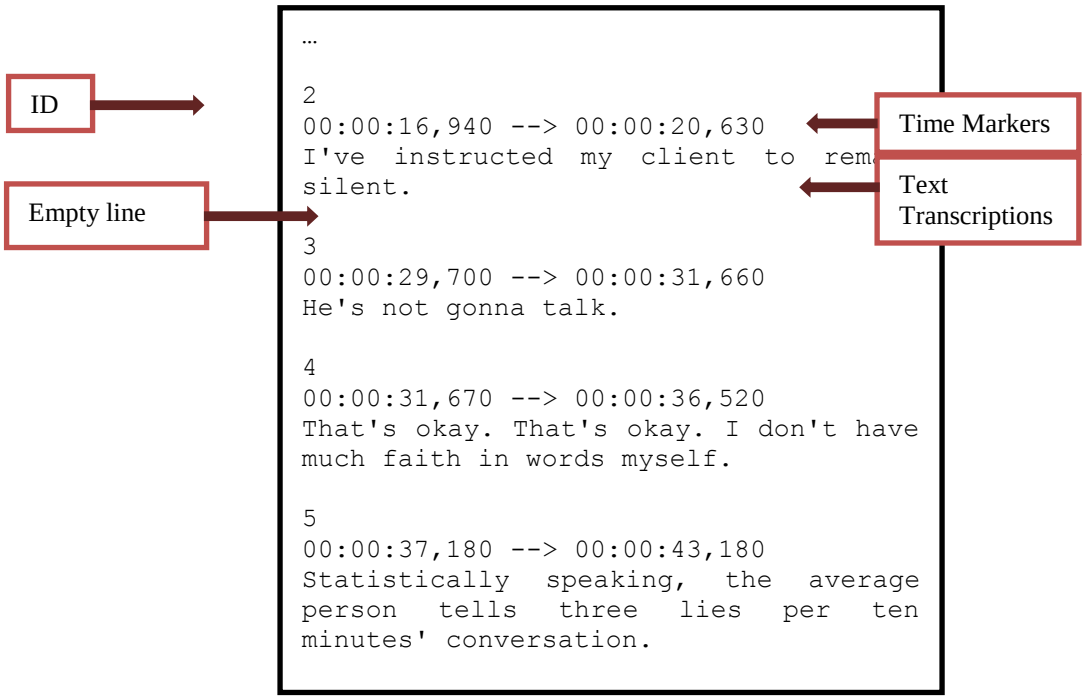


Transcript conversion

Transcript conversion is a process that converts a raw subtitle file into a simpler format to make it easier for a user to use. The standard format of a raw subtitle (SRT) file, shown in

Figure 1a, consists of subtitle indices, time markers, and transcriptions separated by new line. For each item of the subtitles there is a line with blank space right below to separate it from the other items. The initial proposed method on how to parse raw subtitles in the SRT file is to use a counter to keep track of the line. For instance, the first line contains the index; the second contains the time markers; and the third contains the transcription. Note that after the third line there is an empty line before the next subtitle begins. The problem with this assumption is that the text transcriptions may contain more than two lines. If that happens, the parser will extract the subtitles incorrectly.

Figure 1a: Example contents of subtitle (SRT) file.



The next attempt to reformat the subtitles purposed the use three regular expressions to determine what each line in the raw subtitle file is representing, then rewrite all the subtitles into a new text file that separates each subtitle by a new line and use "|" as delimiter to separate index, time markers, and transcriptions. Based on this idea, they developed an application called Movie Transcript Parser (Ramdhan & Beharry, 2009). The output of this application, as shown in Figure 2, shall be used for creating utterances and corresponding text transcriptions.

However, the Movie Transcript Parser was developed on Mac OS X environment as a preliminary research specifically for subtitle reformatting in 2009. The development of this application has been discontinued at this point. Although the program is very easy to use and perform the task quickly, a user must have a Mac OS X system in order to use it. Hence, a

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

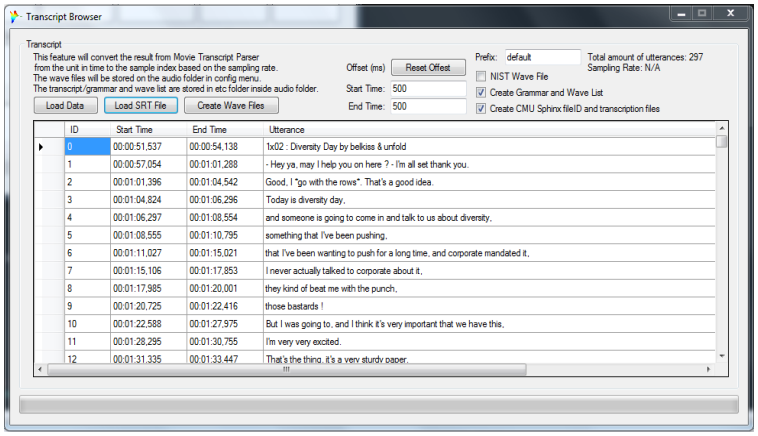
Windows user cannot use this application to reformat the subtitles into text transcriptions, which is needed for further corpus generation processes.

Figure 2: Example contents of subtitle in simpler format.

```
# Start Index|End Index|Start Time|End Time|Transcription
2|2|00:00:16,940|00:00:20,630|I've instructed my client to remain
silent.
3|3|00:00:29,700|00:00:31,660|He's not gonna talk.
4|4|00:00:31,670|00:00:36,520|That's okay. That's okay. I don't
have much faith in words myself.
```

One of the tools from the Data Collection Toolkit is the Transcript Browser. It, allows a user to load the result from the Movie Transcript Parser and then populate them on a grid-data table as shown in Figure 3. For a user who does not have an access to use Mac OS X, the Transcript Browser also allows a direct loading of a SRT subtitle file as an alternative. Hence, it will bypass the need of using Movie Transcript Parser. After the subtitles have been loaded into the data table, they can be used for creating utterance and text transcription files.

Figure 3: Screenshot of transcript browser.



Audio segmentation

Audio segmentation is a process, which generates a set of utterances from the time markers of the subtitles as illustrated in Figure 4. Prior to this process can begin, the audio from the AVI video file extracted from a DVD needs to be stored as a wave file using Microsoft's wave file tag header. The program used for this work to extract the audio is called AOA Audio Extractor. This program is chosen because it is capable to extract a mono-channel audio from multiple videos in batch mode (AoAMedia.Com, 2009). The sampling frequency of these extracted audios is set to 22 KHz, which is the lowest sampling frequency that the program allows a user to choose.

The tool also provides an additional option that allows a user to set the prefix of the utterances' file name as shown in Figure 57. After a user clicks on "Create Wave Files" button, the

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Transcript Browser will create several utterances and save them into wave files based the time markers of the subtitles. Each file name consists of the prefix followed by "_[Subtitle Index]". For example, if a prefix is set to "Office_S1E1", then the new segmented audio files will be generated as Office_S1E1_0.wav, Office_S1E1_1.wav, Office_S1E1_2.wav, until it reaches Office_S1E1_n-1.wav, where n is the total number of the utterances for that particular set of subtitles.

With the information from the subtitles populated on the data table, the Transcript Browser tool can begin the process of audio segmentation. First, the tool will convert the time markers of an utterance's text transcriptions (where it starts and ends) from "Hours: Minutes: Seconds, Milliseconds" format into sample indices using Equation 1. This pair of the sample index is then used for copying the portion of the audio from the DVD and to save it in a new wave file. The process will repeat until all the utterances have been created.

Figure 4: Example of audio segmentation based on given time markers.

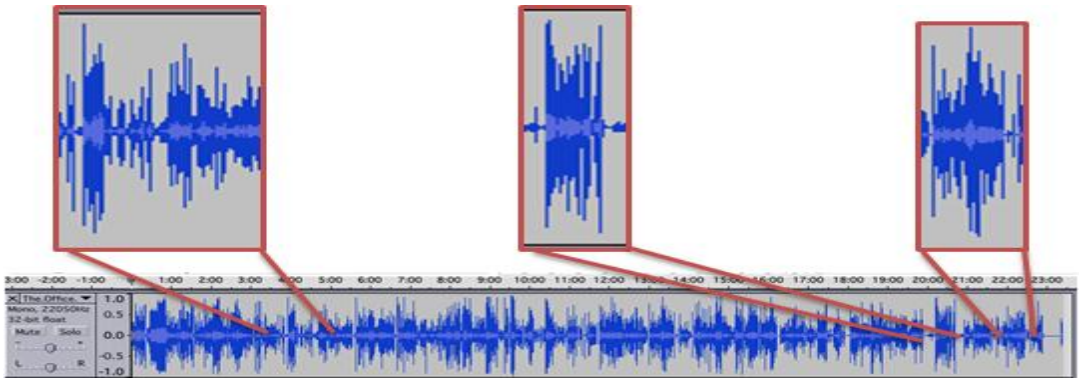
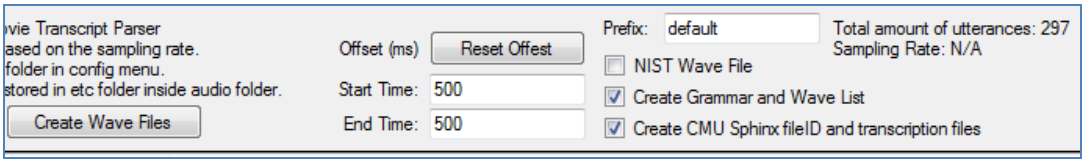


Figure 5: Screenshot of transcript browser's audio segmentation options.



In some cases, the subtitles obtained from the internet can have the overall time markers misaligned by a few seconds. This can happen due to the fact that the creation of these subtitles is done with a different version of the video recording. For example, that video may have its introduction part of the show removed to reduce the file size while the video extracted from the DVD includes the introduction part in it.

Equation 1: Conversion of time marker to sample index.

