ALISA

An **A**utomatic **L**ightly **S**upervised Speech Segmentation and **A**lignment Tool

Version 1.5 - October 2014

Adriana Stan
Yoshitaka Mamiya
Oliver Watts
Peter Bell
Junichi Yamagishi
Rob Clark
Simon King

Contents

ACKNOWLEDGEMENT

The development of this tool has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 287678 ”Simple^{4}All”.

1 Introduction

One of the most important problems of text-to-speech synthesis (TTS) or automatic speech recognition (ASR) systems is the lack of speech data. Lots of time and computation effort are invested into recording or transcribing speech from multiple sources. Hence, the readily available data, such as speech with approximate ortographic text transcripts should be exploited. There are theoretically an unlimited number of online resources which could be used, including audiobooks, podcasts, video streams with subtitles etc. in most of the world's languages. But for all of these, the correspondence between audio and text is not always a 100% accurate. This means that, taking the case of an audiobook, the reader might have omitted, inserted or substituted words or sequences of words, leading to the inability of a forced alignment tool for example, to achieve the necessary accuracy.

The aim of this tool is not to correctly transcribe the entire speech resource available, but to make the most out of them. Using no previous knowledge of the language, nor any additional resources, and with minimal user intervention, ALISA can correctly transcribe around 70% of the initial speech data.

A more elaborate description of the tool can be found in the following articles:

- Adriana Stan, Peter BELL, Simon KING, *A grapheme-based method for automatic alignment of speech and text data*, In Proc. IEEE Workshop on Spoken Language Technology, Miami, Florida, USA, December 2012
- Yoshitaka Mamiya, Junichi Yamagishi, Oliver Watts, Robert A.J. Clark, Simon King and Adriana STAN, *Lightly Supervised GMM VAD to use Audiobook for Speech Synthesiser*, In Proc. ICASSP, May 2013
- Adriana Stan, Peter Bell, Junichi Yamagishi, Simon King, *Lightly Supervised Discriminative Training of Grapheme Models for Improved Sentence-level Alignment of Speech and Text Data*, In Proc. Interspeech, Lyon, France, August 2013
- A. Stan, O. Watts, Y. Mamiya, M. Giurgiu, R. A. J. Clark, J. Yamagishi, S. King, *TUNDRA: A Multilingual Corpus of Found Data for TTS Research Created with Light Supervision*, In Proc. Interspeech, Lyon, France, 2013.

2 Prerequisites

2.1 Software Prerequisites

HTK 3.4.1 - HMM Toolkit – can be downloaded from `http://htk.eng.cam.ac.uk/`

Python v 2.7.2 - any version should be ok, as no specific libraries are used.

Edinburgh Speech Tools - for *ch_wave* in the forced alignment tool

sox - SOund eXchange – for sampling frequency conversion `http://sox.sourceforge.net/`. You can use some other tool if you would like to.

SPTK - Speech Signal Processing Toolkit `http://sp-tk.sourceforge.net/` - used in the segmentation (VAD) step. SPTK 3.8 has some API changes, so we recommend using an older version.

Wavesurfer or Praat - for marking silence segments in the segmentation step.

NLTK - for a better sentence tokenizer - `http://nltk.org/`. After installing, also add the Punkt Tokenizer Model by running *nltk.download()* and selecting Models/punkt.

2.2 Data Preparation

In order to use the tool you need one or several speech files and their full, although **not fully accurate** text transcript.

Text - should be in a single **.txt* file, using Unix newline characters. Some light normalisation can be performed, but it is not necessary. Stripping the parts of text which you suspect the reader might not have read (e.g. table of contents, footnotes etc.) will increase the performance.

Speech - should be in **.wav* format, single channel. For the alignment part, speech should be segmented into shorter files. This can be done using the segmentation (Voice Activity Detection - VAD) algorithms provided.

VERY IMPORTANT NOTE - before starting the alignment, please make sure that the text corresponds more or less to the length of the audio you are using. For example, if you are running a trial of the tool, and you only use half of the audio data, remove the corresponding text as well.

3 Running the Tool

Steps for obtaining the aligned speech and their transcripts:

