Speech Recognition Based on Open Source Speech Processing Software

Piotr Kłosowski, Adam Dustor, Jacek Izydorczyk, Jan Kotas, and Jacek Ślimok

Silesian University of Technology, Institute of Electronics
Akademicka Str. 16, 44-100 Gliwice, Poland
{piotr.klosowski,adam.dustor,jacek.izydorczyk}@polsl.pl,
{kotas.janek,jacek.slimok}@gmail.com
http://iele.polsl.pl

Abstract. Creating of speech recognition application requires advanced speech processing techniques realized by specialized speech processing software. It is very possible to improve the speech recognition research by using frameworks based on open source speech processing software. The article presents the possibility of using open source speech processing software to construct own speech recognition application.

Keywords: speech recognition, speech processing, open source software.

1 Introduction

Division of Telecommunication, a part of the Institute of Electronics and Faculty of Automatic Control, Electronics and Computer Science Silesian University of Technology, for many years has been specializing in advanced fields of telecommunication engineering. One of them is speech signal processing [1–4]. Main research areas on this field are: speech synthesis, speech recognition and speaker verification and identification.

Creating of speech recognition application requires advanced speech processing techniques realized by specialized speech processing software [5]. Which software should be used for speech recognition application development? There are many possibilities. To create own speech recognition application can be used:

1. programming languages such as C++, Java, Python, etc.,
2. high level commercial computing environment for implementation of speech recognition algorithms such as: MATLAB with Signal Processing Toolbox,
3. open source speech processing software.

Hypothesis can be formulated as follows: It is possible to improve the speech recognition research by using frameworks based on open source speech processing software. The article focuses on open source applications that can be used to construct their own speech recognition system.

A. Kwiecień, P. Gaj, and P. Stera (Eds.): CN 2014, CCIS 431, pp. 308–317, 2014.

2 Open Source Speech Processing Software for Speech Recognition

This section presents to the most frequently used speech processing open source software tools as a basis for testing and building your own speech recognition application.

2.1 CMU Sphinx

CMU Sphinx is a group of speech recognition systems, which have been developed at Carnegie Mellon University in Pittsburgh, Pennsylvania, United States. Sphinx consists of multiple software systems, not all of which are open source. First version of Sphinx, developed by Kai-Fu Lee, offered a system that demonstrates the feasibility of accurate, large-vocabulary speaker-independent, continuous speech recognition [6]. It has been later replaced by newer, better performing versions. Sphinx 2, the successor of the original, was the first Sphinx system to be released as open source. The biggest improvement was, apart from fewer recognition errors overall, the capability to handle much larger vocabulary size. For 5,000-word speaker-independent speech recognition, the recognition error rate has been reduced to 5 %. It is used in real-time speech recognition systems, however this project is no longer developed. Sphinx 3 is a system designed with accuracy in mind and therefore does not currently allow real-time speech recognition. It is, however, in active development and recent improvements made to its algorithms, as well as hardware performance growth have allowed to achieve near real-time recognition results. It adopts widely used continuous hidden Markov model. Together with SphinxTrain – an acoustic model trainer – it gives access to modeling techniques, such as MLLR (Maximum-Likelihood Linear Regression), LDA/MLLT (Linear Discriminant Analysis/Maximum Likelihood Linear Transform) or VTLN (Vocal Tract Length Normalization). Newest version of Sphinx is Sphinx 4. It is a rewritten Sphinx engine (entirely in Java), providing a more flexible framework for research in speech recognition. In addition to systems mentioned above, a lightweight version of Sphinx, named PockedSphinx, is currently in active development and provides access to speech recognition engine designed specifically for embedded devices, such as mobile phones. More details about the Sphinx project are available at: `http://cmusphinx.sourceforge.net`.

2.2 SPRACH

SPRACH is an abbreviation for Speech Recognition Algorithms for Connectionist Hybrids. It involves usage of HMM (Hidden Markov Models), ANN (Artificial Neural Networks), statistical inference in said networks, as well as hybrid HMM-ANN technology in order to further improve current research on continuous speech recognition. The project was developed across multiple universities in Europe and therefore one of its main goals was to adapt hybrid speech recognitions to languages other than English, French and Portuguese in particular. More details about the project are available at:
`http://www.icsi.berkeley.edu/~dpwe/projects/sprach/sprachcore.html`.