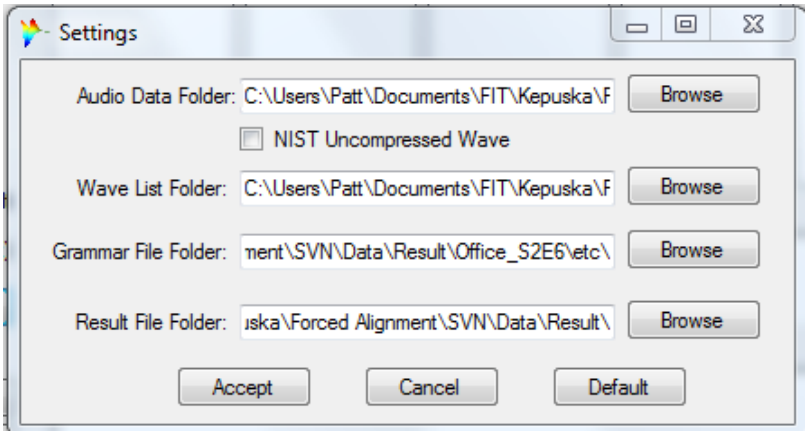
$$Sample\ Index = \{(Hours\ x\ 60\ x\ 60\ x\ 1000) + (Minutes\ x\ 60\ x\ 1000) + (Seconds\ x\ 1000) + Milliseconds\}\ x\ \frac{sampling\ rate}{1000}$$

To overcome this misalignment issue, Transcript Browser tool allows a user to set two global offsets. One is to subtract from the start time, another one is to add the end time of the utterance. A user can make some adjustments on the two global offsets to ensure that the overall subtitle time markers are matched with the actual speech utterances.

After the utterances have been created by Transcript Browser, a user can easily verify if the overall time markers match with speech utterances by double clicking on the text transcriptions on the data-table. By doing so, the Transcript Browser will invoke a media player to play the selected utterance. If the utterance does not match with the transcriptions, a user can change the offsets again and recreate the utterances' wave files. The new wave files will overwrite the old one without a need to ask for permission. Hence, this process can be iterated over and over until the utterances are consistent with the transcriptions.

After the utterances are aligned properly with the subtitles, the corpus is then successfully generated. A user can access the generated corpus by going to folder with the same name as the prefix. This prefix is located inside the path of the "Result File Folder" defined in the settings window in the main page of Data Collection Toolkit as shown in Figure 6. Note that if a user changes the path of the resulting file folder, then the utterances will need to be recreated again in order to be stored in a new path.

Figure 6: Screenshot of data collection toolkit settings.



A structure of a corpus folder created from Data Collection Toolkit's Transcript Browser is organized in the same way as the corpora dedicated to CMU Sphinx speech recognition toolkit as shown in Figure 7: a "wav" folder will contain all wave files of the utterances; an "etc" folder will contain the following files:

- **FileIDs** – contains a list of the utterances (there is no file extension after the name).
- **Transcription** – contains the list of transcriptions nested by <s> … </s> tag and followed by the corresponding utterance in parentheses.
- **Wave list** – contains a list of all wave files (specified file extension).
- **Grammar list** – contains a list of transcriptions only.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Figure 7: Structure of DVD corpus generated with data collection toolkit.

```
[Result Path]\[Prefix]\
                    \etc\
                          [prefix].fileids
                          [prefix].transcriptions
                          [prefix]_grammarlist.txt
                          [prefix]_wavelist.txt
                    \wav\
                          [prefix]_0.wav
                          [prefix]_1.wav
                          …
                          [prefix]_n-1.wav
```

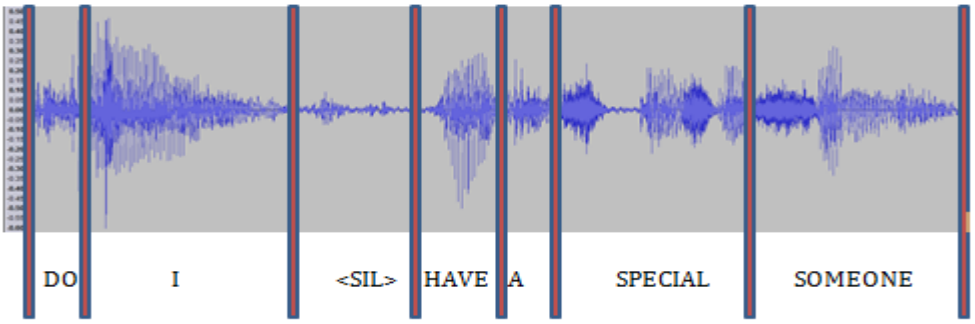
The file ID and transcription files are used as inputs for CMU Sphinx's acoustic model training while the wave list and grammar list are used for SAPI Force Alignment. The detailed descriptions of these files are mentioned in the later sections of the work.

Force alignment

The term force alignment refers to a technique that forces a speech recognition engine to try to recognize each utterance given its corresponding text transcription. This process is illustrated in Figure 8 below. Force alignment needs to use the speech recognition engine that is capable of recognizing the utterances and then returning the time markers of the words in the selected utterances. If the engine cannot locate all the given words as part of the selected utterance, then it will not return the result (e.g., it will fail to perform the recognition).

The time markers for each word generated by the force alignment can be used for several tasks such as prosodic analysis. To preform prosodic analysis it is required to have time marker of a targeted words in order to be able to estimate/compute pitch or energy of that section of the utterance. The time markers can also be used for finding the start time of the first word and the end time of the last word. The time markers of the entire utterance can be used for trimming the wave file of that utterance to ensure that the text transcriptions are matched with the actual speech audio.

Figure 8: Illustration of force alignment.



ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Recognized Utterance	Pronunciation	Confidence	Time Start (ms)	Time End (ms)
DO	du	0.0257358234	0	50
I	ai	0.325056046	50	580
HAVE	hæv	0.858041942	980	1240
A	ei	0.9428072	1240	1340
SPECIAL	spɛʃəl	0.902040541	1340	1880
SOMEONE	sʌmwʌn	0.9509962	1880	2440

The current speech recognition engine chosen to perform a force alignment is bundled with Microsoft Windows 7 (or Vista). This engine can be interfaced via Microsoft's Speech Application Interface (SAPI), which is a part of the .NET Framework (MSDN, 2010). Hence, the speech recognition engine becomes a force alignment engine that was termed SAPI Force Alignment.

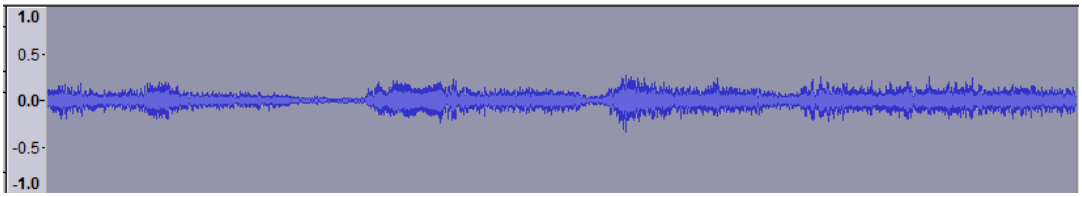
The main reason for integrating the SAPI Force Alignment into the Data Collection Toolkit is because of the acoustic model that uses in this engine has a good recognition rate without any training required. Also, the toolkit itself is developed on .NET Framework in C# language, which can invoke the functions from the SAPI on the same framework. In the other words, using SAPI and its built-in engine is easier to integrate it into the toolkit comparing to using other open source engines, which usually are developed on Linux environment.

Since the SAPI Force Alignment does not have a built-in force alignment function, additional coding on top of the SAPI's speech recognition function is required in order to gain the ability to perform a force alignment.

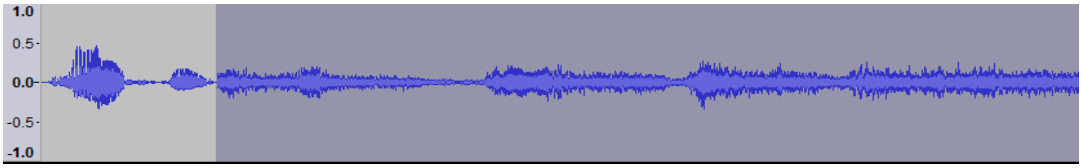
First, the recognition result from the SAPI only returns the time markers relatively to the first word it detects. In the other words, it will return the time markers of the first word to be 0, which is not always true for all utterances. There is a chance the first word may be located in the middle of the recording. This will cause SAPI to return incorrect time markers for all the words in that utterance.

To solve this issue, the utterance that will be used as the input to the SAPI will need to be concatenated by a filler word that has a known duration. For example, the word "VoiceKey" was chosen for this purpose. This word that has the duration of 400 milliseconds is concatenated in front of the selected utterance as shown in Figure 9. The text of this filler word will be added into the list of the words from the selected utterance. This list is also known as a grammar list, which the SAPI Force Alignment will try to force align the audio of the utterance with each word from the list and return the time markers if it can successfully recognize them.

Figure 9: Arbitrary utterance concatenated with filler word "Voicekey."



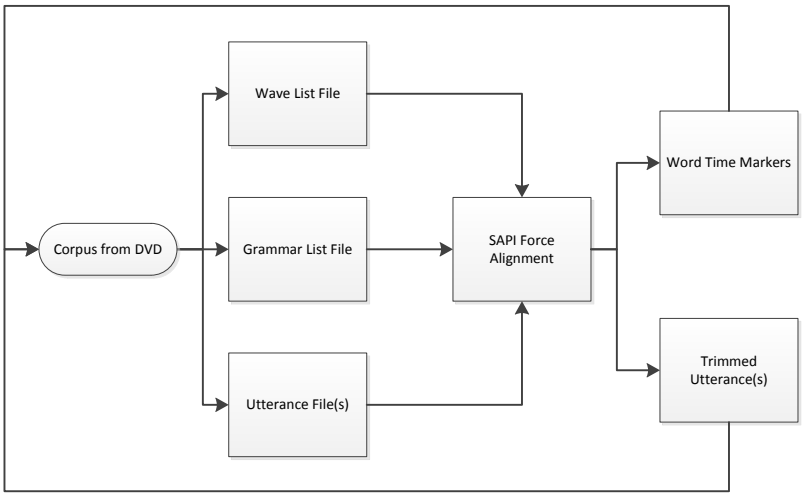
ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy



Another step prior creating a SAPI force alignment function, and allowing it to run in batch mode, is to use loop iteration technique. For each time the program iterates, it will load one file and one grammar into the speech recognition engine using SAPI. After completing processing of that utterance, it will erase the inputs before loading the next pair. This process will repeat until all the files from the lists have been processed. The SAPI Force Alignment tool and the process that it applied is illustrated in Figure 12.

Once the SAPI Force Alignment tool has finished its task, the tool will save the result files into the resulting folder with the name defined by "Result File" text box as shown in Figure 10. It also allow a user to save the force alignment results similar to TIMIT format if a user checks on "Save Force Aligned Utterance Result". These results will be stored on the same folder with the wave files of the corpus, so that it can be used in Force Alignment Result Browser tool.

Figure 12: Overview of SAPI force alignment tool in data collection toolkit.