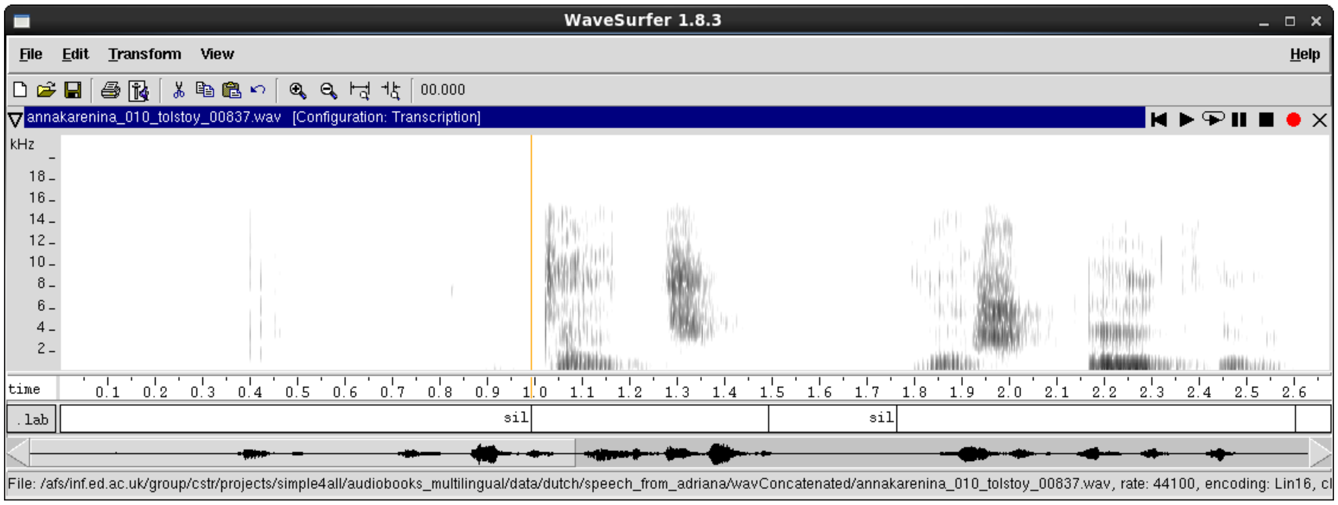
NOTE: The *testData/* folder contains a sample minicorpus on which you can do a test run of ALISA. You will find the raw wav files, a 10 minute file which is annotated with silence (aTrampAbroad-50UttsSil.wav,lab) and the corresponding text for the raw wav files (*aTrampAbroad.txt*). If you do not alter the silence marking, then the transcription for the initial training data (i.e. *aTrampAbroad-initialTextDataBrut*) should be correct. But do a light check against the *initialWav16/* folder after running the segmentation.

STEP 1

Copy the speech files into a folder of choice (e.g. *aTrampAbroadData/rawWavs*), making sure that they have a common prefix, for example: *chapter*01.wav, *chapter*02.wav, etc. Also, prepare the text and make sure it has a simple name and a **.txt* extension. For example: *aTrampAbroad.txt*.

STEP 2

Cut the first aprox. 10 minutes[1] of speech from the entire data. Using a tool of choice, e.g. Wavesurfer or Praat, manually annotate the silence parts which **separate the sentences**[2]. For reference on labelling see Fig. 3.0a. One important thing in labelling the silence is to make sure not to include any speech in the silence segments and vice-versa. Otherwise the GMM-based VAD will fail.



File: /afs/inf.ed.ac.uk/group/cstr/projects/simple4all/audiobooks_multilingual/data/dutch/speech_from_adriana/wavConcatenated/annakarenina_010_tolstoy_00837.wav, rate: 44100, encoding: Lin16, cl

Figure 3.0a: How to label the silence segments in Wavesurfer

The result should be a **.wav* file, named for example *initialWav.wav* and an associated label file, for example *initialWav.lab*.

Convert the label file to the following format:

[1]The longer the annotated segment, the better the performance, although 10 minutes should be enough for a kick start

[2]Please note that not all the silence segments need to be marked, as this step tries to segment the entire speech data into sentence-length chunks.

```
#
0.197500 sil
2.4575
2.942489 sil
6.482489
7.032489 sil
8.212489
8.582489 sil
10.112489
10.442489 sil
...
```

You can use the *scripts/convertWavesurferLabels.py* script to convert a Wavesurfer type label to the one described above.

STEP 3

Next step is to run the segmentation algorithm. There is a configuration file named *config.template* which you should edit according to the data you are using. Each of the following steps requires you to fill in the appropriate variables. For the segmentation step, you should complete the following info:

INPUTFOLDER - path to the folder which contains the **raw speech files**.

OUTPUT - path to the folder where you would like to output the alignment from ALISA[3].

INWAV - path to the 10 minute speech file for which you marked the inter-sentence silences segments.

LABELFILE - path to the actual silence label file.

SPTKDIR - path to the SPTK bin/ folder.

MINLEN - the result of the VAD can sometimes output very short speech files, which are very hard to align. Therefore, you can set this variable to concatenate consecutive speech files, so that they have the specified minimum duration (e.g. MINLEN=5 means all files will be at least 5 seconds long). If you set it to 0, no concatenation takes place.

Once you set the above variables, you can run **./runVAD.sh config.template**. The segmented speech will be written to *$OUTPUT/wav/*. A sampling frequency conversion will also take place. This is because the alignment algorithm needs 16kHz files–output to *$OUTPUT/wav16/*. The final aligned output will however maintain the original sampling frequency.

STEP 4

The previous step will additionally create a folder named **$OUTPUT/initialWav16/** which contains either the segmented first 10 minutes of speech which you previously annotated with silence, for the GMM-based VAD, or the first 50 segmented utterances of speech for the other methods. You will need to manually transcribe these speech files. You should create a text file and accurately transcribe the speech, **one utterance per line, and in the order that they are in the *initialWav16/* folder**. Do not worry about punctuation or uppercase letters, these will be dealt with in the *./runTP.sh* script.

