JavaSpeakerRecognition – Interactive Workbench for Visualizing Speaker Recognition Concepts on the WWW

Andrzej Drygajlo and Gary Garcia Molina

Signal Processing Laboratory
Swiss Federal Institute of Technology Lausanne, Switzerland
andrzej.drygajlo@epfl.ch

Abstract
The purpose of this paper is to introduce a user-friendly computer assisted learning (CAL) workbench in order to support traditional teaching in the Speaker Recognition area. The workbench (an interactive on-line laboratory) is based on Java and Java-enabled Web browser. The first prototype demonstrator developed at the Swiss Federal Institute of Technology Lausanne (EPFL) for speaker identification training consists of four modules: Dynamic Time Warping (DTW), hidden Markov Modelling (HMM), Vector Quantization (VQ) and Gaussian Mixture Modelling (GMM). These four modules aim at presenting, visualizing and investigating in a uniform way basic concepts of speaker recognition in a single user-friendly environment and allow for easy and highly illustrative learning through experiments with real speech data. They can be used for conventional classroom experiments, in the students’ laboratory or can provide self-study means for distance learning applications or for further free exploration.

1. Introduction
In the framework of the ISCA Education Special Interest Group (EduSig), the author of this paper and his colleagues of the group committee are preoccupied by the identification of best education practice in the Spoken Language Engineering (SLE) domain and its dissemination especially using the challenging and new possibilities of information technology such as the Internet [1].

Recent developments in computer networking technologies offer new ways to teach and learn in the Speech Science domain. Speech processing is inherently multimedia in nature, involving both sound and vision, and Speech Science students need to acquire practical skills in listening, analysis, manipulating and interpretation performance. Yet, there seems to be a lack of tools that could help students in getting an appropriate starting and experience in some specific speech processing domains, in particular in Speaker Recognition. Some of the existing software allows learning on the experimental base; however, it is often just a run-test-see-result approach. We believe that a good education workbench should demonstrate and explain the methods and their principles rather than only emitting numbers, scores, etc., which is an essential demand, particularly, in such a complex area as Speaker Recognition.

The key to successful speaker recognition training is to create tools that open up the field to interactive investigation by the student. We need learning that is individualized to the needs of each student. All students are different, with different backgrounds, knowledge, interests, and learning styles. Only highly interactive approach can discover individual problems and offer relevant learning experience. A major problem with learning today is the increasing tendency to confuse information with learning. The key to achieving interactive learning capabilities is to use the interactive capabilities of modern computers and communication data networks.

The introduction of the Java programming language provides users with the freedom to create interactive content for the Web by developing new data types and the methods to operate on them. The computer can now become an active participant and a friendly computational tool in the interactive learning process. We are currently using Java to develop prototype interactive learning tools for the speech processing laboratory [2,3]. The first interactive learning tool, called JavaSpeechLab, developed by the EPFL Speech Processing Group (SPG), allows the student to build and test new speech analysis algorithms in the laboratory and experiment with speech signals without having to learn a great deal about irrelevant mechanics of programming when these algorithms are already implemented. A reasonably complete set of short-term analysis procedures (time-domain analyses, spectral and time-spectral analyses (Fourier and multi-resolution)) permits a surprising variety of speech decomposition and analysis concepts to be demonstrated. In our software laboratory the common frame-window environment is provided for all types of analysis from the simplest one (e.g. short-term energy or zero-crossing rate) to the most complex based on multi-resolution wavelet packet transforms which use different windows and frames at different frequency sub-bands.

The newest interactive learning tool, recently developed by the EPFL SPG, is called JavaSpeakerRecognition. It is a visualization system consisting, at present, of four main modules: Dynamic Time Warping (DTW), hidden Markov Modelling (HMM), Vector Quantization (VQ) and Gaussian Mixture Modelling (GMM). The whole system is oriented on explaining major aspects of the text-dependent and text-independent speaker identification task in a uniform framework. Such a task seems to be an appropriate one for introduction into the speaker recognition area. This tool can be used for conventional classroom experiments, in the students’ laboratory or can provide self-study means for further exploration, which can help students, understand speaker recognition principles and facilitate their analysis.

2. Pedagogical aspects
Concrete information is more easily depicted, more imaginable by learners, better remembered and more consistently interpreted than abstract information.

