LDC Forced Aligner

Xiaoyi Ma
Linguistic Data Consortium
3600 Market St. Suite 810
Philadelphia, PA 19104
E-mail: xma@ldc.upenn.edu

Abstract

This paper describes the LDC forced aligner which is designed to align audio and transcripts. Unlike existing forced aligners, LDC forced aligner can align partially transcribed audio files, and also audio files with large chunks of non-speech segments, such as noise, music, silence etc, by inserting optional wildcard phoneme sequences between sentence or paragraph boundaries. Based on the HTK tool kit, LDC forced aligner can align audio and transcript on sentence or word level. This paper also reports its usage on English and Mandarin Chinese data.

Keywords: speech, transcript, forced alignment

1. Introduction

Aligned audio and transcript is a valuable source for many HLT applications, including speech recognition, text-to-speech synthesis, speech indexing, etc. Alignment can be done on different levels, such as utterance, word or phoneme level. Manual alignment of audio and transcript is very expensive and time consuming, so with a few exceptions, most aligned audio-transcript corpora were created by automatic aligners, some with manual validation and adjustment performed afterwards.

Existing aligners require that 1) audio files have been completely transcribed; 2) there are no long periods of non-speech regions, such as silence, music, background noise. These requirements are difficult to meet in some cases, for example, most close captioning of TV broadcast isn't complete, i.e. part of a program may not have been transcribed. In some other cases, the requirement is impossible to meet because the audio file contains non-speech signals, such as noise, music, and silence (most existing aligners can handle short silence well, but not silence over 10 seconds), because these non-speech signals cannot be transcribed using a phoneme set.

With the existing aligners, aligning audio and its incomplete transcript will have to be done in two steps. In the first step, one splits the audio and transcript into paired trunks so that each alignable audio trunk has a complete transcript. Then as the second step, each paired chunks is aligned automatically. The first step has to be done manually, and it involves annotators listening to the audio file and identifying the parts that have been transcribed, then mark the boundary on the audio file and the transcript file. It's a labour intensive process.

The LDC aligner was designed to align audio files with incomplete transcripts, which is often the case for broadcast transcripts since music, commercials, noise and sometimes parts of programs are not transcribed.

The rest of the paper is laid out as follows: section 2 describes the overall structure of the aligner; section 3 contains details of the tool; section 4 describes experiments and some preliminary results; section 5 concludes the paper.

2. Overall Structure

The LDC forced aligner is based on the HTK toolkit[1]. However, unlike other aligners, which use HTK's forced alignment feature (HVite with –a option) to align audio and transcripts, the LDC aligner runs HTK's recognition function (HVite without –a option) and get the alignment as a by-product. This allows us to inject optional wildcard phoneme sequences, which can match untranscribed audio of arbitrary length, into the word lattice that HTK uses for recognition.

In pattern matching with regular expression, one can use wildcard to match zero or more occurrences of certain symbols. For example, AB*C matches any sequence that starts with an A and ends with a C and with zero or more Bs in between.

The wildcard phoneme sequences that we used are similar in nature. Its purpose is to match any non-speech (noise, music) and untranscribed speech in a given audio file. Because of the wide range of acoustic properties of noise, music, and speech with and without background music and noise, we need more than one wildcard phoneme sequence. Through experiments we created 24 such wildcard sequences for each language (English and Chinese) so the tool has the best chance of matching the wildcard with untranscribed signals.

With the 24 wildcards created above and a wildcard for silence, we have 25 wildcards in total.

During force alignment, we first segment the transcript by paragraph, or sentence, or utterance, depending on the source of the corpus. Then we insert the wildcards between those transcript segments. The resulting grammar looks like the following:

sentence1 {WILDCARD1 | WILDCARD2 |…|WILDCARD25} sentence2 {WILDCARD1 | WILDCARD2 |…|WILDCARD25} sentence3 {WILDCARD1 | WILDCARD2 |…|WILDCARD25}

where curly bracket indicates zero or more occurrences of the patterns inside, according to HTK's grammar syntax.
HTK will then take the grammar as input and generate a word lattice which will be used in speech recognition.
We configured HTK in such a way that it not only outputs the words it identifies (which we already know), but also the time boundaries of each word, which is what we want.
As the final step, we take the HTK recognition output, discard the entries for wildcards and return the remaining words and their time stamps as output. The script can be configured to return word or sentence level alignment.