2.3 GMTK

GMTK is an acronym for Graphical Models Toolkit and a project that was developed by Prof. Jeff Bilmes, Richard Rogers and a number of other individuals at University of Washington, United States. GMTK is a feature-rich toolkit, allowing rapid prototyping of statistical models, by using DBM (Dynamic Graphical Models) and DBN (Dynamic Bayesian Networks) [7]. Among many of its features, one can specify the following: exact and approximate inference, dense, sparse and deterministic conditional probability tables, native support for ARPA backoff-based factors and factored language models, parameter sharing, gamma and beta distributions etc. [8]. In addition, it includes Markov chains of arbitrary orders, graph viewer in form of Graphical User Interface; supports multiple file formats; allows offline and online mode during parameter learning and prediction. More details about GMTK project are available at: `http://ssli.ee.washington.edu/~bilmes/gmtk/`.

2.4 SONIC

SONIC is a system developed by Bryan Pellom and Kadri Hacioglu at the University of Colorado, Boulder, United States. It is designed in order to allow development of new algorithms of continuous speech recognition. SONIC's main features include: phonetic aligner, phonetic decision tree acoustic trainer, core recognizer, speaker adaptation, live-mode recognition, voice activity detection, language portability, speech compression interface, as well as application programming interface (API) with examples [9]. It is worth mentioning, that SONIC has been successfully ported to over 15 languages. It is based on CDHMM (Continuous Density Hidden Markov Model). SONIC has been developed for 4 years until May 2005 and at the time of the last update it was still considered unfinished by the authors [10]. More details about SONIC project are available at: `http://www.bltek.com/virtual-teacher-side-menu/sonic.html`.

2.5 HTK

HTK is a toolkit designed for building HMM (hidden Markov models). This toolkit at its core is multi-purpose and may be used in order to model any time series, however it has been created with speech recognition in mind. HTK consists of two major tool sets. The first, namely HTK training tools, requires both: speech data and its transcription and is done in order to estimate parameters of a set of HMM. Once this stage is complete, HTK recognition tools can be used in order to transcribe unknown speech data [11]. An invaluable advantage of the described toolkit is its extensive documentation providing numerous examples of usage. More details about HTK project are available at: `http://htk.eng.cam.ac.uk/`.

2.6 ALISE

ALIZE is open source platform for biometrics authentication. It provides single engine for face and voice recognition. Project's goal is to provide access

to biometric technologies for industrial and academic usage. It consists of low-level API and high-level executables. Thanks to this attribute, ALIZE allows the user to quickly create speech recognition system, as well as provides tools for industrial voice processing applications. The project has been made open source, because it is believed that allowing broad scientific research on speech recognition algorithms results in quicker improvement of such systems, by making them more accurate and resistant to noise [12]. Project has been evaluated by NIST, SRE, RT and French ESTER and achieved very good results. It is written in C++ and allows multi-platform implementation, including a possible use in embedded devices. More details about the project are available at: `http://mistral.univ-avignon.fr/index_en.htm`.

2.7 SPRO

SPRO is an open source speech processing toolkit. It was designed for both: speaker and speech recognition. SPRO has been created for variable resolution spectral analysis, but it also supports classic mechanics used in speech processing. It provides runtime commands, as well as standard C library for implementing new algorithms and applications. After compilation, the user is given access to tools used for following purposes: filter-bank based speech analysis, linear predictive speech analysis, comparing streams, extracting speech parameters. The library provides additional signal processing functions, which can be used in custom speech recognition applications, such as FFT, LPC analysis and feature processing, such as lifter, CMS and variance normalization. SPRO is distributed under GNU Public License agreement. More details about the project are available at: `https://gforge.inria.fr/frs/index.php?group_id=532`.

2.8 Other Open Source Speech Processing Tools

Other open source speech processing tools that can be used to construct own speech recognition application are:

- AT&T FSM LibraryTM – Finite-State Machine Library
  `http://www2.research.att.com/~fsmtools/fsm/`
- LIBSVM – A Library for Support Vector Machines
  `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`
- SVMlight – Support Vector Machine
  `http://svmlight.joachims.org/`
- PVTK – Periodic Vector Toolkit
  `http://old-site.clsp.jhu.edu/ws2004/groups/ws04ldmk/PVTK.php`
- LAPACK – Linear Algebra PACKage
  `http://www.netlib.org/lapack/index.html`
- Standard Template Library
  `http://www.sgi.com/tech/stl/`
- MIT Finite State Toolkit
  `http://people.csail.mit.edu/ilh/fst/`

3 Examples of Use SPRO to Speech Features Extraction

This section presents to the examples of use open source speech processing tools called SPRO to speech features extraction as a basis for testing and building own speech recognition application. After compilation SPRO provides the following tools for speech recognition:

- `scompare` – tool dedicated to compare input streams tool,
- `scopy` – feature manipulation tool,
- `sfbank` – tool dedicated to filter-bank based speech analysis,
- `sfbcep` – tool dedicated to filter-bank based speech analysis,
- `slpc` – tool takes as input a waveform and output linear prediction derived features,
- `slpcep` – tool dedicated to linear predictive analysis of speech signals,
- `splp` – tool dedicated to linear predictive analysis of speech signals.