Desirable features of CAL systems for spoken language engineering (SLE) training result from the above cited
learning principle and an integrated approach to teaching and learning in this domain. To introduce the SLE background in the framework of undergraduate courses at the Electrical Department of the Swiss Federal Institute of Technology Lausanne in Switzerland, a course "Speech Processing" (28 hours of lectures and 56 hours of laboratory for 8th term (final year) students, has been given since 1994. In the LTS Speech Processing Laboratory working with real signals and practical algorithms is necessary to grasp fully the difficulties of speech signal processing. Only a limited amount of time is available for laboratory work, moreover, each experimental set-up requires a non-negligible learning period. For that reason it was necessary to develop a highly user-friendly laboratory workbench that can be used with most speech processing experiments and to be completed by new algorithms developed by students themselves. On the other hand the students have access to several computer platforms and the Internet, and they can continue the investigations outside the laboratory. The laboratory is presently only available for approximately 10% of our students because the laboratory experiments are run on several dedicated computer platforms and not across Internet.

The purpose of our present project in the framework of the New Learning Technologies Programme is to revolutionise this approach and to transform all the resources of the students’ laboratory into a virtual workbench available to all students in an asynchronous interactive way.

For the development of this on-line laboratory, all elements – text information, diagrams, figures and laboratory exercises – are to be integrated on a single electronic medium (Web browser) in order to fully exploit the possibilities of contemporary teaching, learning and communication technology and to lead corresponding practices far into a new, partially still unknown area. Phases of conveying factual knowledge and of laboratory exercises are to be introduced in a self-guiding manner and explored in four main levels: content, task description, demonstration description and laboratory simulations. This approach is not without risks and, thus, has to be carefully accompanied with empirical pedagogical methods, as well as continuous evaluation.

The core design guidelines are as follows: interactive – the user can do more then simply press and watch; immediate – if possible, the system responds to user actions in real time; exploratory – an important and relevant section of parameter space underlying any effect or process is made available to the user for direct manipulation; uniform – a consistent style of interface design is followed; easy-to-use – if possible, enable direct manipulation of objects; documentation-rich – comprehensive and uniform HTML documentation is available for each defined task and demonstration, complete with suggestions for exploration and further reading.

3. Java and its promise

The key to a successful speech processing training is to create tools that open up the field to interactive investigation by the student. This section deals with Java tools chosen for developing the speech analysis laboratory for the Web and discusses some of their implementation aspects.

Java is both a program environment and a programming language [4]. Java has been tailored specifically for networked computing, such as that which takes place on the Internet. The Java source code is written in the same way that code for any other language is written. Java is an object-oriented programming language. Java source code is compiled using a Java pre-compiler, which creates platform-independent Java byte-code. This byte-code is unlike normal machine code in that it is written for the “Java Virtual Machine” (JVM) and is not, therefore, native to any particular processor or operating system. Any program that is written in Java can run on any computer as long as the JVM is inside. The Java programming language allows the developer to build both applets and stand-alone applications. Applets are pieces of compiled code that are referenced within Web pages (HTML documents) on the WWW. An applet requires a Java-enabled browser to run. Such a browser (which is platform-dependent software) reacts by downloading the referenced Java class. If the specified class requires code that is contained in one or more additional classes (which generally is the case), these supporting classes are then retrieved from the server. As each class is retrieved it is examined and any additionally required supporting classes are retrieved. This process continues until the browser has downloaded all the classes that have been referenced within any of the already downloaded codes. Once the classes are downloaded, the browser executes the code through a Java interpreter (which is the software that emulates the JVM). Java is often referred to as an "interpreted and compiled" language. Java applets give the Web the power of continuous, interactive, real-time, visual, and aural instruction. Web documents come alive because the computer can respond instantly to a user's input. Stand-alone Java application programs are written in almost the same way as applets, but do not require, nor do they use, a Web browser.

4. Basic concepts for speaker recognition teaching

Speaker recognition is the process of automatically recognizing who is speaking on the basis of information obtained from recorded speech signals [5]. Closed-set speaker identification task, which seems to be reasonably simple (easier to understand that speaker verification) and consequently appropriate one for introduction into the speaker recognition area, is the process of determining from which of the registered speakers a given utterance comes. In speaker identification, a speech utterance from an unknown speaker is analyzed and compared with models of known speakers. The unknown speaker is identified as the speaker whose model best matches the input utterance. In other words, we have to calculate a “distance” (similarity measure) between the input utterance and each of the speaker models. The smallest distance identifies the unknown speaker.

Learning is more correctly attributable to well orchestrated design strategies than to the inherent superiority of various media. Having in mind this principle, we developed in the EPFL SPG an interactive learning tool called JavaSpeakerRecognition, which transforms the WWW page in an easy-to-use, self-guided speaker identification laboratory. Since concepts enable us to simplify, categorize, and thus better cope with the diversity of recognition methods, we designed the recognition visualization procedure based on a clear and intuitive concept of two windows executed in the same manner for all speaker recognition techniques. They are: distance diagram window and accumulated distance diagram window (Fig. 1).
The four speaker recognition techniques chosen for the JavaSpeakerRecognition tool are Dynamic Time Warping (DTW), Hidden Markov Modelling (HMM), Vector Quantization (VQ) and Gaussian Mixture Modelling (GMM). Two of them are deterministic methods (DTW and VQ) and the other two are statistical methods (HMM and GMM).