3. Wildcard Phoneme Sequences

The 24 wildcard sequences were created as follows for each language:

1) we trained a speech recognition system on broadcast news data that have speech with or without non-speech background; Non-speech segments, such as coughing, laughing, music and other noises, were excluded from the training data;
2) we divided our development data into four categories: speech, music, noise, and speech with non-speech background;
3) we run the speech recognizer obtained in 1) on each category of 2);
4) we selected the most frequent six triphones from each of the four categories;

4. Implementation

4.1. Pronouncing Dictionary

For English, we use the CMU Pronouncing dictionary[2], which contains pronunciations for over 125,000 words. The version used in this work is CMUDict v0.6. The CMU phoneme set has 39 phonemes (see Table 1), for which the vowels may carry lexical stresses. The lexical stresses, however, were not used in this work. Some words may have multiple pronunciations, which HTK does allow and can choose the one that gives the highest score during the training or recognition process.
As for OOV words, some of them can be dealt with by a transducer train on CMUDict, but others, mostly numbers, dates, etc., require a rule based system so possible alternative pronunciations can be generated.

Phoneme	Example	Translation
AA	odd	AA D
AE	at	AE T
AH	hut	HH AH T
AO	ought	AO T
AW	cow	K AW
AY	hide	HH AY D
B	be	B IY
CH	cheese	CH IY Z
D	dee	D IY
DH	thee	DH IY
EH	Ed	EH D
ER	hurt	HH ER T
EY	ate	EY T
F	fee	F IY
G	green	G R IY N
HH	he	HH IY
IH	it	IH T
IY	eat	IY T
JH	gee	JH IY
K	key	K IY
L	lee	L IY
M	me	M IY
N	knee	N IY
NG	ping	P IH NG
OW	oat	OW T
OY	toy	T OY
P	pee	P IY
R	read	R IY D
S	sea	S IY
SH	she	SH IY
T	tea	T IY
TH	theta	TH EY T AH
UH	hood	HH UH D
UW	two	T UW
V	vee	V IY
W	we	W IY
Y	yield	Y IY L D
Z	zee	Z IY
ZH	seizure	S IY ZH ER

Table 1 English Phoneme Set

The Mandarin Chinese pronouncing dictionary contains pronunciations of 7,333 Chinese characters. Some of the characters have multiple pronunciations. Mandarin Chinese is a tonal language, but tones are not marked in the pronouncing dictionary, nor used in training of the acoustic models.
The Mandarin Chinese phoneme set contains a total of 38 phonemes (see Table 2).
Modern Chinese have a limited number of characters. The pronouncing dictionary contains all characters in the GB2312 encoding standard. It has less characters than the more inclusive GB12345 standard, but it does

contain all the characters we see in the training, development and subsequent testing. Thus OOV words wasn't an issue for Chinese in our case. There are, however, English words (mostly names) in our training and development data. Sentences or paragraphs containing such words were excluded from training and testing.

It's easy to expand the current pronouncing dictionary to include characters in GB12345 standard, if it becomes necessary.

Phoneme	Example	Translation
>	多	d w >
@	哀	@ y
&	策	c &
%	吃	C %
a	八	b a
b	八	b a
c	擦	c a
C	插	C a
d	代	d @ y
e	杯	b e y
E	鞭	b y E n
f	发	f a
g	哥	g &
h	很	h & n
i	闭	b i
I	词	c I
j	积	j i
k	哭	k u
l	拉	l a
m	马	m a
n	鞍	@ n
N	肮	a N
o	博	b w o
p	品	p i n
q	其	q i
r	然	r @ n
R	而	R
s	色	s &
S	山	S & n
t	谈	t @ n
u	毒	d u
U	居	j U
w	保	b a w
W	学	x W E
x	习	x i
y	北	b e y
z	早	z a w
Z	张	Z a N

Table 2 Chinese Phoneme Set

4.2. Acoustic Models

We trained acoustic models for English and Mandarin Chinese. The training data were both from LDC hub4 broadcast news corpora [3][4]. To create clean HMM training data, we filtered out the segments that contain partial words, mispronounced words, interjections, non-lexemes, idiosyncratic words, unintelligible speech and speech in foreign languages.

This gives us about 67 hours of English training data and 63 hours of Mandarin Chinese training data in 8-bit 16KHz NIST sphere file format.

The coding uses Mel Frequency Cepstral Coefficients (MFCCs) as the energy component, the frame period is 10msec. The FFT uses a Hamming window and the signal has the first order preemphasis applied using a coefficient of 0.97. The filterbank has 26 channels and outputs 12 MFCC coefficients. Energy normalization was performed on recorded audio files.

Monophone HMMs were trained and they consist of 3-state left-right with no skips.