Each utterance will then have the two files with the same name as the wave file:

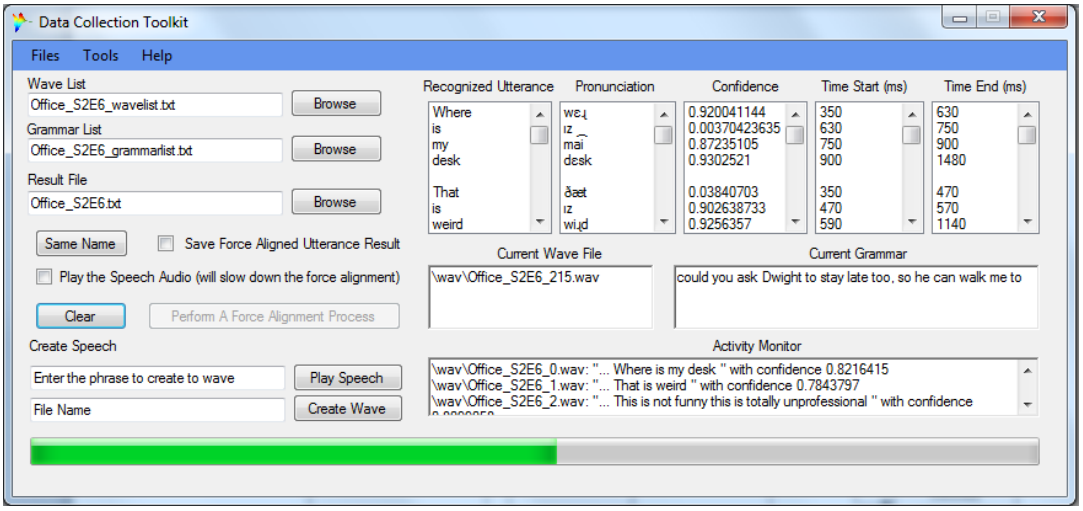
1. **TXT_FA** – contains the time markers for the sentence.
2. **WRD_FA** – contains the time markers for individual words.

Note that the contents of the TXT_FA and WRD_FA files are stored in the same format as TIMIT's TXT and WRD files, respectively. They have "_FA" suffix embedded in their name, so that they will not overwrite the default TIMIT's files when the SAPI Force Alignment tool is being used on TIMIT corpus for the evaluation.

Another tool from the Data Collection Toolkit called TIMIT Corpus Browser can load these four files and populate a grid-data table for a user to compare the time markers of the generated results that uses force alignment with the TIMIT's default time markers.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Figure 10: Screenshot of SAPI force alignment tool.



Language analysis

Language analysis is a process, which takes a line of text transcriptions as an input into a program that extracts that line and then returns a list of dependency relationship between its words. The current program used for this process is called RelEx Dependency Relationship Extractor, which is a module of OpenCog the artificial intelligence application project (OpenCog Wiki, 2010).

The main purpose of the language analysis is for the study of prosodic analysis, where the context of a keyword is determined based on its relationship within its sentence or phrase. For example, a keyword "Junior" is used in the table below in two different sentences. The context of this same word from the two sentences is actually different from each other. The highlighted texts from Table 22 are shown to be relationship tags that are distinguishable between the two contexts.

Table 2: Example results from RelEx.

Context	Referential	Alerting
Example Sentence	Junior grabs a pen and hands it to me.	Junior, grab a pen and hand it to me please?
Dependency Relation	_subj(and, Junior) tense(and, present) inflection-TAG(and, .j-v) pos(and, verb) _obj(grab, pen) tense(grab, present) inflection-TAG(grab, .v) pos(grab, verb) inflection-TAG(pen, .n) pos(pen, noun) noun_number(pen, singular) pos(a, det) gender(Junior, masculine) definite-FLAG(Junior, T) inflection-TAG(Junior, .m) person-FLAG(Junior, T) pos(Junior, noun)	to(hand, me) _obj(hand, it) _advmod(hand, please) tense(hand, present) inflection-TAG(hand, .v) pos(hand, verb) inflection-TAG(please, .e) pos(please, adv) pos(?, punctuation) pronoun-FLAG(me, T) gender(me, person) definite-FLAG(me, T) pos(me, noun) noun_number(me, singular) inflection-TAG(to, .r) pos(to, prep) pronoun-FLAG(it, T)

	noun_number(Junior, singular) to(hand, me) _obj(hand, it) tense(hand, present) inflection-TAG(hand, .v) pos(hand, verb) pos(., punctuation) pronoun-FLAG(me, T) gender(me, person) definite-FLAG(me, T) pos(me, noun) noun_number(me, singular) inflection-TAG(to, .r) pos(to, prep) pronoun-FLAG(it, T) gender(it, neuter) definite-FLAG(it, T) pos(it, noun)	gender(it, neuter) definite-FLAG(it, T) pos(it, noun) tense(and, infinitive) HYP(and, T) inflection-TAG(and, .j-v) pos(and, verb) _to-do(Junior, and) tense(Junior, imperative) gender(Junior, masculine) definite-FLAG(Junior, T) inflection-TAG(Junior, .m) person-FLAG(Junior, T) pos(Junior, noun) noun_number(Junior, singular) pos(,, punctuation) _obj(grab, pen) tense(grab, present) inflection-TAG(grab, .v) pos(grab, verb) inflection-TAG(pen, .n) pos(pen, noun) noun_number(pen, singular) pos(a, det)

Although RelEx is considered as open source software, it has not yet been integrated into Data Collection Toolkit. That is because the program itself and its dependency software are developed on Linux environment, which are not fully compatible to be run on Microsoft Windows environment. There have been several attempts to install the software on Windows by using Cygwin, a shell that emulates Linux environment on Windows. However, the attempt was not successful due to the incompatibility of the dependency software.

Because of the compatibility and limitation of time issues, the most feasible way to determine the context of the key words from the sentences is to line them all up in MS Excel spreadsheet and then put a tag label on each of them manually: alerting, referential, or none. After that, use the sort function in the spreadsheet to group up the sentences of the same context. Then they can be used for further tasks, such as training a classification model for prosodic analysis.

This component of the data collection system requires further development in order to automate the entire process of separating the set of utterances by their contexts, as is described in Future Work section.

APPLICATIONS OF DATA COLLECTION TOOLKIT

The Data Collection Toolkit originated as a force alignment program, which then gradually evolved into a more versatile toolkit used for collecting data from DVDs and then generating corpora from them. The applications of the toolkit include but are not limited to the following tasks: TIMIT corpus browsing, prosodic analysis, and acoustic model training.

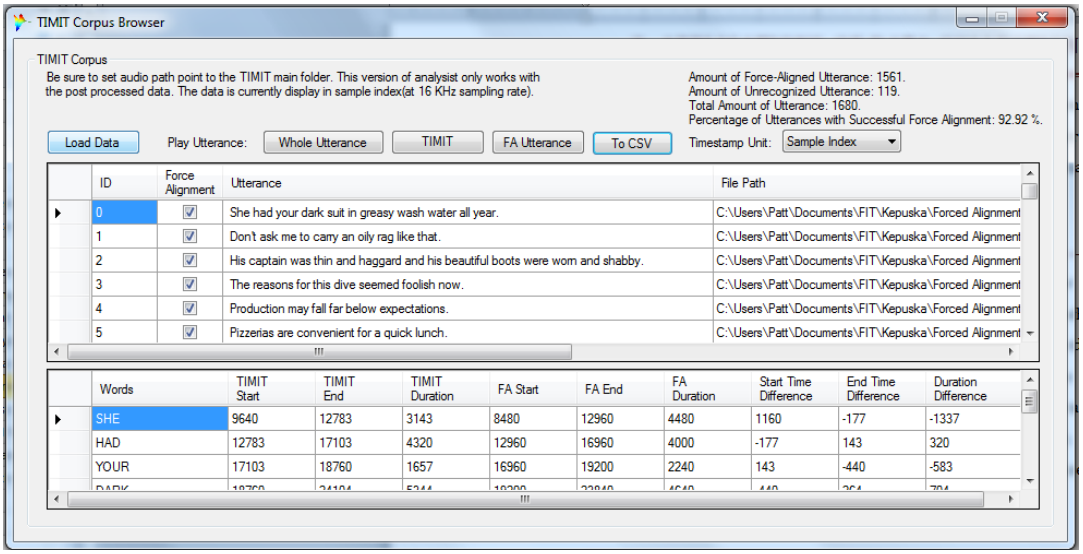
TIMIT corpus browsing

The Data Collection Toolkit provides a tool called TIMIT Corpus Browser (see Figure 14), which allows a user to load and view TIMIT corpus in a form that is more organized and easier to view TIMIT's utterances. It also provides options to play an audio of any utterance and its individual words using time markers labeled by TIMIT or SAPI Force Alignment (if available).

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

There is also an option to export the information of the TIMIT corpus into a Comma Separated Value (CSV) file in case a user need to perform further data analysis such as drawing graphs, histograms, or any other numerical computation using Microsoft Excel spreadsheet (see Figure 15).

Figure 14: Screenshot of TIMIT corpus browser.



There are two types of representation for the words' time markers. One is in sample index, which is useful for those who need to locate a word in a wave file when it is loaded into data memory as a binary data. Another one is in milliseconds, which is useful for human to observe the corpus.

Equation 2 and Equation 3 shows how these two units for time markers can be converted between each other.

Figure 15: Example CSV file generated from TIMIT corpus browser in MS Excel

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Data Collection Toolkit: TIMIT Corpus Browser													
2	Value Display: Sample Index													
3	word	TIMIT star	TIMIT enc	FA start ti	FA end tir	delta star	delta end	delta dur	file path	whole transcription				
4	SHE	9640	12783	8480	12960	1160	-177	-1337	C:\Users\I	She had your dark suit in greasy wash water all year.				
5	HAD	12783	17103	12960	16960	-177	143	320	C:\Users\I	She had your dark suit in greasy wash water all year.				
6	YOUR	17103	18760	16960	19200	143	-440	-583	C:\Users\I	She had your dark suit in greasy wash water all year.				
7	DARK	18760	24104	19200	23840	-440	264	704	C:\Users\I	She had your dark suit in greasy wash water all year.				
8	SUIT	24104	29179	23840	29280	264	-101	-365	C:\Users\I	She had your dark suit in greasy wash water all year.				
9	IN	29179	31880	29280	32000	-101	-120	-19	C:\Users\I	She had your dark suit in greasy wash water all year.				
10	GREASY	31880	38568	32000	38560	-120	8	128	C:\Users\I	She had your dark suit in greasy wash water all year.				
11	WASH	38568	45119	38560	44480	8	639	631	C:\Users\I	She had your dark suit in greasy wash water all year.				
12	WATER	45624	51033	44480	51840	1144	-807	-1951	C:\Users\I	She had your dark suit in greasy wash water all year.				
13	ALL	52378	55461	51840	55040	538	421	-117	C:\Users\I	She had your dark suit in greasy wash water all year.				
14	YEAR	55461	60600	55040	62560	421	-1960	-2381	C:\Users\I	She had your dark suit in greasy wash water all year.				

