The WaveSurfer Automatic Speech Recognition Plugin

Giampiero Salvi and Niklas Vanhainen
KTH, School of Computer Science and Communication,
Department of Speech Music and Hearing, Stockholm, Sweden
{giampi, niklasva}@kth.se

Abstract

This paper presents a plugin that adds automatic speech recognition (ASR) functionality to the WaveSurfer sound manipulation and visualisation program. The plugin allows the user to run continuous speech recognition on spoken utterances, or to align an already available orthographic transcription to the spoken material. The plugin is distributed as free software and is based on free resources, namely the Julius speech recognition engine and a number of freely available ASR resources for different languages. Among these are the acoustic and language models we have created for Swedish using the NST database.

Keywords: Automatic Speech Recognition, Free Software, WaveSurfer

1. Introduction

Automatic Speech Recognition (ASR) is becoming an important part of our lives, both as a viable alternative for humans-computer interaction, but also as a tool for linguistics and speech research. In many cases, however, it is troublesome, even in the language and speech communities, to have easy access to ASR resources. On the one hand, commercial systems are often too expensive and not flexible enough for researchers. On the other hand, free ASR software often lacks high quality resources such as acoustic and language models for the specific languages and requires expertise that linguists and speech researchers cannot afford.

In this paper we describe a plugin for the popular sound manipulation and visualisation program WaveSurfer[1] (Sjölander and Beskow, 2000) that attempts to solve the above problems. Firstly, the plugin, as the WaveSurfer program, is free software. We chose to release the plugin under the GPL License. Secondly, it makes the functionality of the Julius[2] recognition engine available to the user in a simple way, without the need to learn how Julius works. Finally, we use ASR resources that are freely available, and we promote the development of such resources for languages with a relatively small number of speakers, such as Scandinavian languages.

To support this effort, we created an ASR dedicated website[3] where we will collect all the resources that we develop or that become available to us. We recently trained acoustic and language models for large vocabulary speech recognition in Swedish (Vanhainen and Salvi, 2014) that are available for download at that site and that are included in the plugin.

Although still in a early stage of development, this plugin should enable linguists and speech researchers to take advantage of the possibilities offered by speech recognition. It should, e.g., simplify the task of annotating speech material as well as provide a useful tool for educational purposes.

[1]http://www.speech.kth.se/wavesurfer/
[2]http://julius.sourceforge.jp/
[3]http://www.speech.kth.se/asr

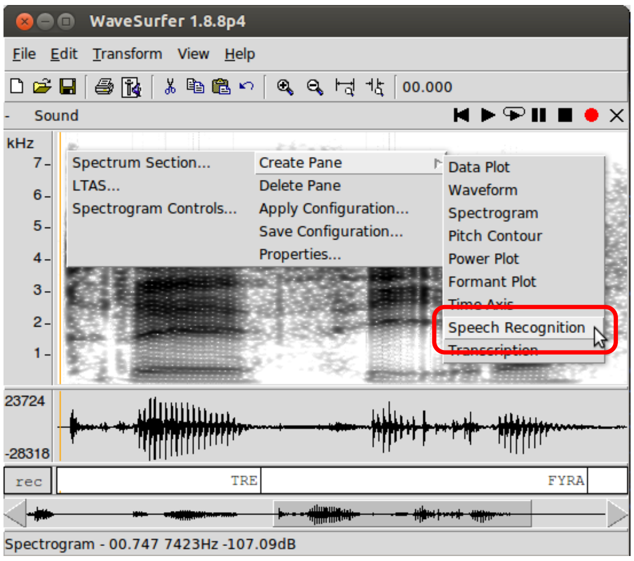


Figure 1: The Speech Recognition plugin adds an entry in the `Create Pane` menu.