The following examples illustrate the use of SPRO tools to the speech feature extraction.

3.1 Extracting Speech Features Using Filter-Bank Analysis Tools

SPRO provides an implementation of filter-bank analysis – a spectral analysis method based on representing the signal spectrum by the log-energies at the output of a filter-bank, where the filters are overlapping band-pass filters spread along the frequency axis. This representation gives a approximation of the signal spectral shape while smoothing out the harmonic structure if any. When using variable resolution analysis, the central frequencies of the filters are determined so as to be evenly spread on the warped axis and all filters share the same bandwidth on the warped axis. This spectral analysis technique is also applied to Mel frequency warping, a very popular warping in speech analysis which mimics the spectral resolution of the human ear. The Mel warping is approximated by function [13]:

$$\mathrm{mel}(f) = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \quad . \tag{1}$$

Filter-bank analysis implementation uses triangular filters on the FFT module. The energy at the output of channel i is given by:

$$e_{_i} = \log \sum_{j=1}^{N} h_i(j) \cdot \|X(j)\| \tag{2}$$

where N is the FFT $length^2$ and h_i is the filter's frequency response as depicted above. The filter's response is a triangle centered at frequency f_i with bandwidth $[f_{i-1}, f_{i+1}]$, assuming the f_i's are the central frequencies of the filters determined according to the desired spectral warping.

The SPRO tools named `sfbank` and `sfbcep` are dedicated to filter-bank based speech analysis. The first filter-bank analysis tool `sfbank` takes as input a waveform and output filter-bank magnitude features. For each frame, the FFT is performed on the windowed signal, possibly after zero padding, and the magnitude is computed before being integrated using a triangular filter-bank.

The second SPRO tool `sfbcep` takes as input a waveform and output filter-bank derived cepstral features. The filter-bank processing is similar to what is done in `sfbank`. The cepstral coefficients are computed by DCT'ing the filter-bank log-magnitudes and possibly liftered. Optionally, the log-energy can be added to the feature vector. In `sfbcep`, the frame energy is calculated as the sum of the squared waveform samples after windowing. As for the magnitudes in the filter-bank, the log-energy are thresholded to keep them positive or null. The log-energies may be scaled to avoid differences between recordings. Finally, first and second order derivatives of the cepstral coefficients and of the logenergies can be appended to the feature vectors. When using delta features, the absolute log-energy can be suppressed using the `--no-static-energy` option.

```
$ sfbcep -F PCM16 -f 16000 input.wav output.mfc
```

The file with `.mfc` extention contains the result of the calculation in raw format. Example Bash script performs speech features extraction from all files on the list saved as `timit.lst` file is presented below:

```
#!/bin/bash

### List of file of speech database
fileslist="timit.lst"
speakerlist="speakers_all.txt"

### Deleting old feature files
rm -r ./timit/features

### Creating empty feature directories
mkdir ./timit/features
for dir in $(cat "$speakerlist")
    do
        mkdir "./timit/features/$dir"
    done

### SPRO features extraction
for file in $(cat $fileslist)
    do
        echo -n "Processing file : $file ... "
        inputfile="./timit/speech/$file.WAV"
        outputfile="./timit/features/$file.raw.mfc"
        ./SPRO/sfbcep -F PCM16 -f 16000 $inputfile $outputfile
        if (( $? )); then echo "ERROR"; exit; else echo "OK"; fi
    done
```

File `timit.lst` contains all processed speech files from TIMIT speech database. The TIMIT corpus of read speech is designed to provide speech data for acoustic-phonetic studies and for the development and evaluation of automatic speech recognition systems. TIMIT contains broadband recordings of 630 speakers of eight major dialects of American English, each reading ten phonetically rich sentences [14]. The TIMIT corpus includes time-aligned orthographic, phonetic and word transcriptions as well as a 16-bit, 16 kHz speech waveform file for each utterance. Corpus design was a joint effort among the Massachusetts Institute of Technology (MIT), SRI International (SRI) and Texas Instruments, Inc. (TI). The speech was recorded at TI, transcribed at MIT and verified and prepared for CD-ROM production by the National Institute of Standards and Technology (NIST). Sample contents of the file `timit.lst` is shown below:

```
speaker001/SA1
speaker001/SA2
speaker001/SI1565
speaker001/SI2195
speaker001/SI935
speaker002/SA1
speaker002/SA2
speaker002/SI1081
speaker002/SI1202
speaker002/SI1711
```