Equation 2: Computation for a time marker in sample index.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

$$Time_{sample} = \frac{Time_{ms} x\ Sampling\ Rate}{1000}\ [sample\ index]$$

Equation 3: Computation for a time marker in milliseconds.

$$Time_{ms} = \frac{Time_{sample} x\ 1000}{Sampling\ Rate}\ [ms]$$

There is also additional information that is computed from the time markers: the word duration and the time differences. These values can be computed by Equation 4 and Equation 5 respectively.

Equation 4: Word duration.

$$Word\ Duration = End\ Time - Start\ Time\ [ms\ or\ sample\ index]$$

Equation 5: Time Difference between TIMIT's and force alignment's time markers.

$$Time\ Difference = Time\ Stamp_{TIMIT} - Time\ Stamp_{FA}[ms\ or\ sample\ index]$$

Note that a user can use this tool to listen to the utterances and their words (more details on TIMIT corpus are described in Appendix B), which is more convenience than going through each utterance file directly one by one in the file folder.

The TIMIT Corpus Browser tool is used for evaluating the time markers generated from SAPI Force Alignment tool's results with TIMIT corpus's default time markers manually prepared by humans.

Prosodic analysis

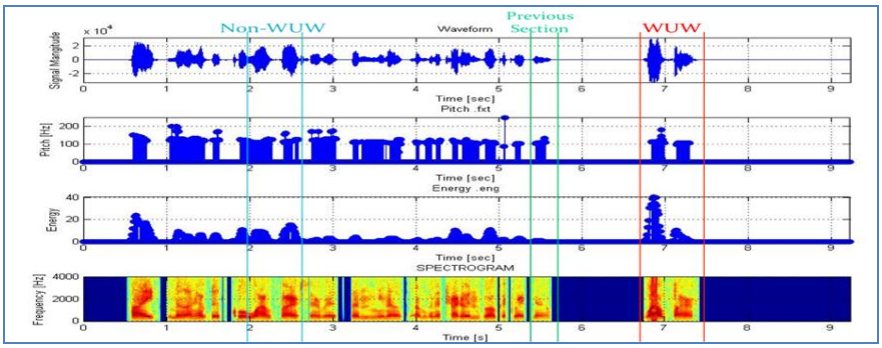
Prosodic analysis is a study of intonation and rhythm of a word in an utterance. The goal of this study is to create additional prosodic features, which then can be used for developing Wake-Up-Word (WUW) speech recognition system, particularly in context detection modeling. Basically, the system will continuously listen to the speech input via an audio stream (i.e. microphone) and check if the speech input contains a keyword. If so, it will determine which context this word belongs in order to ensure that the system will only be in alert state if a user specifically intends to wake the system up with the keyword (Këpuska & Shih, 2010).

The other members of research team have already developed the system that can generate the prosodic features based on the pitch and energy of the selected utterance as illustrated in **Error! Reference source not found.**. Currently, the prosodic features extracted from WUW-II corpus have already been used for training a WUW's context detection model with Support Vector Machine (SVM) tool called LIBSVM. The results showed that the model using pitch-based features performed at 80.98% accuracy rate while the model using energy-based features performed at 76.79% accuracy rate. When these two types of features are used, the model can perform at 83.67% (Sastraputera, 2009).

In this example, Figure 16, the utterance of WUW "Wildfire" is used in referential and alerting context: "Hi you know I have this cool wildfire service and you know I'm gonna try to invoke it right now Wildfire!"

Figure 16: Illustration of prosodic analysis.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy



However, to improve the accuracy of the context detection model, the more natural corpora are needed. In addition, each utterance from the corpora requires having the time markers for individual words in order to be used for computing prosodic features (Shih, 2009).

The need of obtaining more data for prosodic analysis make the data collection system from the DVDs becomes useful for this study. Corpora generated from this system have utterances that contain natural speech from the dialogue of professional actors/actresses, which will help improving the quality of the WUW context detection model. Also, the SAPI Force Alignment tool can generate time markers of individual words, so a user will not have to label the words manually. Using this system to create the corpora from the DVDs can save a significant amount of time to perform tedious tasks and potentially reduce human errors.

Acoustic model training

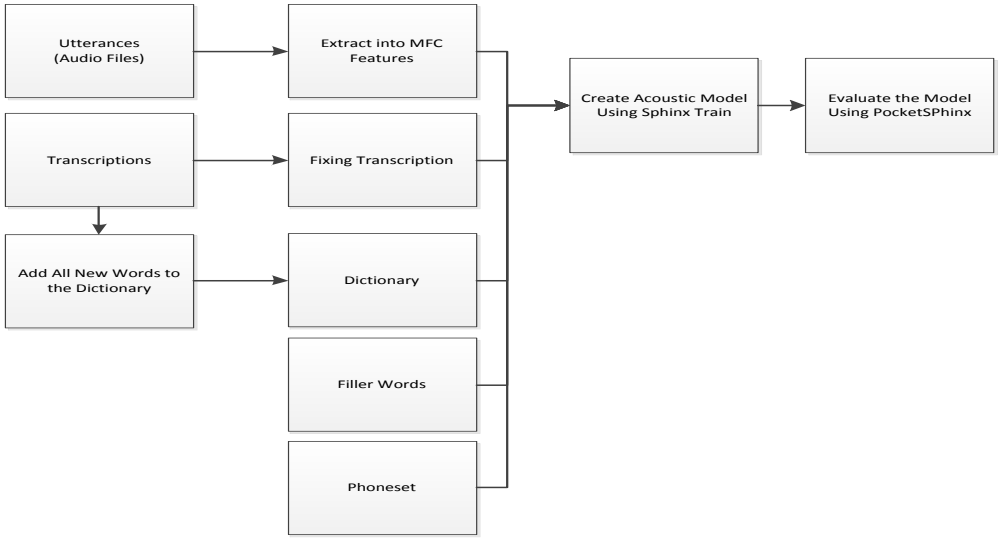
An acoustic model is used by a speech recognition engine to recognize and convert a speech utterance into its corresponding textual representation. Similar to any other supervised classifiers, in order for a user to create a speech recognition system, there is a need to train an acoustic model using a training set of data, which in this case are the speech corpora. The toolkit is chosen to generate the data in order to perform the acoustic model training using CMU Sphinx toolkit. This tool is open source software developed by Carnegie Melon University, which contains software tools required for creating a speech recognition system.

The Figure shows the flowchart diagram of the processes required to create the acoustic model for the speech recognition system. Two tools from CMU Sphinx Toolkit are used: SphinxTrain for training the acoustic model and PocketSphinx for evaluating that model. The tutorial on how to install and use these two tools is located in Appendix A.

In addition to the corpus's fileIDs and transcription files generated from the Data Collection Toolkit's Transcript Converter tool, the CMU Sphinx also requires three more files prior the acoustic model training can begin:

Figure 17: Flowchart Diagram of Acoustic Model Training.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy



- **Dic** – A dictionary phonetic file, which lists a phonetic transcription of all words. By convention, this file should have all the unique words from the transcriptions listed in alphabetical order followed by their set of sounds (or phonemes). If a word has more than one way it can be pronounced, then there will be an additional line of the same word followed by (n) and its phonemes, where n is a number of the additional ways of pronunciation as shown in
- Figure 18.

Figure 18: Example of dictionary file MyCorpus.dic.

```
JUST   JH AH S T
JUST(2)       JH IH S T
LISTENING     L IH S AH N IH NG
LISTENING(2)  L IH S N IH NG
ME     M IY
MICHAEL       M AH CH IY L
NO     N OW
ON     AA N
ON(2)  AO N
OUT    AW T
PAM    P AE M
PAM!   P AE M
PRIZE  P R AY Z
SO     S OW
STUPID S T UW P AH D
TELL   T EH L
THIS   DH IH S
TO     T UW
TO(2)  T IH
TO(3)  T AH
WIN    W IH N
YOU    Y UW
```


ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

- **Filler** – A filler dictionary contains filler phones that are not covered by language model such as breathing sound, hmm, laugh, and/or silence. The filler file normally stored in .filler extension. From Figure , each line of the filer word file contains two set of texts separated by tab. A set of texts on the left shows a word, which appear on the transcription file. The one on the right shows its match in the dictionary file. For example, every time the Sphinx Train sees <s>, </s>, or <sil>, it will treat them as if they are SIL (silence sound). This filler dictionary is also useful for reducing the redundancy and the size of the dictionary file.

Figure 19: Example of filler words in MyCorpus.Filler.

```
<s>                SIL
</s>               SIL
<sil>              SIL
++AH++             +AH+
++BEEP++           +TONE+
++NOISE++          +NOISE+
++DOOR_SLAM++      +SLAM+
++BREATH++         +BREATH+
++GRUNT++          +GRUNT+
++LIP_SMACK++      +SMACK+
++PHONE_RING++     +RING+
++CLICK++          +CLICK+
```

- **Phone**- A phone set file that lists all the phones used in the phonetic dictionary, in alphabetical order. A universal phone set file can be used as long as the file contains at least all the phones listed in the dictionary file. Any extra phones listed in the phone set file, but are not being listed in the dictionary file will flag warnings on the CMU Sphinx Train, but does not interfere with the training process.

- Table shows the list of all phones used by CMU Sphinx. Note that in for contents of a .phone file requires to have only one phone per line.

Table 4: List of all phones used by CMU Sphinx.