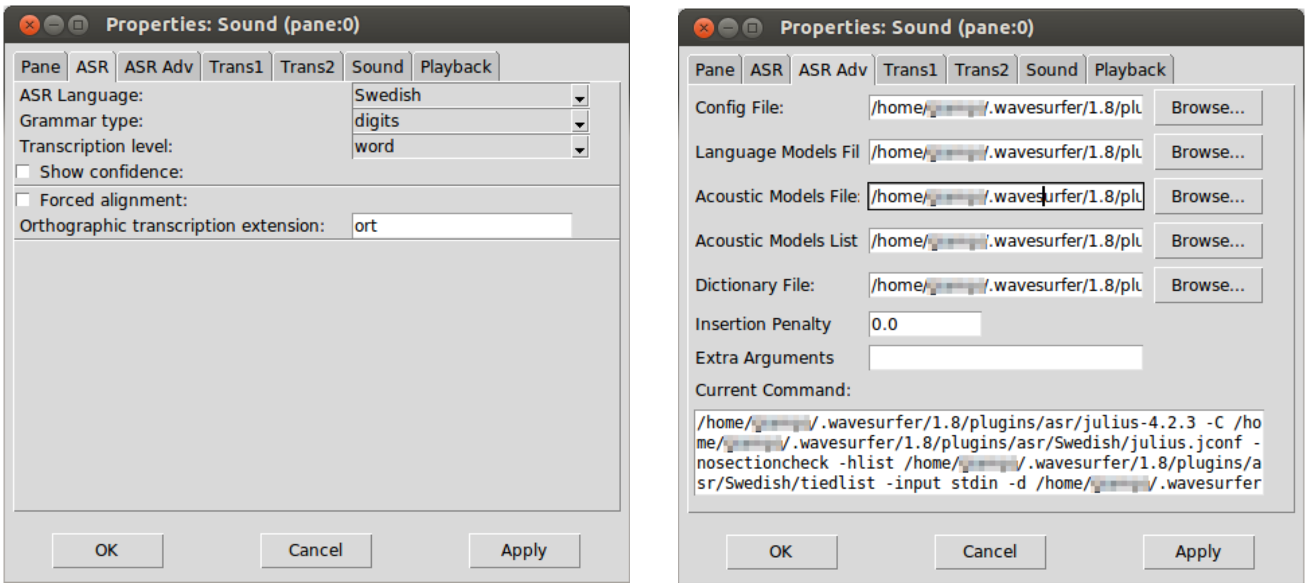
1.1. Related Work

Many free software packages are available that implement general ASR functionality. The most popular, besides the already mentioned Julius, are CMU Sphinx[4] (Lamere et al., 2003), and Kaldi[5] (Povey et al., 2011). Although these provide the fundamental building blocks to perform speech recognition, they usually require a certain degree of expertise from the user for setting up and running.

A few software packages provide tools for speech annotation and speech analysis in a user friendly manner. Probably the best known are Praat[6] (Boersma, 2002), and WaveSurfer. However, at the time of writing, we are not aware of any ASR functionality implemented for neither of these software packages.

Recently, Bigi (2012) developed a tool for phonetic segmentation of speech that is available on Linux, Mac OSX and MS Windows[7]. The tool, as our plugin, is dependent

[4]http://cmusphinx.sourceforge.net/
[5]http://kaldi.sourceforge.net/
[6]http://www.fon.hum.uva.nl/praat/
[7]http://aune.lpl-aix.fr/~bigi/sppas/



ASR Language: Swedish
Grammar type: digits
Transcription level: word
Show confidence:
Forced alignment:
Orthographic transcription extension: ort

Config File: /home/[redacted]/.wavesurfer/1.8/plu
Language Models Fil /home/[redacted]/.wavesurfer/1.8/plu
Acoustic Models File: /home/[redacted]/.wavesurfer/1.8/plu
Acoustic Models List /home/[redacted]/.wavesurfer/1.8/plu
Dictionary File: /home/[redacted]/.wavesurfer/1.8/plu
Insertion Penalty 0.0
Extra Arguments
Current Command:

```
/home/[redacted]/.wavesurfer/1.8/plugins/asr/julius-4.2.3 -C /ho
me/[redacted]/.wavesurfer/1.8/plugins/asr/Swedish/julius.jconf -
nosectioncheck -hlist /home/[redacted]/.wavesurfer/1.8/plugins/a
sr/Swedish/tiedlist -input stdin -d /home/[redacted]/.wavesurfer
```

Figure 2: The properties window for the automatic speech recognition plugin. Left: basic properties, right: advanced properties.

on the Julius speech recogniser and is used to facilitate the annotation of speech material at the phonetic level.
The goal of our plugin is to provide user-friendly, but highly configurable speech recognition functionality in a software package that is already equipped with many other speech related functions. In this way, the user can take advantage of the power and flexibility of popular software packages, all integrated in the same tool.

2. The Plugin

WaveSurfer's functionality is organised in *panes*. For each sound object, the user can add different panes with the desired functions, ranging from waveform display, to spectrogram, multiple transcriptions, and so forth.
Once installed, the Speech Recognition plugin adds an entry called `Speech Recognition` in the `Create Pane` menu (see Figure 1).
The Speech Recognition pane is a transcription pane with embedded ASR functionality. If we choose to create a Speech Recognition pane, the plugin will use the default settings to attempt to recognise the content of the current sound object and will display the results in the transcription.