These four techniques can also be divided into text-dependent (DTW and HMM) and text-independent methods (VQ and GMM). The former require the speaker to provide utterances of the key words or sentences having the same text for both training and recognition trials, whereas the latter do not rely on a specific text being spoken.

The current version of JavaSpeakerRecognition uses linear prediction derived cepstral coefficients (LPCCs) as elements of the acoustic vectors for all the four speaker recognition techniques.

The distance diagram gives a cartographic map of the association between the acoustic vectors representing a tested speaker utterance and the components of the reference speaker model. The coloured rectangles of the map (local distances \(d_{k,l}\) between the acoustic vectors and the corresponding elements of the reference models) visualize the space where the test-reference mapping is searched. The path of the minimal accumulated distance of pattern matching is highlighted by the rectangles \(d_{k,G}(k)\). The curve of the accumulated distance \(\check{d}(k)\) is drawn in the accumulated distance diagram and the total distance value \(\check{d}(T)\) is used for speaker identification decision.

In this uniform visualization framework of the distance diagram and the accumulated distance diagram, the distances are computed differently for the four different speaker recognition techniques. In the case of the DTW method, the accumulated distance is an average cepstral distance obtained for time-aligned acoustic vectors of the test and reference utterances, using a dynamic programming with local constraints. For the VQ method, the accumulated distance is given by the average cepstral distance between acoustic vectors of the test and the centroids of the vector quantization codebook of the reference acoustic vectors. In the case of statistical methods (HMM and GMM), the speaker identification system is a maximum-likelihood classifier. For HMM the accumulated distance corresponds to a log likelihood function of the most probable sequence of model states. In the GMM case, the accumulated distance is calculated as a log likelihood function over all the acoustic vectors of the test utterance knowing the GMM model of the reference speaker. For visualization purposes only the rectangles corresponding to the most probable mixture of the Gaussian probability density functions are highlighted.

The text-dependent methods are usually based on template matching techniques in which the time axes of an input speech sample (acoustic vectors) and each reference template (DTW) or reference model (HMM states) of registered speakers are aligned, and the similarity (distance) between them accumulated from the beginning to the end of the utterance is calculated. Fig. 2 gives a detailed clarification of the time-alignment problem in the case of hidden Markov modelling. For text-independent speaker modelling (VQ and GMM), it is important to model the acoustic classes, which
comprise a person’s speech but not the temporal ordering of the acoustic classes. An example of such an accumulated distance modelling for the VQ method is presented in Fig. 3.

5. The JavaSpeechRecognition application and applet

A JavaSpeakerRecognition applet has been implemented to demonstrate how Java could be used to develop an interactive, uniform learning environment for speaker identification on the WWW. The Java Standard Development Kit SDK 1.3 and JavaBuilder v.4 have been used in applet and application development.

After the programming experiment with the JavaSpeakerRecognition the authors can state that Java has some advantages and some disadvantages for the development of the interactive workbench for speaker recognition. The Java language is unique in its in-built handling of the network protocols required to send and receive data across networks such as the Internet. Java is making Web pages dynamic. The user provides his/her input with the familiar computing tools of keyboard and mouse and receives feedback through data, image and sound. Java allows a Web environment in which numerical analysis can be performed on computationally intensive engineering problems. The browser's Java Virtual Machine runs the Java applets locally, rather than requiring a constant and slow continual transfer of data from a remotely located Web server. This limits sound files registration and read-write functions for the local disk. Consequently the training of speaker models is not available in the applet version and the speaker model files and speaker sound files are pre-registered and are loaded with the applet. The application version of JavaSpeakerRecognition offers the full functionalities of the program including the acquisition of speech signals from a microphone and speaker model training.

6. Conclusions

The aim of the work presented in this paper was to introduce a versatile and user-friendly on-line laboratory accessible through Internet in order to successively replace classical laboratory exercises, and support the Speech Processing lectures.

The future of development effort on the Web utilizing Java appears to be very promising. A demonstration applet written in the Java language has been developed for the speaker identification application. It provides fast manipulation of speaker recognition algorithms, such as Dynamic Time Warping (DTW), Hidden Markov Modelling (HMM), Vector Quantization (VQ) and Gaussian Mixture Modelling (GMM), and gives immediate feedback when answers are being compared with concepts. It reduces computational effort and therefore allows students to understand principles in a uniform framework. The applet program can be executed on any computer connected to the Internet and running a Java-enabled Web browser by accessing http://scgwww.epfl.ch/JavaSpeakerRecognition.

7. References