Phones	Example	Phoneme		Phones	Example	Phoneme
AA	ODD	AA D		L	LEE	L IY
AE	AT	AE T		M	ME	M IY
AH	HUT	HH AH T		N	KNEE	N IY
AO	OUGHT	AO T		NG	PING	P IH NG
AW	COW	K AW		OW	OAT	OW T
AY	HIDE	HH AY D		OY	TOY	T OY
B	BE	B IY		P	PEE	P IY
CH	CHEESE	CH IY Z		R	READ	R IY D
D	DEE	D IY		S	SEA	S IY
DH	THEE	DH IY		SIL	(Silence filler)	(Silence filler)

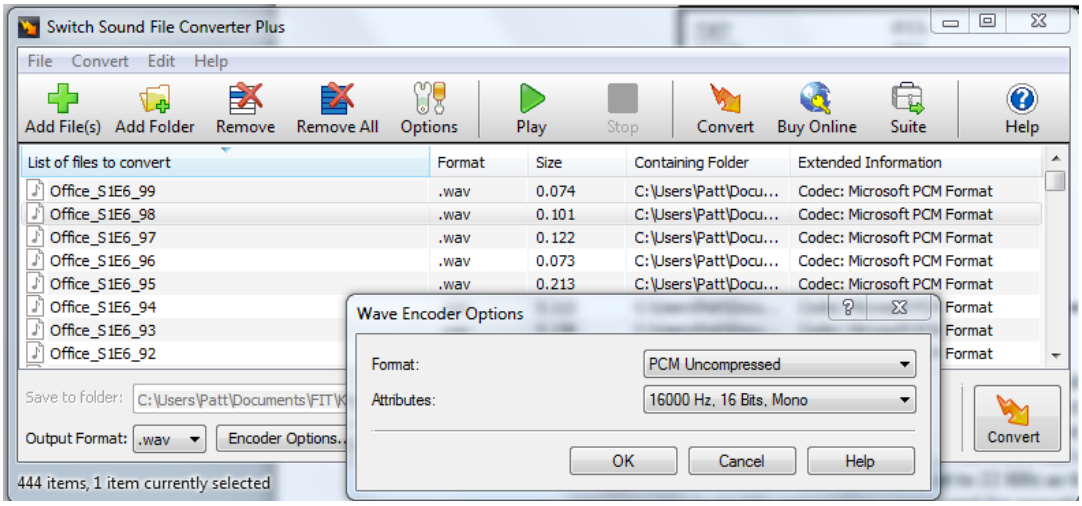
ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

EH	ED	EH D		SH	SHE	SH IY
ER	HURT	HH ER T		T	TEA	T IY
EY	ATE	EY T		TH	THETA	TH EY T AH
F	FEE	F IY		UH	HOOD	HH UH D
G	GREEN	G R IY N		UW	TWO	T UW
HH	HE	HH IY		V	VEE	V IY
IH	IT	IH T		W	WE	W IY
IY	EAT	IY T		Y	YIELD	Y IY L D
JH	GEE	JH IY		Z	ZEE	Z IY
K	KEY	K IY		ZH	SEIZURE	S IY ZH ER

The first step before a corpus generated from a DVD can be used for CMU Sphinx's acoustic model training is to make sure that all the utterances are stored in 16-bit 16 KHz mono-channel wave files. Since in this work the AOA Media Extractor is being used to extract the audio channel from the DVD's video, the audio file has a sampling frequency at 22 KHz. Hence the utterances generated from this audio will have to be converted to 16 KHz. A free version of NCH's "Switch" Sound File Converter is chosen for the audio conversion since the software is capable of converting wave files into the required encoding (PCM uncompressed) and sampling frequency (16 KHz) in batch mode as shown in Figure , which make the conversion process to be done automatically once it is being set to the proper configurations (NCH Software, 2010).

After the wave files of the utterances are all converted to 16 KHz, the next step is to look into the transcriptions of the utterances. The first thing to look for is the alphabet character mismatch. There is a chance that the subtitle file may be written in different encoding from the one that is used in Data Collection toolkit (i.e. Unicode vs. UTF-8). For example, a transcription created from a result of Movie Transcript Parser somehow shows the accent alphabet "í" on the place where it is supposed to be an apostrophe. For example, an arbitrary sentence transcript "You havenít told me!" should be "You haven't told me!" instead. This kind of mismatch can be easily fixed by using the Perl script as described in Appendix B.

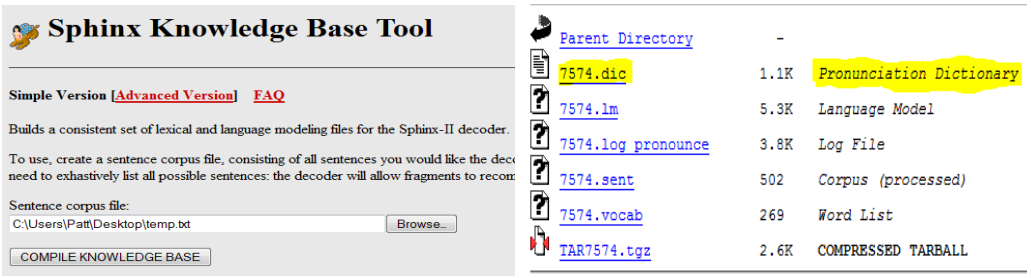
Figure 20: Screenshot of "Switch" the batch wave file converter software.



Once all the character encoding issue and other typos in the transcriptions have been solved, a user can start running Sphinx Train to train the acoustic model. There may be errors on the first run. If all the input files are prepared correctly, then it is very likely that the words on the transcriptions are not included in the dictionary. The user then needs to add all new words and their set of phone labels into the dictionary.

The web service from Sphinx Knowledge Base Tools known as LMtool can be used for generating a set of phones for these new words (Rudnicky, 2010). The tool will take an input as text file, which contains corresponding transcriptions of a sentence (or an utterance) one per one line. After the LMtool has done processing, it will forward to the next web page, which allows a user to download the dictionary file into the system drive as shown in Figure . The words from this new dictionary then need to be added to the current universal dictionary.

Figure 21: Screenshot of LMtool used for expanding dictionary file.



Note that the LMtool has a limitation, which it can only process a text file that contains up to 5000 sentences (or utterances). If the corpus transcription file has a larger size than LMtool's limit, CMUclmtk is another language model tools that can be used, but it required an installation into the user's computer prior being able to process the transcription (CMU Sphinx, 2010).

The next step is to look into all the text transcriptions for those which contain numerical digits. Since CMU Sphinx Train recognizes everything that appears on the transcriptions as words separated by white space, so it recognizes a group of digits as words as well. That means in order for the CMU Sphinx to be able to train acoustic model with the transcript with a set of digits, it has to have all possible pronunciations of these digits in the dictionary.

Adding all pronunciations of numbers into the dictionary can significantly expand the size of the dictionary file, which will decrease the efficient of the Sphinx's tools since it will take much longer to load a dictionary and perform a search for pronunciation of each word in a transcription. One proposed method is to remove these utterances that have digits in their transcriptions, but the size of corpus will be reduced and will take more time to modify the fileIDs and transcriptions files to remove all the transcriptions with digits.

Table 5: Example of various ways to represent digits in words.

Numerical Digits	Possible Representation in Words
2010	• TWO ZERO ONE ZERO • TWO THOUSAND TEN

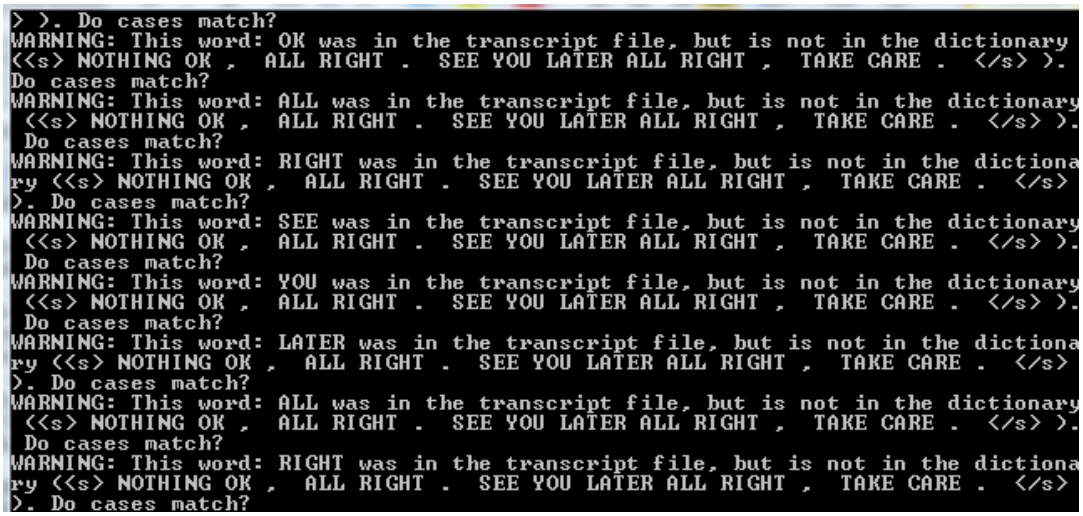
ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

	• TWO THOUSAND AND TEN • TWENTY TEN
1996	• NINETEEN NINETY SIX • ONE NINE NINE SIX • ONE THOUSAND NINE HUNDRED AND NINETY SIX

Another proposed method is to go into the transcription that has digits, listen to the utterance and rewrite the digits as words as shown in Table . This way, a user only has to edit the transcription file and also gets to verify if the transcription actually matches with the utterance.

Note that a user can also use Sphinx Train to identify the problematic lines of the transcriptions since it points out which words are not located in the dictionary files, an example of which is shown in Figure . When these warnings happen, they can be triggered by either: (1) the words are incorrectly spelled, (2) they are not in the dictionary, or (3) they are a set of numerical digits.

Figure 22: Screenshot of the warning report by CMU Sphinx Train.



After all the texts in the transcription file have been corrected, Sphinx Train should be able to perform all its tasks to create an acoustic model and store into the folder /model_parameter/. The contents in this folder are the parameters for the Hidden Markov Model (HMM) that CMU Sphinx uses for performing a speech recognition task for recognizing speech utterances and convert it into textual representation. The evaluation of the acoustic models created from Sphinx Train using different corpora created from DVD of movies and TV series is discussed later.

FORCE ALIGNMENT EVALUATION

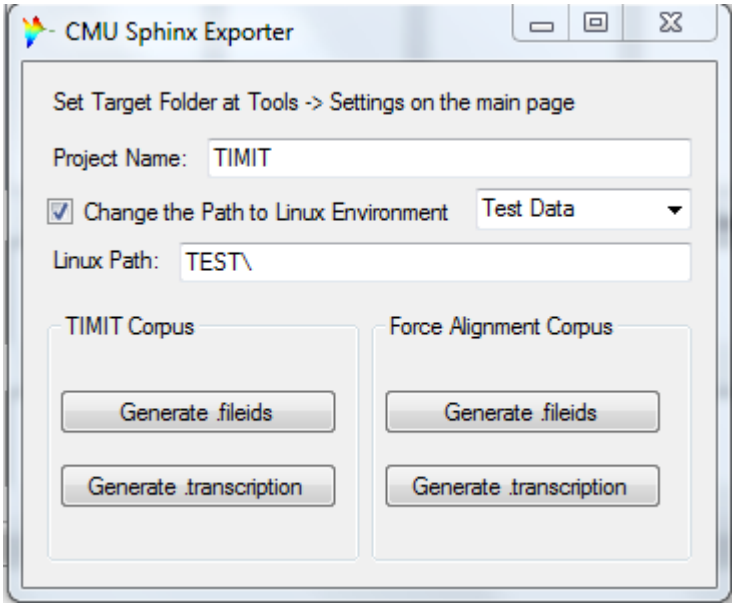
The TIMIT corpus was used for evaluation in part because it has manually aligned words as well as phonemes as well as it is designed for Training as well as Testing. In order to perform an evaluation of the SAPI Force Alignment tool on TIMIT data, the grammar list and wave list files needed to be generated first. The TIMIT corpus does not contain a file that lists all the utterances

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

and their transcriptions. Going through all TIMIT's utterances, and then adding all the wave files into the corresponding list file along with all the transcription from the TXT file to the grammar list file would take a significant amount of time since there are 6,300 utterances in the corpus.