3.2 Extracting Speech Features Using LPC Analysis Tools

LPC (Linear Prediction Coding) is a popular speech coding analysis method which relies on a source/filter model if the speech production process. The vocal tract is modeled by an all-pole filter of order p whose response is given by [5]:

$$H(z) = \frac{1}{1 + \sum_{i=1}^{p} a_i z^{-i}} \quad . \tag{3}$$

The coefficients a_i are the prediction coefficients, obtained by minimizing the mean square prediction error. The minimization is implemented in SPRO using the auto-correlation method.

PLP (Perceptual Linear Prediction) is combination of filter-bank analysis and linear prediction to compute linear prediction coefficients on a perceptual spectrum [15]. The filter-bank power spectrum is filtered using an equal loudness curve and passed through a compression function $f(x) = x^{1/n}$ where usually $n = 3$, thus resulting in an auditory spectrum from which the autocorrelation is computed by inverse discrete Fourier transform. Linear prediction coefficients are then carried out as usual from the autocorrelation.

SPRO provides two different tools `slpc` and `slpcep` for linear predictive analysis of speech signals. The tool `slpc` takes as input a waveform and output linear prediction derived features. For each frame, the signal is windowed after pre-emphasis and the generalized correlation is computed and further used

to estimate the reflection and the prediction coefficients which can, in turn, be transformed into log area ratios or line spectrum frequencies. The default is to output the linear prediction coefficients however reflection coefficients can be obtained with the `--parcor` option, log-area ratios with `--lar` option and line spectrum pairs with the `--lsp` one. Optionally, the log-energy can be added to the feature vector. In `slpc`, the log-energy is taken as the linear prediction filter gain, which is also the variance of prediction error, and thresholded to be positive or null. The log-energies may be scaled to avoid differences between recordings using the `--scale-energy` option.

The tool `slpcep` takes as input a waveform and outputs cepstral coefficients derived from the linear prediction filter coefficients. The linear prediction processing steps are as in `slpc` and cepstral coefficients are computed from the linear prediction coefficients using the recursion previously described. The required number of cepstral coefficients must be less then or equal to the prediction order. As for `slpc`, the log-energy, taken as the gain of the linear prediction filter, can be added to the feature vectors. Mean and variance normalization of the static cepstral coefficients can be specified with the global `--cms` and `--normalize` options but do not apply to log-energies. The normalizations can be global or based on a sliding window whose length is specified with `--segment-length`. Finally, first and second order derivatives of the cepstral coefficients and of the log-energies can be appended to the feature vectors. When using delta features, the absolute log-energy can be suppressed using the `--no-static-energy` option.

The tool `splp` takes as input a waveform and outputs cepstral coefficients derived from a perceptual linear prediction analysis. Note that, although not explicitly mentioned in the program name, `splp` does output cepstral coefficients, not linear prediction coefficients. The LPC order must be less than or equal to the number of filters in the filter-bank while the number of cepstral coefficients must be less than or equal to the prediction order. The log-energy is taken from the frame waveform as in the filter-bank tools.

Example of extracting speech features using PSRO LPC analysis tools is presented below:

```
$ slpc sa.wav slpc.out
$ scopy -o ascii slpc.out slpc.txt
$ slpcep sa.wav slpcep.out
$ scopy -o ascii slpcep.out slpcep.txt
$ splp sa.wav splp.out
$ scopy -o ascii splp.out splp.txt
```

The files with `.txt` extentions contains the result of the calculation in ASCII format.

4 Summary

Table 1 presents comparison among presented open source speech processing tools. This paper and comparison can be helpful in choosing the right speech processing tool for a specific speech recognition application.

