# JAVA TOOLS FOR TEACHING SPEECH RECOGNITION

*Stephan Euler*

Fachhochschule Gießen-Friedberg, University of Applied Science
Wilhelm-Leuschner-Straße 13, D-61169 Friedberg, Germany

## ABSTRACT

In this paper we present our concept for a sequence of experiments with speech recognizers used in teaching speech recognition techniques. The experiments are performed with a combination of own tools and the Hidden Markov Toolkit (HTK). The first experiment demonstrates speaker dependent recognition based on the dynamic time warp algorithm. In the course of this experiment all utterances from the students are recorded and used to build up a data base. Both the recognizer and the tool used for viewing and editing the speech data are written in Java making them platform independent and easy to extend. The recorded speech data is then utilized to train and test a speaker independent recognizer.

## 1. INTRODUCTION

The lecture is designed mainly for students in computer science and gives an overview of the entire field of speech recognition. Starting from the basic properties of speech signals the whole processing chain of feature extraction, recognition, syntactic and semantic analysis and dialog modelling are covered.

The presentation of the methods and technologies is accompanied by a number of practical exercises. We use HTK for all experiments with Hidden Markov models (HMMs). Additionally, we implemented two tools on our own:

- fbview: an audio file viewer
- fbdtw: a dynamic time warp (DTW) recognizer

Both tools are programmed in Java. We choose Java as programming language mainly because it is freely available and platform independent. We also found that the resulting software is fairly easy to maintain and extend. Combining HTK and our own tools we developed a sequence of experiments starting from feature extraction leading to recognition with phonetic units. Speech data is collected throughout the DTW experiment and reused for the HMM training. In this way no further data is required. Additionally, using own speech recordings makes the experiments more transparent and motivating for the students.

## 2. AUDIO FILE VIEWER

The central tool for our experiments in speech recognition is an audio file viewer called fbview. Originally developed to simplify browsing through a large number of audio files we added over the time more and more features. The basic capabilities are:

- fast access to a large number of files
- various file formats for wave form and label information
- basic wave form editing functions
- transcribing the speech signals

Other features include signal analysis and a first version of a viewer for network files. Figure 1 shows a screen shot of the tool.

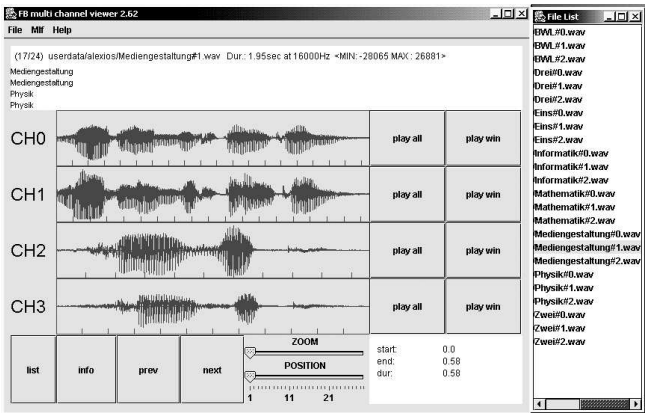


**Fig. 1**. The audio file viewer fbview.

### 2.1. File formats

The tool uses the Java library classes for reading files. It tries to open a specified file as an audio input stream. Then the format specification is obtained by calling the appropriate methods. Otherwise, i.e. if an exception of type UnsupportedAudioFile occurs, the file is either handled as a NIST SPHERE waveform or as a headerless file with raw

audio data. Although the various codings in e.g. wav-files are recognized internally, the present version of fbview can display only single channel, 16 bit PCM data. Additionally, two multi-channel formats are supported:

- SpeechDat-Car four-channel file format [2]
- multicolumn ASCII files: each column is used as one channel.

The label information is retrieved either from the header of the file or a separate label file. An efficient way of storing label information are master label files (MLFs). Fbview is mostly used together with MLFs. The labels are shown in a separate label area. Optional segment boundaries are marked in the wave form display.

In general, the programm recognizes the file formats more or less automatically and is able to find the corresponding labels. But some combinations can lead to an unexpected behavior. Then it is necessary to specify the proper settings manually with the command line options.

### 2.2. Signal analysis

For each wave a separate spectral viewer is available. Figure 2 shows a screen shot of a spectral view. Both the FFT and the LPC spectrum of the selected segment is shown with a small offset added to the LPC spectrum for the sake of clarity. It is possible to store one or more spectra as reference and compare them with the current spectrum.