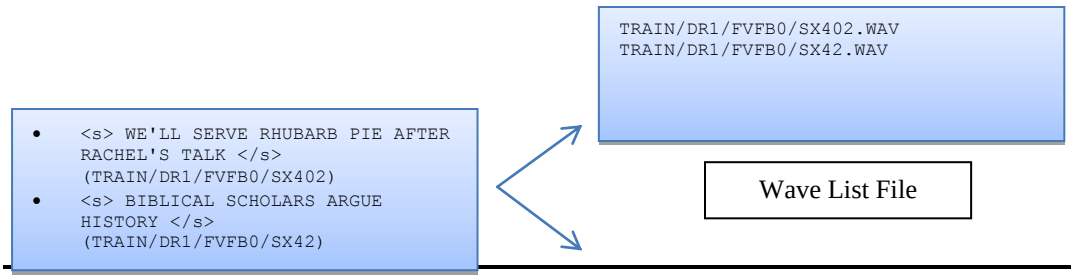
In this work we have proposed a simpler solution to this task; a user may use Data Collection Toolkit's CMU Sphinx Exporter (as shown in Figure 2) to generate a transcription file for TIMIT corpus. In a transcription file, each line contains a text nested with <s> … </s> followed by a file name nested with parentheses. The transcription part (without <s>…</s> tags) will be stored in a grammar list file and a file name part will be stored in wave list file (need to include .WAV extension) as shown in Figure .

Figure 23: Screenshot of data collection toolkit's CMU sphinx exporter tool.

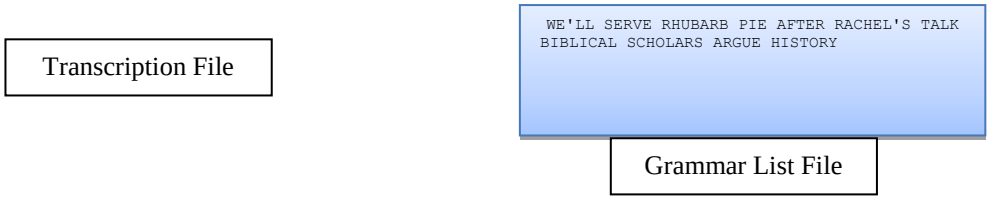


These two input files extracted from the transcription file are then fed into the SAPI Force Alignment tool. Note that since the paths of the utterances in the wave list file start with "/TRAIN/" or "/TEST/" folder, so the audio path in the SAPI Force Alignment's settings needs to be pointed to the main folder of TIMIT corpus which is on top of TRAIN and TEST folders. Also input wave file type needs to be set to NIST, so the wave files are read with the correct header tag. In addition to that, a user must check the option "Save Force Aligned Utterance Result" before starting a force alignment process. With that option checked, the results will be stored into TIMIT's subfolders, which can be viewed with the TIMIT Corpus Browser tool.

Figure 24: Extracting wave list and grammar list from transcription file.



ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy



Once the initial settings have been set, the force alignment process can begin. Due to the size of TIMIT corpus (6300 utterances), the process may take up to two to three hours to finish the force alignment of the entire corpus. After the force alignment process is finished, the TIMIT Corpus Browser tool is then used for loading the entire TIMIT corpus to show which utterances have a successfully force aligned as percentage of the total. The formula used for this computation is given in Equation . SAPI Forced Alignment tool was successful in 91.94% as shown on

Table 6.

Equation 5: Computation for force alignment recognition rate in percentage.

$$Percentage = \frac{Number\ of\ Successful\ Force\ Aligned\ Utterances}{Total\ Number\ of\ Utterances\ in\ a\ Corpus} x100\%$$

Table 6: TIMIT corpus utterances.

Corpus Type	Number of Utterances	Force Aligned Utterances	Successful Force Aligned Percentage
Test	1680	1561	92.92%
Train	4620	4231	91.58%
Total	6300	5792	91.94%

The next step of the evaluation is to compare the time markers created by the SAPI Force Alignment tool with the default time markers that are provided in TIMIT corpus. As mentioned before, these time markers are compared based on the time markers of TIMIT corpus subtract by the time markers by the SAPI Force Alignment's result. Table shows the interpretation of the values from obtain the differences of the two time markers.

Table 7: Interpretation of the time difference values.

Time Difference	Interpretation of Time Difference Value
Time Start	• **Positive value:** Force alignment time marker starts earlier than TIMIT time marker. • **Negative value:** TIMIT time marker start earlier than force alignment time marker. • **Zero value:** Both time markers start at the same time.
Time End	• **Positive value:** Force alignment time marker ends earlier than TIMIT time marker. • **Negative value:** TIMIT time marker ends earlier than force alignment

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

	time marker. • **Zero value:** Both time markers end at the same time.
Time Difference	• **Positive value:** The word duration marked by TIMIT is longer than the duration of the same word marked by force alignment • **Negative value:** The word duration marked by TIMIT is shorter than the duration of the same word marked by force alignment. • **Zero value:** Both word durations are the same.

To compare the time markers even further, the TIMIT Corpus Browser tool can export the information into the CSV file. The contents of this file is then loaded into Microsoft Excel and used for drawing histograms as shown in Figure 25, Figure 26, and Figure 27. Based on these histograms, the most frequency difference values of the time markers and word durations are located between -50 to 50 milliseconds and the largest frequency is at 0. Some extreme differences values turned out to be that the time markers of either from TIMIT corpus or SAPI Force Alignment's results have included the silence in them.

Once the individual word has been played by TIMIT Corpus Browsers, the sounds of the word from both sources are barely noticeable by human's ears. Hence, the results show that the time markers from SAPI Force Alignment's results are reasonable and the number of miss-alignments is tolerable.

Figure 25: Start time difference between TIMIT and SAPI force alignment.

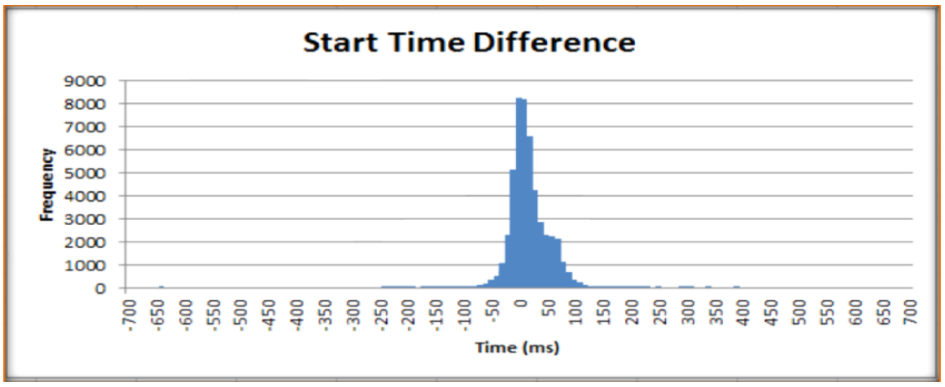


Figure 26: End time difference between TIMIT and SAPI force alignment.

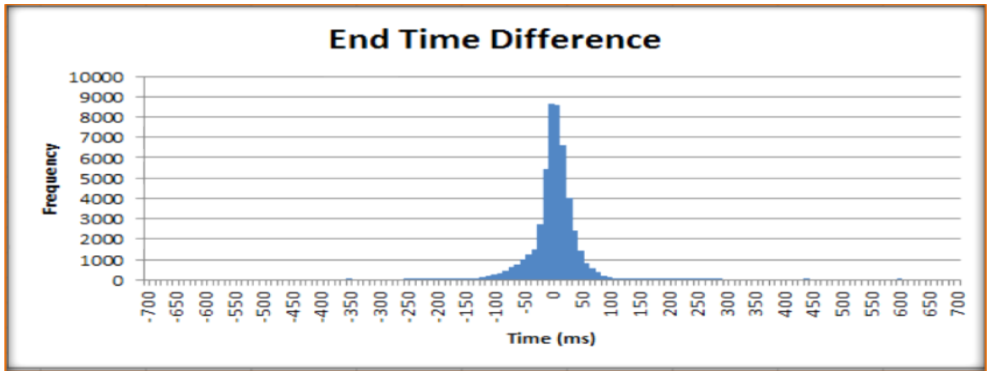
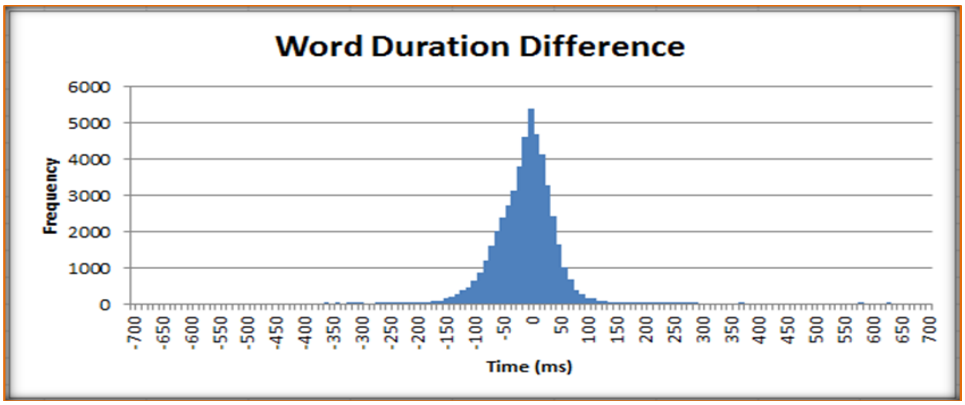


Figure 27: Word duration difference between TIMIT and SAPI force alignment.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy



ACOUSTIC MODEL EVALUATION

To evaluate the accuracy of any acoustic model it should be used on testing set. The recognition result will be compared with the corresponding transcriptions that come with the corpus. For the quality comparison between each model, the two error rates are chosen. One is a sentence error rate (SER), which computed by Equation . Another one is a word error rate (WER), which computed by Equation .