2.1. Options and Functionality

By right clicking on this special transcription pane, and selecting `Properties`, the user has access to the recognition parameters (see Figure 2). Two tabs are added to WaveSurfer property window. The first, entitled `ASR` (Figure 2 left), contains basic settings for the recogniser. The second, entitled `ASR Adv` (Figure 2 right), contains advanced settings.
The first option in the basic settings allow the user to select among the installed languages. The plugin has been designed in a modular way, so every time a new language is installed, a new entry will automatically be created in the `ASR Language` selector.

The `Grammar type` option lets the user choose among few example grammars that were provided by the language package. Examples can be `digits`, `1kwords` for 1000 word grammar, and so on. These examples are provided as a starting point to test the recogniser. Depending on the recogniser performance in the specific language, it may be necessary to constrain the recognition grammar to the specific task the user is interested in. At the moment of writing, defining new grammars is not possible through the plugin, but we release scripts that may help users achieving this offline and then using the custom grammar files with the help of the advanced options.
The `Transcription level` option lets the user select if the recognition output should be displayed at the word or phoneme level.
If the `Show confidence` checkbox is activated, the plugin will display the confidence of the recognizer for each word by varying the font colour. Black fonts correspond to high confidence, whereas red fonts to low confidence. This options is based on the normalised confidence measure called "cmscore1" in Julius and is therefore only available for word-level transcriptions. An example of words recognised with different levels of confidence is shown in Figure 3.
The next section in the `ASR` property tab, defines the operation of the recogniser. If the `Forced alignment` checkbox is selected, an orthographic transcription of the speech file (with equal file name and default extension `ort`) will be loaded, and the recogniser will attempt to align this transcription in time to the speech file. The extension of the optional orthographic transcription can also be configured. If the checkbox is deselected, on the other hand, the recogniser will use a predefined grammar to run free recognition.
The advanced option tab (Figure 2 right), contains configuration parameters that allow the user to change any option of the recogniser. In order to properly change any of these configurations, however, the user needs to be familiar with

the functionality of the Julius recogniser in details. The full command that is run by the plugin is also displayed to simplify the task of debugging problems whenever the user decides to change any advanced option.
The design of these two property tabs was motivated by the attempt to keep the plugin simple for non expert users, but to give at the same time full control for the expert users. By means of the advanced options, users can even use their own language resources, provided they are able to save them in a format that is compatible with Julius.
Finally, all the options can be saved using WaveSurfer `Save Configuration...` standard command, making it easy to recover them at a later stage.

3. Installation

The simplest way to install plugins in WaveSurfer is to copy the relevant files into the WaveSurfer preference directory. In order to do this, WaveSurfer has to be already installed, and executed at least once. Note that the plugin installation files come with a binary version of the Julius recogniser, so it is not necessary to install Julius separately. The following sections give instructions depending on the architecture, however you should always refer to the instructions coming with the latest version of the plugin:

GNU Linux:

1. download the plugin package including the word `linux` in the file name
2. extract the files into the directory `~/.wavesurfer/<version>/plugins/`

Where ~ is an alias for the user's home directory, and `<version>` is the WaveSurfer major version number (`1.8` at the time of writing)

Mac OS X:

1. download the plugin package including the word `osx` in the file name
2. same as point 2 for Linux

MS Windows:

1. download the plugin package including the word `win` in the file name
2. extract the files into the directory `$HOME\.wavesurfer\<version>\plugins\`

Where `$HOME` points to the user's home directory, and `<version>` is the WaveSurfer major version number (`1.8` at the time of writing)

4. Language Support

4.1. How to add new languages

As mentioned in Section 2.1., the plugin architecture makes it easy to add resources for different languages whenever these are available in Julius format. The simple way to add support for a language is to store the relevant files in a directory under `~/.wavesurfer/<version>/plugins/asr/`. The directory name should correspond to the language name or any string the users wishes to be displayed in the `ASR Language` selector (Figure 2 left). This directory should also contain the file `lang.conf` specifying all the default parameters for the recogniser. Any Julius compatible files can be included in the new language directory, provided that the file `lang.conf` contains enough information for the plugin to understand how to run Julius properly. Probably the easiest way to achieve this is by simply defining a Julius configuration file with all the proper parameters, and then pointing to that file in the `lang.conf` file.
Note that the same goal can be achieved by defining a standard WaveSurfer configuration file with all the proper definitions, and storing the language specific files in any location on your filesystem. The advantage of using the `lang.conf` method is that the new language will automatically show up in the language selection option.

