

# FIVE - Framework for an Integrated Voice Environment

Alexandre Maciel  
Informatics Center  
Federal University of Pernambuco  
Recife, Brazil  
amam@cin.ufpe.br

Edson Carvalho  
Informatics Center  
Federal University of Pernambuco  
Recife, Brazil  
ecdbcf@cin.ufpe.br

*Abstract* - This paper presents a framework for an integrated voice-based interfaces development. The FIVE consists of five modules: speech recognition, speech synthesis and speaker verification, representing the digital signal processing areas and other two modules: integration and application. This is an ongoing working and currently we are improvements stage of first three modules. We can achieve 90% of accuracy in recognition task and almost 80% of intelligibility in our synthesizer's words.

**Keywords:** *Voice User Interface, Framework.*

## I. INTRODUCTION

Human beings have always been looking for new ways to ease the human-machine interaction. With the evolution of hardware and software for digital signal processing (DSP), human speech became one of the most promising means of interaction.

Although there have been advances, the development of speech-based applications remains unattractive to most programmers and small enterprises. Basically this is due to two factors: DSP technology requires specific knowledge and still there aren't appropriated tools to manage every step of the speech interface development process in an integrated way.

In recent years, numerous initiatives have taken place to make speech-based interfaces more appealing. While the building frameworks have greatly facilitated the development of these applications, they still work isolated in specific speech processing tasks, hence their integration is not an easy deal.

The FIVE idea is to join the best available techniques for each speech processing task and offer the developers an integrated tool to seamlessly build speech recognition, speech synthesis and speaker verification interfaces for any desired application.

This paper is divided as follows: section II presents a speech technologies review; section III presents the FIVE framework with a specific description for each module; section IV presents the partial results; and finally section V presents considerations and future works.

## II. REVIEW OF SPEECH TECHNOLOGIES

### A. Speech Recognition

The Automatic Speech Recognition (ASR) task is essentially a pattern recognition task. According to Duda et al. [1] most pattern recognition systems can be partitioned into four components as shown in Figure 1.

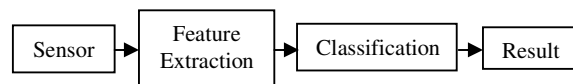


Figure 1: Pattern classification components

Specifically for speech recognition systems, the sensor is responsible for audio acquiring and conversion into digital samples, the feature extractor computes signal data in symbolic and numeric information, and the classifier uses this information to assign categories. Finally, the result block takes account of the considerations.

According to O'Shaughnessy [2] the main approaches for feature extraction are Linear Predictive Coding (LPC) and Mel-Frequency Cepstral Coefficient (MFCC). The LPC parameters were largely used in the 70's and became a dominant speech recognition representation, as an automatic and efficient method to represent speech because of the advantages that linear predictive provides in terms of generating a smooth spectrum, free of pitch harmonics, and its ability to model spectral peaks reasonably well. On the other hand, MFCC parameters take advantage of the perception properties of the human auditory system by sampling the spectrum at mel-scale intervals.

For the pattern classification task O'Shaughnessy [2] mentions four techniques: Dynamic Time Warping (DTW), Hidden Markov Models (HMM), Artificial Neural Networks (ANN) and Support Vector Machine (SVM). Each of these approaches has been extensively studied over the years and complete descriptions can be found at [3, 4, 5, and 6].

### B. Speech Synthesis

A Text-To-Speech (TTS) synthesizer is a computer-based system that should be able to read any text aloud, whether it was directly introduced in the computer by an operator or scanned and submitted to an Optical

Character Recognition (OCR) system. The TTS procedure consists of two main phases. The first one is Natural Language Processing (NLP), where the input text is transcribed into a phonetic or some other linguistic representation, and the second one is Digital Signal Processing (DSP), where the acoustic output is produced from this phonetic and prosodic information [7]. A simplified version of the procedure is presented in Figure 2.

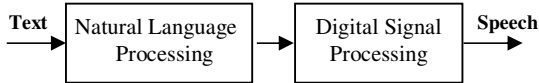


Figure 2: A simple diagram for TTS systems.

The main goal of the NLP module is to produce the phonetic representation from an input text along with the prosody information. To perform this task, the NLP module is divided into three steps: the text preprocessing step, where numerals, special characters, abbreviations, and acronyms are expanded into full words; the pronunciation analysis step, where the pronunciation of certain words, including homographs and proper names, are determined; and the prosodic analysis step, where the prosodic features of speech are determined.

The NLP module information is used as input for the DSP module. The DSP objective is to produce synthesized speech, which can be done by several different methods. The methods are usually classified into three groups: articulatory synthesis, which attempts to directly model the human speech production system; formant synthesis, which models the pole frequencies of speech signal or transfer function of vocal tract based on source-filter-model; and concatenative synthesis, which uses different lengths of prerecorded samples derived from natural speech [8].