Several acoustic models are created by using utterances from a corpus generated from the TV series called the Office Season 1 Episode 1 to 6 (Daniels, 2005). The main reason why this series is being chosen is because there are plenty of dialogues without much background music throughout each episode. The results of the evaluation for several set of utterances are shown on Table . Note that for the initial testing, there is no adjustment made in the parameters of Sphinx Train's configuration, except the pointers to the input files. TIMIT's training set is used for creating an acoustic model and then evaluated by TIMIT's testing set to establish the baseline for comparison. Hence, all other acoustic models are also evaluated by TIMIT's testing set as well to ensure the fairness in the evaluation process.

Equation 6: Formula for computing sentence error rate.

$$SER = \frac{Total\ Number\ of\ Sentences - Number\ of\ Recognized\ Sentences}{Total\ Number\ of\ Sentences} * 100\%$$

$$= \frac{Number\ of\ Sentences\ that\ Cannot\ Be\ Recognized\ Correctly}{Total\ Number\ of\ Sentences} * 100\%$$

Equation 7: Formula for computing word error rate.

$$WER = \frac{Total\ Number\ of\ Words - Number\ of\ Recognized\ Words}{Total\ Number\ of\ Words} * 100\%$$

$$= \frac{Number\ of\ Words\ that\ Cannot\ Be\ Recognized\ Correctly}{Total\ Number\ of\ Words} * 100\%$$

The original attempt on generating a corpus from a DVD is to separate the training set and testing set by the ratio 75% to 25% respectively. The model created from the force aligned

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

training set (utterances that cannot be force aligned are excluded from the training) of the Office Season 1 has the highest error rate. The first hypothesis is that the size of the corpus maybe too small. So the second trial is to train an acoustic model with all Office Season 1 (both force aligned training and testing sets combined). Turns out the error rate goes down after that. This result shows that a larger the corpus size is the better acoustic model.

Table 8: Evaluation of acoustic models using default training parameter.

Acoustic Models #Den = 8, #Tied State =1000 (Default)	SER (Erroneous Sentences/ All Sentences)	WER (Erroneous Words/All Words)
TIMIT Training Set	42.1% (707/1680)	26.9% (3906/14527)

Next evaluation is to check whether the trimming utterances with force alignment would reduce the error rate. First, all the default utterances from Office Season 1 are used for training the third acoustic model. Turns out the model performed better than the models that only use force aligned utterances (again, the default has more utterances). Next step is to use the force aligned utterances combined with those that cannot be forced aligned, which are excluded from the acoustic model created from force aligned utterances only. The result from this model turns out to perform even better than the model trained by default utterances, which may have some transcriptions that do not exactly match with their utterances. This result has proved that the more utterances match with their transcription also will generate better acoustic model. However, the model with the least error rates still does not perform as good as TIMIT's training set. It is assumed that this is due to the fact that the parameters in Sphinx Train are not yet adjusted to be suitable with a specific corpus.

Based on CMU Sphinx Wiki, the two main parameters that required for reducing error rates are the following:

- **$CFG_FINAL_NUM_DENSITIES** – The final numbers of densities, which has to be set in 2 bit shifting: 2, 4, 8, 16, 32, 64, etc…
- **$CFG_N_TIED_STATES** – The total numbers of shared state distributions. The default value is at 1000.

The values of these two parameters are varied to much the data of a corpus. For example, those parameters will depend on a total numbers of words or a size of particular corpus (CMU Sphinx, 2010).

The acoustic model from the force aligned utterances combined with non-force aligned from the Office Season 1 Corpus is being chosen for the next test. This test uses global optimal search of the two parameters (**$CFG_FINAL_NUM_DENSITIES** and **$CFG_N_TIED_STATES**). The result from Table shows that the optimal parameters for this particular corpus are 8 for final number of densities and 200 for numbers of tied states.

To make the comparison even fairer, the acoustic model created from TIMIT's training set also has its two training parameters adjusted as well. The results from Table 9 shows that given 16 final numbers of densities and 1000 numbers of tied states yield the lowest error rates. This test shows that each corpus has different optimal values for the training parameters.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Table 9: Evaluation of the model with different training parameters.

Acoustic Models Created from Office Season 1 with Force Aligned Utterances and Non Force Aligned Utterances		Sentence Error Rate (SER)	Word Error Rate (WER)
$CFG_FINAL_NUM_DENSITIES	$CFG_N_TIED_STATES		
4	200	58.4%	31.1%
8	100	100%	111.1%
8	200	51.9%	26.4%
8	500	67.6%	37.2%
8	1000	85.5%	50.2%
8	2000	98.6%	73.8%
8	4000	99.8%	88.5%
16	4000	100%	96.7%
16	6000	100%	97.9%

After the optimal acoustic models from the Office Season 1 corpus and TIMIT corpus have been obtained, the Office Season 1 corpus has much better performance improvement from its original set up. However, the error rates are still higher than TIMIT's training set when its training parameters are adjusted.

Table 8: Model from TIMIT's training set with different training parameters.

Acoustic Models Created from the Training Set of TIMIT Corpus		Sentence Error Rate (SER)	Word Error Rate (WER)
$CFG_FINAL_NUM_DENSITIES	$CFG_N_TIED_STATES		
8	200	49.1%	37.2%
8	500	41.5%	28.9%
8	1000	42.1%	26.9%
8	2000	44.9%	27.4%
16	500	40.5%	27.8%
16	1000	40.2%	26.0%
16	2000	48.3%	29.6%

In extension to previous test, two more acoustic models are introduced. The first acoustic model is created from the corpus of the Office Season 2 Episode 1 to 6. The result from the training with the optimal training parameters yields 52.1% SER and 26.2% WER, which are still higher than the model trained by TIMIT's training set. Another model was created from the Office Season 1 and 2 combined. The acoustic model created from the Office 1 and 2 corpora perform better than the model created from the training set of TIMIT corpus when used for evaluating the TIMIT's testing set as shown in 9, with 38.6% SER and 20.2% WER.

Table 9: Comparison of acoustic model evaluation results.

Acoustic Model	# of Utterances	SER	WER

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Office Season 1 (E1 – E6) Corpus	2089	51.9%	26.4%
Office Season 2 (E1 - E6) Corpus	2187	52.1%	26.2%
Office Season 2 (E8 - E13) Corpus	2591	41.5%	20.6%
Office Season 2 (E1 – E6 & E8 – E13) Corpus	4778	41.2%	20.4%
Training Set of TIMIT Corpus	4620	40.2%	26.0%
Office Season 1 and 2 Corpus	6867	38.8%	18.5%

In summary, these are the following factors, which will help improving the quality of the acoustic model trained with a corpus generated from a DVD of a movie or TV series:

1. A size of a corpus – a larger the corpus the better its performance. Can be improved by adding more utterances into that corpus.
2. Accuracy of utterances and their transcriptions – using SAPI Force Alignment tool to trim some of the utterances that are successfully force aligned will help improve the quality of the model.
3. Adjust the training parameters in Sphinx Train – These numbers are varied based on the characteristics of the corpus (i.e. utterance size, total words, duration of each utterance).

CONCLUSION

A need for obtaining natural and inexpensive speech corpora for prosodic analysis in in general for any speech recognition had inspired the research for the data collection system from DVDs of movies and TV series. The C# .NET Framework application Data Collection Toolkit was developed to facilitate this research.

For the evaluation of the force alignment, the time markers of individual words generated from SAPI Force Alignment have shown to be very accurate when they are being compared to the time markers of the TIMIT corpus. After careful examination of the data (minimum at -637 milliseconds and maximum at 622 milliseconds), the outliers were due to that fact that time markers have included some silence as part of the word. Hence, when the sound from the word is played by using the time markers from two sources, they are barely noticeable by human's ears.

For acoustic model training, based on the evaluation using TIMIT's testing set, the acoustic models created from DVD corpus can perform better than the model created from the default TIMIT's training set, given that the size of the DVD corpus is large enough. The test results also show that if utterances are trimmed to exactly correspond to their transcriptions, an acoustic model created from them also performs better than the one created from the same utterances without trimming.

Ultimately, the concepts of the data collection system and Data Collection Toolkit have shown to be useful and versatile for creating speech corpora from DVDS of movies and TV series. Not only take less time for tedious task, it also is free given that the DVDs are already obtained. The

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

speech corpora obtained from this system can be used for any research and development in speech recognition and any other research that required speech utterances and/or transcriptions.

FUTURE WORK

Using SAPI for performing a force alignment is simple and efficient due being part of .NET framework. However, it contains certain parameters that cannot be controlled by a user, being Microsoft's proprietary technology, such as selecting a language model or set the threshold used for determining whether the utterances are successfully force aligned or not. Hence the accuracy and recognition rate of the force alignment process cannot be improved any further.

The solution to this issue is to develop an alternative speech recognition engine to perform this task instead. The CMU Sphinx also has an ability to do force alignment. However, it will require significant amount of time and extensive knowledge in programming to develop a module that can integrate it into Data Collection Toolkit partially since CMU Sphinx Toolkit is developed in C (SphinxTrain, Sphinx 2 & 3, PocketSphinx) and JAVA (Sphinx 4).

However, given enough time and manpower to rewrite the entire Data Collection Toolkit in JAVA, then it may be possible to switch SAPI Force Alignment with CMU Sphinx 4. In that case, then the toolkit will be able to run on multiple platforms, which will make this tool to be more versatile and broaden the scope of the users and developers who are interested in generating speech corpus from DVDs of movies and TV series.

Once the Data Collection Toolkit is rewritten into JAVA, then it will be easily to portable into the Linux environment. Hence, it is possible to integrate the RelEx into the toolkit and create a more robust way to handle the language analysis. This optional part of the data collection system is used for prosodic analysis.

The data collection system can be expanded further by adding video data collection. This could be useful for the study that requires feature extraction from image sequences such as training a model that can perform gesture recognition and any pattern recognition tasks relating to image data.

REFERENCES

AoAMedia.Com. (2009). AoA Audio Extractor.

CMU Sphinx. (2010). *CMU Sphinx: Open Source Toolkit for Speech Recognition*. Retrieved 8 13, 2010, from http://cmusphinx.sourceforge.net/

Daniels, G. (Director). (2005). *The Office Season 1* [Motion Picture].

Garofolo, John S., et al. (1993). TIMIT Acoustic-Phonetic Continuous Speech Corpus. Philadelphia, Pennsylvania, USA.

Hemming, C., & Lassi, M. (2010). *Copyright and the Web as Corpus.* Retrieved October 27, 2010, from Swedish National Graduate School of Language Technology: http://hemming.se/gslt/copyrightHemmingLassi.pdf

  ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Këpuska, V., & Klein, T. (2009). A Novel Wake-Up-Word Speech Recognition System, Wake-Up-Word Recognition Task, Technology and Evaluation. *Nonlinear Analysis: Theory, Methods & Applications* .