4.2. Currently available languages

In general, freely available resources for automatic speech recognition (ASR) are scarce. Julius and CMU Sphinx come with acoustic and language models for the English language trained on the Voxforge corpus[8]. Unfortunately this corpus is rather limited if compared to the corpora used for commercial ASR systems. Other models, available for download, may not be freely distributed because they are trained on commercial corpora.
We are making an effort to provide high quality language resources that may be freely used by the community. Recently, we have trained acoustic and language models for Swedish (Vanhainen and Salvi, 2014) on the NST corpus. This corpus is freely distributed after bankruptcy of the company that collected it (Nordisk Språkteknologi, NST), and is comparable in size and quality with commercially used corpora. This makes Swedish, to our knowledge, the first language for which high-quality ASR resources are truly freely available.
Additionally, models for American English were trained by Keith Vertanen on the Wall Street Journal data set[9]. We converted these models to Julius format and packaged them for the plugin as well.
In the future, we plan to build acoustic and language models for Danish and Norwegian, for which similar NST corpora are available.

5. Plan for Future Versions

The plugin presented in this paper is at a functional, but early stage of development. In order to make this software useful for the widest possible audience, a number of improvements should be developed. In the following we will make a list of the ones we believe should have highest priority. However, we are aware of the fact that with free software, it is often the user community that decide in which direction development should go.

[8]http://www.voxforge.org/
[9]http://www.keithv.com/software/htk/us/

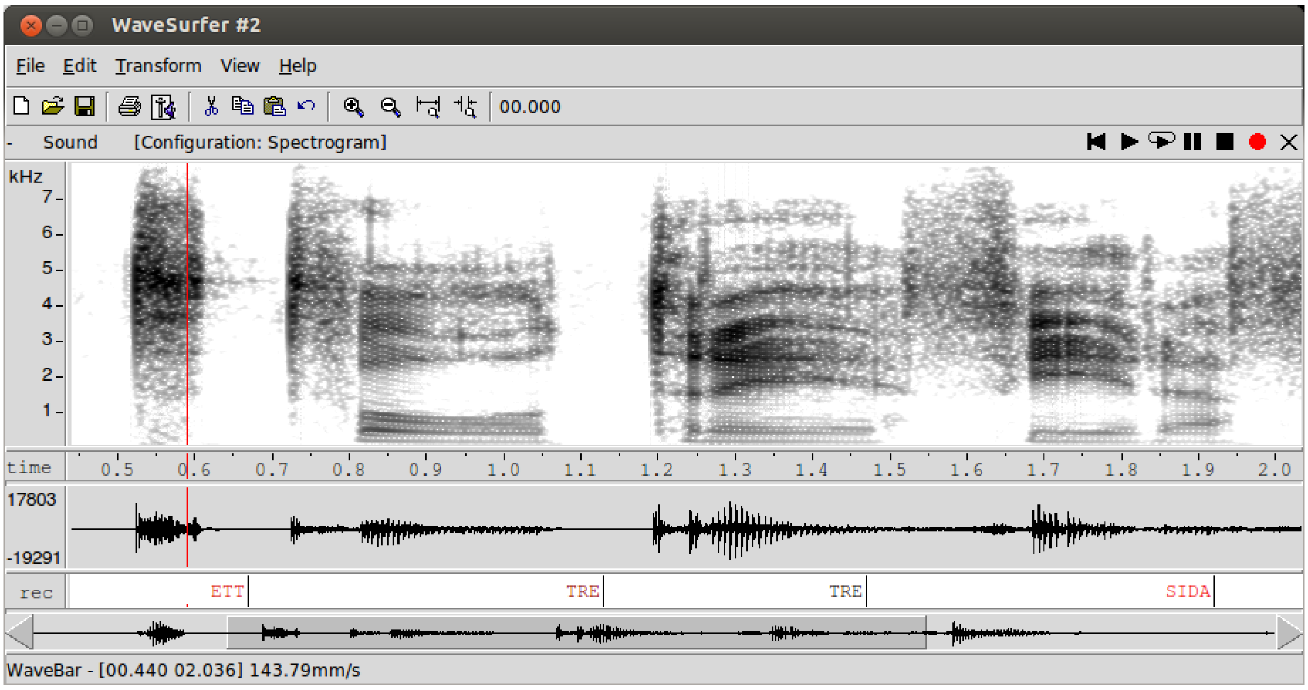

WaveSurfer #2
File Edit Transform View Help
00.000
Sound [Configuration: Spectrogram]
kHz
7
6
5
4
3
2
1
time 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0
17803
-19291
rec
ETT
TRE
TRE
SIDA
WaveBar - [00.440 02.036] 143.79mm/s