4.3. Pre- and post-processing

There are two issues with the English forced aligner that we solve by pre- and post-processing: 1) OOV words; 2) preserving the surface form, including capitalization and punctuations that are connected to the word. The first issue is obvious and easy to understand, the second needs some explanation. It's desirable that the output of the forced aligner, that is words with start and end time, has the same spelling as the original text in order to make validation and down stream processes easier. This is not possible with HTK, because HTK is not case sensitive.

There are three kinds of OOV words. The first kind consists of proper names, such as *Anajay*, *Abagil*, and newly coined words, such as *vacationary, facebook,* which are not included in the pronouncing dictionary. The second kind consists of mostly numbers, dates, times, symbols (currency, percentage), which are not spelled out. The third kind involves punctuations, for example *Bush's* (policy), where *Bush's* become an OOV if we tokenize by space.

To deal with the first kind of OOV words, we trained a transducer on the CMU dictionary which takes a word in its surface form and output its pronunciation in CMU phoneme set. Acronyms such as *NIST* and *IBM* receive special treatments. We provide two alternative pronunciations for every acronym, one pronouncing it as a word, the other as letters.

To deal with the second kind of OOV words, which mostly involves numbers and symbols, we first use a rule-based system to spell out these words, then we look

up the resulting words from the pronouncing dictionary. Some examples of the words covered by this approach:

Numbers: 19104, 2010
Symbols: $, £, €, %, &
Date and time: 03/05/2010, Mar. 5, 7:30pm

Again, we try to provide all possible readings of these words. For example, the possible readings of 2010 include:

Twenty ten
Two thousand ten
Two thousand and ten
Two zero one zero
Two oh one oh

The third kind of OOV words that involve capitalization and punctuation are first parsed into multiple parts, each containing only alphanumeric characters, then the pronunciations of these parts are looked up or generated separately. Finally the pronunciations of these parts are merged to form the pronunciation of the full word.
It's worth noting that during each run of the English forced aligner, CMU dictionary was only used as the seed dictionary. The transcript is parsed and a pronouncing dictionary is created dynamically, based on entries from the CMU dictionary and analysis of the OOV words.
To address the second issue, which is to preserve the spelling of the words in the transcript, we created a mapping table between the transcript words and their HTK uppercase counterpart. After HTK is run, the mapping table is then used to convert the aligned words or sentences into its original spelling.

5. Experiments

To test the forced aligners, we selected 30 hours of broadcast news each for English and Chinese. All these audio files contain some music, advertisement that cannot or have not been transcribed. The length of audio files range from 10 to 60 minutes, with the majority around 30 minutes.
To make the task more difficult for the aligner, we randomly removed parts of the transcripts. The audio durations without transcripts range from 10.5 to 35.8 per cent.
The transcripts do not contain any time stamps on any level.
Two experiments were run to test the forced aligner's performance on sentence and word level alignment respectively. For experiment on sentence level alignment, we first break the transcripts into sentence before feeding them into the forced aligner.
To create a baseline system, we first create word and sentence level alignment of the segments that do have accurate transcripts. This involves extracting each audio segment individually, aligning it to its corresponding transcript using the same aligner, then adding an offset to the timestamps to reflect their real positions in the original speech file.
The output of the forced aligners is then compared to the baseline result. We computed the average offset between the timestamps of forced aligned results and those of the baseline system. For sentence level alignment, the average offset is 15ms for English and 18ms for Chinese. For word level alignment, the average offset is 10ms for English and 12ms for Chinese.
Both results indicate the forced aligner's results on incomplete transcripts are comparable to those from complete transcript.
We also manually checked the word and sentence level alignment of randomly selected files, and the findings are consistent with the finding above.
There are about one percent of files that failed the forced alignment, i.e the aligner doesn't produce any output for these files. These files were excluded from the statistics above. A closer look finds that these files usually have very poor acoustic qualities.

6. Conclusion and Future Work

We introduced the LDC forced aligner which completely eliminates manual annotation when aligning audio with incomplete transcripts. The results demonstrate that the aligner's results are as good as the results from files with manual sentence/utterance level alignment.
The LDC forced aligner has been used to align about 300 hours of English broadcast news audio and transcripts and 250 hours of Mandarin Chinese data. All transcripts are incomplete transcription of the original audio. Some of these transcripts were closed captioning, some were transcripts prepared before a broadcast, some were outsourced transcription with instructions to transcribe only parts of a file. None of the transcripts had any timestamps on them on sentence, utterance, or paragraph level. The forced aligner has proven to be a very cheap and efficient way of aligning these kinds of audios and transcripts.
We intend to expand this work to other languages.

7. References

[1] http://htk.eng.cam.ac.uk/
[2] http://www.speech.cs.cmu.edu/cgi-bin/cmudict
[3]http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC98S71
[4]http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC98S73