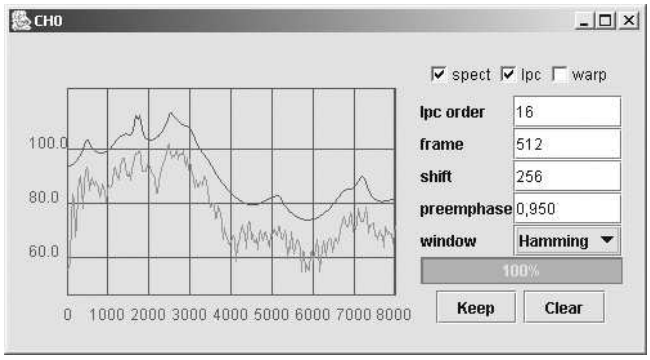


**Fig. 2**. The spectral analysis in fbview.

With regard to multichannel data we have also implemented a correlation tool. With this tool, the correlation between two channels is calculated. Again, a correlation function can be stored as reference for comparison.

### 2.3. Transcriptions

The transcriptions are shown in a window area at top of the wave views. This is a standard text area and all text editing techniques including cut and paste can be applied. In order to simplify the work, a lexicon can be specified. From this lexicon a window is constructed in which each entry is assigned to a button. Clicking on a button then inserts the label into the transcription line of the active wave. This approach is very helpful in the case of small vocabularies. It significantly reduces the time needed and helps avoiding typing errors.

### 2.4. Interface

Other applications can exchange data with fbview through a TCP-based interface. The commands SET and GET are currently implemented. Sending the sequence SET $s_1$ $s_2$ $s_3$ ... $s_n$ causes fbview to create a so-called internal wave representation from the $n$ integer values and appended it to the list of files. After receiving the command GET, fbview sends the values of the samples of the current top view. The ASCII representation of the values can result in fairly long transmission times. But it provides a universal representation, independent from e.g. the programming language.

## 3. FEATURE EXTRACTION

Most of the students do not have a background in signal processing. Therefore it is impossible to go into the details of the feature extraction process. We confine the presentation to a general description of the following steps:

- word boundary detection
- dividing the signal into overlapping frames
- calculating features for each frame
- using delta-coefficients to capture the dynamic behaviour of the speech signal

The students should get a basic understanding of the transformation of the continuous speech signal into a sequence of feature vectors. Furthermore, the concept of vector quantization is described. Although we do not apply vector quantization in the experiments, we need the concept for introducing discrete HMMs. As discrete HMMs are easier to explain we use them as introductory example.

The build-in spectral analyzer of fbview is used to demonstrate some basic properties of the frequency analysis of signals. The effects of e.g. increasing frames size or different window functions can be best seen with well defined reference waveforms such as sine or square waves. We have implemented a signal generator for such wave forms. The tools is connect through the TCP-Interface described in 2.4 with fbview. In this way, the students can reproduce some basic properties of frequency analysis.

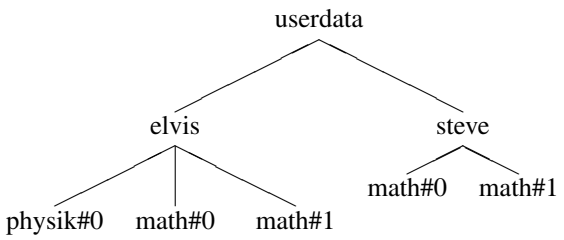


**Fig. 3**. Directory structure used in fbdtw, example with two users (elvis and steve)

## 4. DYNAMIC TIME WARP RECOGNIZER

The first speech recognition experiment deals with speaker dependent, isolated word recognition. The recognizer fbdtw uses a fairly standard implementation of the Dynamic Time Warp (DTW) algorithm. The user builds up the vocabulary by specifying single words or a complete list of words. From each word a given number of references (default is 3) are prompted. Each recorded utterance is displayed for inspection. Sometimes, for example, the automatic word boundary detection fails. The user can reject such utterances with obvious faults.

Each utterance is stored in an own file. The file name is build from the label and a counter as shown in the following example

| physik#5.wav | | | |
|---|---|---|---|
| physik | # | 5 | .wav |
| label | separator | count | extension |

In this case the word *Physik* was spoken. This is the 5-th utterance of this word.

Multiple users are supported. For each user a separate directory is created. All reference utterances from one user are then stored in the corresponding directory. All user directories are placed in a common directory (default name userdata). Figure 3 shows an example with two users.

We use the term *users* but the concept can be employed likewise to handle different sets of words from the same speaker. During recognition any number of users can be selected at the same time. For training a new word, however, exactly one and only one user has to active.

No further information besides the names of the directories and wav-files is needed in fbdtw. Therefore - in addition to the commands in fbdtw - standard techniques can be applied to maintain the data. As an example, copying an user is achieved by simply copying the user directory (and restarting fbdtw).

Figure 4 shows a screen shot of the DTW-tool. In this example three users (steve, test and tmp) are available. Currently only the first speaker is loaded. The vocabulary consists of 11 German city names. The last test utterance is shown in the window below the control window. The recognition yielded the word Berlin. For comparison all scores are shown in an additional window.


**Fig. 4**. The DTW-Tool fbdtw.

By default all test utterances are stored in a directory autosave with subdirectories for individual users. If only one user is activated, the utterances are stored in the corresponding directory. A special directory *unknown* is provided for the case of multiple active users. In both cases the utterances are named in#n.wav, n denoting the counter.