Këpuska, V., & Shih, C. (2010). Prosodic Analysis of Alerting and Referential context of Sentinel Words. Orlando: AIPR.

Këpuska, V., Gurbuz, S., Rodriguez, W., Fiore, S., Carstens, D., Converse, P., & Metcalf, D. (2008). uC: Ubiquitous Collaboration Platform for Multimodal Team Interaction Support. *Journal of International Technology and Information Managment*, 17(3/4), 263-284.

LDC, University of Pennsylvania. (2009, October 19). *Top Ten Corpora*. Retrieved September 22, 2010, from Linguistic Data Consortium: http://www.ldc.upenn.edu/Catalog/topten.jsp

Milam Aiken, M. P. (2009). Machine Translation in a Multilingual Electronic Meeting. *Journal of International Technology and Information Management , 18* (3/4), p. 395-409.

MSDN. (2010). *Microsoft Speech API 5.3*. Retrieved 8 7, 2010, from MSDN: http://msdn.microsoft.com/en-us/library/ms723627%28VS.85%29.aspx

NCH Software. (2010). Switch Audio File Converter Software.

OpenCog Wiki. (2010). *RelEx Dependency Relationship Extractor*. Retrieved September 13, 2010, from OpenCog Wiki: http://wiki.opencog.org/w/RelEx

Ramdhan, R., & Beharry, X. (2009). Movie Transcript Parser.

Rudnicky, A. (2010). *lmtool*. Retrieved Oct 1, 2010, from Sphinx Knowledge Base Tools: http://www.speech.cs.cmu.edu/tools/lmtool.html

Sarhan M Musa, E. U.-A. (2010). Statistical Analysis of VoDSL Technology for the Efficiency of Listening Quality of 640k/640k. *Journal of International Technology and Information Management , 19* (1), 97-113.

Sastraputera, R. (2009). *Discriminating alerting and referential context from prosodic features.* Melbourne: Florida Institute of Technology.

Shih, C.-T. (2009). *Investigation of Prosodic Features for Wake-Up-Word Speech Recognition Task.* Melbourne: Florida Institute of Technology.

Susan K Lippert, P. M. (2007). Personal Data Collection via the Internet: The Role of Privacy Sensitivity and Technology Trust. *Journal of International Technology and Information Management , 16* (1), 17-33.

Wang, J. (2005). Inference-Guiding for Intelligent Agents. *Journal of International Technology and Information Management , Volume 14* (Number 2).

Zuggy, T. V. (2009). SubRip 1.20/1.50b: DVD Subtitles Ripper.

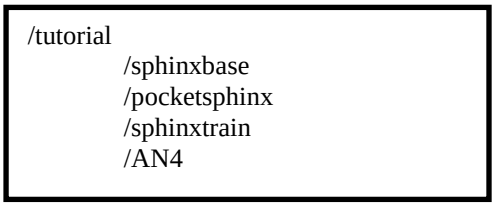
  ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

Appendix A: CMU Sphinx Installtion Tutorial

This appendix section describes the steps required for installing and running CMU Sphinx's Pocketsphinx and SphinxTrain toolkits to create a training model based on the online CMU Sphinx online wiki under "Training Acoustic Model for CMU Sphinx" section (CMU Sphinx, 2010). Note that this installation guide aims for running the Sphinx under a Linux Ubuntu environment.

1. Log into Ubuntu.
2. Go to the website http://CMU Sphinx.sourceforge.net/wiki/download/.
3. Download the following releases: Sphinxbase, Pocketsphinx, and Sphinxtrain.
4. Go to the website http://www.speech.cs.cmu.edu/databases/an4/ .
5. Download the following audio data: NIST's Sphere audio (.sph) format (64 M).
6. Create a folder "tutorial" on a known location in Ubuntu (i.e. on the desktop). This is a root folder for the project.
7. Extract sphinxbase, pocketsphinx, and sphinxtrain compressed files into /tutorial/ folder.
8. Rename of the sub folders from step 7 by removing the version tag. For example, rename sphinxbase-0.6.1 to sphinxbase.
9. Extract AN4 data base file and rename the folder to /AN4/ under /tutorial/ folder.
10. The project structure should be as in
11.
12.
13. **Figure A**after steps 1 to 9 have been followed.

Figure A. Project Structure for CMU Sphinx Tutorial.

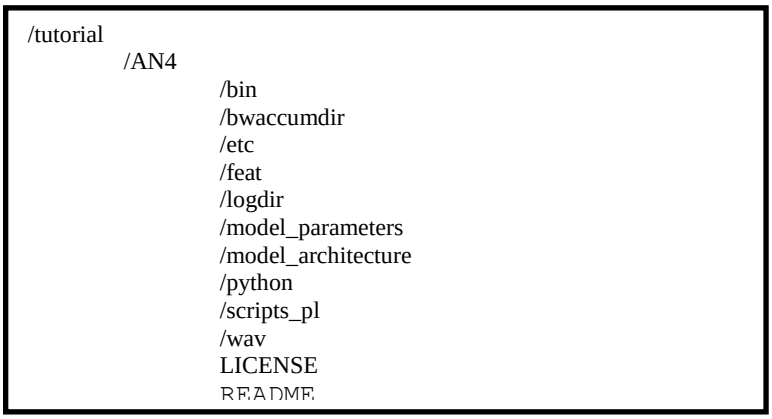


14. Go inside /tutorial/sphinxbase folder.
15. Run sudo ./configure; make; make install. This will configure the installation and install sphinxbase.
16. Get out of sphinxbase folder.
17. Repeat step 11 to 13, but go into folder pocketsphinx and sphinxtrain instead.
18. Go to folder /tutorial/AN4 folder.
19. Enter sudo /tutorial/sphinxtrain/scripts_pl/setup_SphinxTrain.pl –task AN4 to setup the sphinxtrain scripts for AN4 database.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

20. Enter sudo /tutorial/pocketsphinx/scripts/setup_sphinx.pl –task AN4 to setup the pocketsphinx scripts for AN4 database.
21. Inside folder /tutorial/AN4/ should have the contents as in Figure A-1.

Figure A-1: Contents Inside /tutorial/AN4 Folder.



22. If the files inside etc folder are created under Windows platform, they will need to have /r carriage removed before the further processes can be done. Ubuntu is capable to run a script as in
23. Figure A-2 to perform a batch removal of the /r carriage return.

Figure A-2: Ubuntu script for removing /r carriage from /etc folder.

```
for f in etc/*; do tr -d '\r' < $f > $f.new; mv $f.new $f; done
```

24. While still inside the folder /AN4/ enter the following command lines to start the training from sphinxtrain:
    ./scripts_pl/make_feats –ctl etc/an4_train.fileids
    ./scripts_pl/make_feats –ctl etc/an4_test.fileids
    ./scripts_pl/RunAll.pl
25. Waiting until SphinxTrain to finish generating features and training an acoustic model.
26. Measure the accuracy by running the following scripts:

    ./scripts_pl/decode/slave.pl

27. To use this trained acoustic model, run PocketSphinx with following parameter: pocketsphinx_continuous -hmm <your_new_model_folder> -lm your_lm -dict your_dict.

Appendix B: PERL SCRIPT FOR FIXING TRANSCRIPT

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

It is likely that some transcripts that come from the subtitle may require some preprocessing before they can be used for acoustic model training. Fixing just one transcript file can be done simply using text editor software. However, fixing a large number of transcript files can be a rigorous task. This Perl script is designed to help a user to add, edit, or delete transcript from a file.

Figure B: Perl code for transcription editing Part 1.

```
#=============== Beginning of the Code==================
# -*-Perl-*-
#Maintained by Pattarapong Rojanasthien
#Modified from Dr. Këpuska's createTrans.pl
#Last Revision: 10/19/2010

use File::Basename;
$line = join " ", @AGRV;

#Verify if a script is being used with proper arguments.

if (@ARGV < 1) {
        print STDERR "Usage: $0 Transcript newTranscript\n";
        print "[$#ARGV] $line\n";
        exit;
}

#Specify the file
$input_file = shift;
$output_file = shift;

#Open the file and read data - Die with grace if it fails
open (FILE, "<$input_file") or die "Can't open $file: $!\n";

@lines = <FILE>;
close FILE;
```

Figure B: Perl code for transcript editing part 2.

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

```
open (STDOUT, ">$output_file") or die "Can't open $file: $!\n";

#Walk through lines, make changes in each transcript
for ( @lines ) {
        #set what to change below in format:
        #       s/original_string/new_string/g
        #remove "."
        s/\.//g;
        #remove "!"
        s/\!//g;
        #remove "-"
        s/\-//g;
        #remove "/r" carriage
        s/\r//g;
        print;
}

#Finish up
close STDOUT;
print(stderr"\n------------------------------\n");
printf(stderr"Input: %s.\n",$input_file);
printf(stderr"Output: %s.\n",$output_file);
print(stderr"Transcript has been processed\n");
print(stderr"------------------------------\n");
exit;
#================= End of the Code ======================
```

APPENDIX C: BACKGROUND OF TIMIT CORPUS

Texas Instruments and Massachusetts Institute of Technology (TIMIT) corpus of read speech provides a set of speech data. It is intended to be used for acoustic research, development, and evaluation of an automatic speech recognition system. TIMIT corpus consists of recording sounds from 630 speakers in eight American English dialects as shown on Table C, each reading ten phonetically rich sentences (Garofolo, et al., 1993).

Table C: Statistics of speakers in TIMIT corpus categorize by dialect regions.

Dialect Region		#Male	#Female	Total
ID	Region			
DR1	New England	31	18	49
DR2	Northern	71	31	102
DR3	North Midland	79	23	102
DR4	South Midland	69	31	100
DR5	Southern	62	36	98
DR6	New York City	30	16	46
DR7	Western	74	26	100
DR8	Army Brat	22	11	33
Total		438	192	63

ISSN: 1543-5962-Printed Copy ISSN: 1941-6679-On-line Copy

TIMIT corpus is organized in a structure as shown in Figure C. It contains two different data sets: training and testing. Each data set contains sub folders, which categorize the utterances by dialects. Each utterance contains the following files:

- **mfc** file is Mel-Cepstral Frequency Coefficient, normally used for acoustic model training.
- **PHN** and **PHN_SP** files are the list of phones and their time markers.
- **TXT** file contains the time markers for a whole sentence.
- **WAV** is a wave file of an utterance. Note that this wave file is stored in mono 16-bit 16 KHz NIST format. In order to use it on standard media players, this file will need to be converted into Microsoft WAVE format first.
- **WRD** file contains words and their time markers.

Figure C: Folder structure of TIMIT corpus.