Figure 3: Example of automatically generated transcription for a speech utterance in Swedish. The confidence for each word is expressed by font colour in the labels. Black font corresponds to high confidence, red font to low confidence.

5.1. Installation

Although the plugin installation is fairly easy, it would be beneficial to take advantage preexisting software packaging systems available in some architectures. In Debian-derived GNU Linux distributions, for example, software can be conveniently packaged into `deb` packages with the possibility to define dependencies. The plugin package could be made dependent on the already existing Julius package, and the language specific files could be distributed as standard language packages for Julius. This would not only greatly simplify the installation process, but it would make sure that those resources are not duplicated on the computer disk and can be used for other applications as well as this specific plugin.

5.2. Language Models

Whereas acoustic models can usually be applied to different speech recognition tasks without modification, it is critical for an application such as this to allow the user to define the recognition grammar for the current task. Large vocabulary speech recognition without specific context has a level of performance that is not adequate for most applications. In the current version, changing the recognition grammar can be done through the advance options of the plugin, and with the help of some scripts that we provide. This however requires knowledge of the details of how Julius works, and is not accessible to all users. A user-friendly interface to introduce constraints to the list of words to be recognised, for example, would be beneficial.

5.3. Forced Alignment

The forced alignment option in the plugin is at the moment rudimentary. Two areas that are fundamental for forced alignment to work properly are:

1. handling out-of-vocabulary words
2. handling reduction and assimilation within and between words

Both these functions require language dependent knowledge that should be included in future releases of the plugin.

5.4. Adaptation

As with many tools that make use of speech technology, it is very important to adapt the recognition models to the situation they are applied to. In order to make this possible with minimum effort from the user, it would be beneficial to develop a number of extensions to the plugin.
Firstly it should be possible to easily add words and their pronunciations to the lexicon. It should be possible to correct recognition errors from the transcription pane, and at the same time inspect and correct the cause of these errors, e.g., by adding alternative pronunciations in the lexicon. It should be possible to adapt the acoustic models to the current data, based on the corrected transcriptions. The user should finally be able to upload the modifications to the models to our web site, making them available for future versions of the plugin.

6. Conclusion

The work presented in this paper is part of a greater effort to make speech technology publicly available, especially for languages with a relatively small number of native speakers and therefore a limited commercial appeal. We presented an extension to the WaveSurfer software that makes automatic speech recognition (ASR) available to users with no particular expertise in the field. This should allow speech researchers and linguists to take advantage of the possibilities offered by ASR technology. It will simplify the effort of transcribing speech material, and hopefully enable a wide range of possibilities in speech research.

7. References

Bigi, B. (2012). Sppas: a tool for the phonetic segmentation of speech. In Calzolari, N., Choukri, K., Declerck, T., Dogan, M. U., Maegaard, B., Mariani, J., Odijk, J., and Piperidis, S., editors, *LREC*, pages 1748–1755. European Language Resources Association (ELRA).

Boersma, P. (2002). Praat, a system for doing phonetics by computer. *Glot international*, 5(9/10):341–345.

Lamere, P., Kwok, P., Gouvea, E., Raj, B., Singh, R., Walker, W., Warmuth, M., and Wolf, P. (2003). The cmu sphinx-4 speech recognition system. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2003), Hong Kong*, pages 2–5. Citeseer.

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., et al. (2011). The kaldi speech recognition toolkit. In *Proc. ASRU*, pages 1–4.

Sjölander, K. and Beskow, J. (2000). Wavesurfer - an open source speech tool. In *Proc. of Interspeech*, pages 464–467.

Vanhainen, N. and Salvi, G. (2014). Free Acoustic and Language Models for Large Vocabulary Continuous Speech Recognition in Swedish. In Calzolari, N., Choukri, K., Declerck, T., Loftsson, H., Maegaard, B., Mariani, J., Odijk, J., and Piperidis, S., editors, *LREC*. European Language Resources Association (ELRA).