The first task of the students consists of:

- train references for a given list of words.
- test the performance of the recognizer
- if necessary, optimize the performance by editing the references
- test the performance with references from another student

The vocabulary consists of the 8 words shown in Table 1. BWL is the abbreviation of *Betriebswirtschaftslehre*, i.e. business administration. Most recognition errors result from wrong decisions of the word boundary detector. Such errors can be corrected in the references by deleting extra segments (with fbview or another audiotool) and removing truncated utterances. In the second case, a retraining should be performed to complete the set of references.

The recordings - both references and test utterances - of all students are collected. By specifying a common vocabulary we obtain a data base for the HMM experiments. Additionally it is useful material to demonstrate the variability of utterances for the same word from different speakers. The

**Table 1**. Vocabulary for DTW experiments

| | |
|---|---|
| Mathematik (mathematics) | BWL |
| Mediengestaltung (media design) | Eins (1) |
| Physik (physics) | Zwei (2) |
| Informatik (computer science) | Drei (3) |

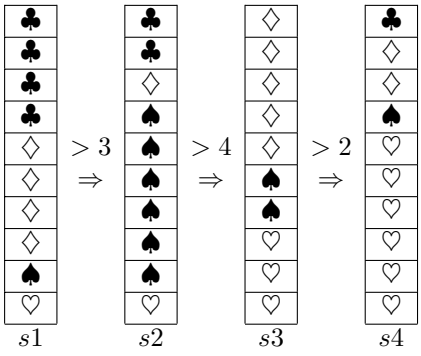


**Fig. 5**. Model with four states

students can compare their own recordings of a given word with these of their colleagues.

## 5. HIDDEN MARKOV MODELS

We start the discussion of HMMs with a simple example. It is a variation of the classic urn and balls model as described e.g. in [3]. We replace the urns with decks. Figure 5 shows an example. This model can be easily performed from the students. Each team of students obtains cards to form the decks and a dice. They start with drawing a card from the the first deck. Then they toss a dice. If the result is higher then the given number, they move on to the next deck.

Starting from the model we explain the application of HMMs for speech recognition. The student then train their own models using the HTK tools. The data base consists of the recorded utterances from the DTW experiment. In a first experiment, all training utterances are put together to form both training and test set. A set of Perl scripts are provided for the individual steps of the training process. The test utterances at this point also serve as test data. Therefore we usually obtain a very high recognition rate. The actual performance is examined in the direct audio mode of the HTK recognizer HVite.

A further test can be performed with the utterances that had been saved during test of the DTW recognizer. These utterances, however, are unlabeled. Therefore, the next task for the students comprises the transcription of at least a subset of the utterances. This task is performed with fbview easily. Using the resulting MLF the recognition performance for the unseen data is measured.

Finally, the generated models are tested in a connected word task. Therefore, the HTK grammar has to be extended. Combinations that could be the name of a lecture such as *Physik Eins* and *Mathematik Drei* should also be covered. Again, the performance is tested with direct audio input. We also have started to build up a small data base of such connected word utterances.

### 5.1. Subword models

Although the data base is far from sufficient for a reasonable training of subword units, the general procedure can be shown. The required phonetic transcriptions of the words are looked up in the freely available machine readable pronunciation dictionary from the university of Bonn [4]. Then the complete training procedure for the phonemes occurring in the mini-vocabulary is performed. Based on the resulting models an automatic phoneme level segmentation of the utterance is obtained. The resulting boundaries are examined with fbview. Due to the limited data base we find in general only a poor match. Finally we combine the 'phoneme' models to new words and test the recognition in the direct audio mode.

## 6. CONCLUSION

We have presented the experiments we use to demonstrate basic speech recognition techniques. Going through the whole process of data collection, training a recognizer and finally testing the performance helps the students to understand the relevant problems and techniques. Currently we check thoroughly all recorded utterances from the last three courses to integrate them into a common data base. The Java tools in their present version can be downloaded from the page www.fh-friedberg.de/users/euler/tools.htm.

## 7. REFERENCES

[1] S. J. Young et al, *The HTK Book*, Cambridge University Engineering Department, 2002.

[2] A. Moreno, B. Lindberg, C. Draxler, G. Richard, K. Choukri, S. Euler, and J. Allen, "Speechdat–Car. A large speech database for automotive environments," in *LREC*, Athen, 2000.

[3] L.R. Rabiner and B.-H. Juang, "An introduction to hidden markov models," *IEEE ASSP Magazine*, vol. 3, pp. 4–16, 1986.

[4] T. Portele, J. Krämer, and D. Stock, "Symbolverarbeitung im Sprachsynthesesystem Hadifix," in *Proc. 6. Konferenz Elektronische Sprachsignalverarbeitung*, Wolfenbüttel, 1995, pp. 97–104.