Table 1. The comparison presented open source speech processing tools

Name	Environment	Portability	Flexibility	Features
CMU Sphinx	C++ Java	Unix (Linux) Embedded sys.	high	model training, speech recognition framework
SPRACH	C++ Tcl/Tk Perl	Unix (Linux)	high	speech recognition framework
GMTK	C++ Tcl/Tk	Unix (Linux)	very high	probabilistic modeling framework
SONIC	C++ Tcl/Tk	UNIX (Linux, Solaris) Windows Mac OS	very high	continuous speech recognition
HTK	ANSI C	Unix (Linux) Windows	high	hidden Markov model toolkit
ALIZE	C++	Unix (Linux) Windows Embedded sys.	medium	speech processing and recognition framework
SPRO	C++	Unix (Linux) Windows Embedded sys.	medium	speech features extraction

The most important elements of each speech recognition system are speech features extraction and classification. The paper presents examples of the use open source speech processing software for speech features extraction. Similarly, the open source software can be used to test and create efficient classifiers that are the basis for designing effective speech recognition applications. For testing and construction of the various classifiers can be used e.g. the Hidden Markov Model Toolkit (HTK) [16] or open source platform for biometrics authentication called ALIZE [17]. Use of the open source speech processing software can significantly improve the construction and testing of modern speech recognition application.

Acknowledgements. This work was supported by The National Centre for Research and Development (www.ncbir.gov.pl) under Grant number POIG.01.03.01-24-107/12 (Innovative speaker recognition methodology for communications network safety).

References

1. Kłosowski, P.: Speech Processing Application Based on Phonetics and Phonology of the Polish Language. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2010. CCIS, vol. 79, pp. 236–244. Springer, Heidelberg (2010)
2. Kłosowski, P., Dustor, A.: Automatic Speech Segmentation for Automatic Speech Translation. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2013. CCIS, vol. 370, pp. 466–475. Springer, Heidelberg (2013)

3. Dustor, A., Kłosowski, P.: Biometric Voice Identification Based on Fuzzy Kernel Classifier. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2013. CCIS, vol. 370, pp. 456–465. Springer, Heidelberg (2013)
4. Kłosowski, P.: Improving Speech Processing Based on Phonetics and Phonology of Polish Language. Przeglad Elektrotechniczny R 89(8), 303–307. Sigma-Not (2013)
5. Rabiner, L.R., Schafer, R.W.: Introduction to Digital Speech Processing. Foundations and Trends in Signal Processing 1(1-2), 1–194 (2007)
6. Tsontzos, G., Orglmeister, R.: CMU Sphinx4 speech recognizer in a Service-oriented Computing style. In: IEEE International Conference on Service-Oriented Computing and Applications (SOCA), pp. 1–4 (2011)
7. Bilmes, J., Bartels, C.: Graphical model architectures for speech recognition. IEEE Signal Processing Magazine 22(5), 89–100 (2005)
8. Bilmes, J., Zweig, G.: The graphical models toolkit: An open source software system for speech and time-series processing. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), p. IV-3916–IV-3919 (2002)
9. Pellom, B.: SONIC: The University of Colorado Continuous Speech Recognizer. University of Colorado, Colorado (2001)
10. Pellom, B., Hacioglu, K.: Recent Improvements in the CU SONIC ASR System for Noisy Speech: The SPINE Task. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Hong Kong (April 2003)
11. Young, S., Evermann, G., Hain, T., Kershaw, D., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., Woodland, P.: The HTK Book. Cambridge University Engineering Department, Cambridge (2002)
12. Bonastre, J.F., Wils, F., Meignier, S.: ALIZE, a free toolkit for speaker recognition. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2005), vol. 1, pp. 737–740 (2005)
13. Stevens, S.S., Volkman, J.: The relation of pitch to frequency. American Journal of Psychology 53, 329 (1940)
14. Garofolo, J.S., Lamel, L.F., Fisher, W.M., Fiscus, J.G., Pallett, D.S., Dahlgren, N.L., Zue, V.: TIMIT Acoustic-Phonetic Continuous Speech Corpus. Linguistic Data Consortium, Philadelphia (1993)
15. Hermansky, H.: Perceptual linear predictive (plp) analysis of speech. Journal of the Acoustical Society of America 87(4) (1990)
16. Ziółko, B., Manandhar, S., Wilson, R.C., Ziółko, M., Gałka, J.: Application of HTK to the Polish language. In: International Conference on Audio, Language and Image Processing, ICALIP 2008, pp. 1759–1764 (2008)
17. Fauve, B.G.B., Matrouf, D., Scheffer, N., Bonastre, J.F., Mason, J.S.D.: State-of-the-Art Performance in Text-Independent Speaker Verification Through Open-Source Software. IEEE Transactions on Audio, Speech, and Language Processing 15(7), 1960–1968 (2007)