HINT: To ease the transcription, you can copy the text from the book and insert newlines after each speech file, delete the additional newlines, correct the reading errors, and add the text that is not in the book. This would save you the trouble of writing the entire text, and should also prevent spelling errors.

STEP 5

[3]These two variables are general ones, and are used by the text processor and aligner as well

Run the **./runTP.sh config.template** script to get the list of graphemes, a dictionary and a bigram list. You need to set the following variables:

BOOK - path to the book text file (e.g. */work/aTrampAbroad/aTrampAbroad.txt*)

INITIALTEXT - path to the raw transcript for the initial speech data (e.g. */work/aTrampAbroad/aTrampAbroad-rawInitial*)

SUPERVISED – if you have a supervised lexicon for the language you are aligning, you can set this option to 1, and the tool will run with phones instead of graphemes. Also, please add the following entry to the dictionary: *BGHMM bghmm* in the appropriate spot.

SUPERVISEDDICT – absolute path to your supervised lexicon.

LISTOFPHONES – absolute path to the list of phones used in your lexicon

FROMTPSTEP and TOTPSTEP - the steps that should be performed (from 1 to 5):

STEP 1 - Sentence Segmentation;

STEP 2 - Create grapheme list;

STEP 3 - Replace non-standard characters & create bigram list;

STEP 4 - Create HTK Dictionary;

STEP 5 - Prepare initial text data.

You will be prompted to enter ASCII friendly characters for the non-standard graphemes. (e.g. in Romanian, *ş* becomes *sh*), while rare characters that do not belong to that language should be replaced by an in-alphabet one (e.g. in English, *å* becomes *a*).

STEP 6

This step takes quite a long time and it is the most computationally expensive. Run the **./runAlign.sh config.template** script to get the confident aligned files. You need to set the following variables:

FROMSTEP and TOSTEP - selects which steps of the acoustic model training should be performed:

STEP 0 - Initialise workspace;

STEP 1 - Initial models trained on 10 minutes of speech;

STEP 2 - First iteration of mono-grapheme models;

STEP 3 - MMI trained mono-grapheme models;

STEP 4 - Tri-grapheme re-estimation.

NAIVE – the absolute path to the directory where you are running the tool (e.g. */home/work/alisa/*)

HLMTOOLSDIR – the absolute path to the HLMTools bin directory

HTK_BIN – HTKTools path. Make sure you are running version 3.4.1, which is required for the discriminative training step.

ESTDIR – Edinburgh Speech Tools path.

NWIN – the length of the text window for the approximate text step. This is highly dependent on the length of the text. You can leave it set to 1300, which is ok for a text of about 150,000 words. If the aligned percentage seems to be too low, please increase the legth of this window.

MARKER - this is a common prefix found at the beginning of all the speech files – for example 'chp' if the audio is named *chp_00*.wav*. It is used for some of the speech file processing steps, so please make sure that the files are named accordingly.[4]

USEMMI - this option when set to 1 builds discriminatively trained MMI acoustic models on top of the grapheme ML ones before doing tri-grapheme re-estimation.

[4]Hint: In Unix systems you can use the *rename* command to rename all files in a folder. For example: rename 's/^/chp_/' *.wav.

STEP 7

The final step is to run **./runPostProcessinAndCleanup.sh config.template**. This step performs some additional post-processing, such as sentence boundary correction, punctuation restoration, and also cleans up the temporary files. You can set the following options:

SENTB – a variable which sets the sentence boundary correction. If you think that the speech might be segmented at sentence boundary for a large percent of the data, you can use this to make some correction of sentence initial and ending word insertions and deletions.

FINALCONCAT - this option specifies if in the post-processing, the speech files which concantenated consist a sentence from the book text, should actually be concatenated. For example, if the VAD segmented the speech into ”Mary had” and ”a little lamb”, and in the booktext you can find the sentence ”Mary had a little lamb.”, then the two speech files are concatenated.

CLEAN - this option should be set to 1 only when you are satisfied with the results of the alignment, as it removes any intermediate and temporary files created by ALISA. If you plan on repeating some of the alignment steps, you should leave it set to 0.

The **results** will be output in the **$output/ALIGNED/** folder and will contain

– the aligned speech files at the original sampling frequency in the **output/wav/**;

– a speech transcription for all the files, **output/speech_transcript.txt**;

– a folder with speech transcription for each speech file **output/txt/**, without punctuation;

– a folder with speech transcription for each speech file **output/txtWithPunctuation/**, with punctuation. At the end of the alignment, the confident files are compared with the original text, and the original punctuation is restored.

If you set the FINALCONCAT variable to 1, then the concatenated data for all the above folders will also be available.

Please send any errors or suggestions to:

`Adriana.Stan@com.utcluj.ro` or `owatts@inf.ed.ac.uk`