### C. Speaker Recognition

Speaker Recognition refers to two different tasks: Automatic Speaker Identification (ASI) and Automatic Speaker Verification (ASV). In the identification task, an unknown speaker is compared against a database of known speakers, and the best matching speaker is given as result. The verification task consists of deciding whether a voice sample was produced by a claimed person. In this work we will focus in the ASV task [9].

Just as ASR, ASV is a pattern recognition task and can work with the same techniques for feature extraction and pattern classification with some particularities. For speaker recognition, features that expose high speaker discrimination power, high inter-speaker variability, and low intra-speaker variability are desired. For these particularities, the MFCC and LPC parameters can represent them very well.

A large number of methods have been proposed for speeding up the verification process. For text dependent verification systems the Vector Quantization (VQ) technique has presented good results since the speaker's parameters provide great distinction between them, making distance measures very efficient [10]. For text-independent recognition, Gaussian Mixture Model

(GMM) systems [11] have received much attention, since they are considered as the state-of-the-art method [12].

### III. FIVE

FIVE is an object-oriented framework built from open source frameworks written in Java. The frameworks chosen have presented good results in DSP state of the art. The Java language was chosen because it's extremely portable and easy to integrate with the existing frameworks.

Five is composed of five modules in order to assist developers in building voice interfaces for their applications. The base modules represent three areas of speech processing: Automatic Speech Recognition engine (ASR), Text-To-Speech engine (TTS) and Automatic Speaker Verification engine (ASV). The intermediate module represents the Application Programming Interface (API) responsible for integrating base modules and the application development abstraction. The last module is the Application module (APP) where developers can instantiate the base module through the suggested API.

FIVE provides a flexible way in which developers can create speech-based interface. It's possible to build interfaces with distinct base-modules alone or combined. Figure 3 shows a representation of the FIVE architecture.

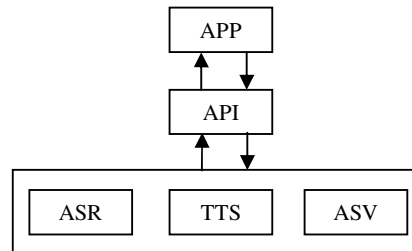


Figure 3: FIVE architecture.

#### A. Module 1:ASR

The ASR module is a speech recognition interface engine which comprises five steps. In the first step the developer builds a word dictionary to use in his application. In the second step the database recording is performed by an integrated audio acquiring application which is able to store files in 8 or 16 kHz, with 8 or 16 bit samples and mono or stereo format. The third step performs the feature extraction from the files obtained on the previous step. For that, the MFCC and LPC techniques are available; in addition, a Voice Activity Detection (VAD) algorithm optimizes the audio processing. The fourth step is responsible for conducting the training process. In this step, samples from the recorded database are used to create reference models, functions or parameters for the training data base. MLP, SVM, DTW and HMM techniques are available. Finally, the last step performs the pattern classification of the test database samples using the same techniques from the training step and provides results in the form of error rates and confusion matrix.

Java based framework were used for each technique implementation. The Neuroph framework was used for the MLP technique. Neuroph is lightweight Java neural network framework used to develop common neural network architectures. It contains well designed, open source Java libraries with small number of basic classes which correspond to basic neural networks concepts [13].

The Weka framework was used for the SVM technique. Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from within a Java code [14].

The Java ML framework was used for the DTW technique. Java ML is a library aimed at software engineers and programmers, composed of reference implementations for machine learning algorithms described in the scientific literature [15].

An adjustment of the HTK framework was used to implement the HMM technique. HTK is a portable toolkit for building and manipulating hidden Markov models which consists of a set of library modules and tools available in C source form. Script files containing calls for speech analysis, HMM training, testing and result analysis were created [16].

*B. Module 2: TTS*

The TTS module is a text-to-speech interface engine comprised by two steps. In the first step, a Natural Language Processing was created using the FurbSpeech framework. FurbSpeech is a Java-based API that converts Portuguese text into speech. It performs a generic input text processing and the correspondent phonemes are joined with prosodic information [17]. In the second step, the Digital Signal Processing, a Mbrola database was used. Mbrola is a project that gathers and offers sets of synthesized speech diphone-based databases in many languages, and provides them free for non-commercial applications [18]. FIVE, through the phoneme list obtained in the NLP step, joined with prosodic information and Mbrola samples, is then able to generate speech using 16 bits codification.

*C. Module 3: ASV*

Analog to the ASR module, a five steps structure was developed. First, the developer builds a speaker database with correspondent login and a password containing the transcription of his access code. The second and the third steps are identical to those from the ASR module. In the second step each speaker records his access code using an integrated audio acquiring application which is able to store files in 8 or 16 kHz, with 8 or 16 bit samples and mono or stereo format. The third step performs the feature extraction from the files obtained on the previous step. The fourth step is responsible for conducting the training process. In this step, samples from the recorded database are used to create reference models, functions or parameters for the training data base. VQ and GMM techniques are available. Finally, the last step performs the speaker verification based on the user identity. To make the best decision, a threshold is calculated for each speaker.

Basic errors measures as false acceptance, false rejection and total success rate are also obtained.

The Sautrela framework implemented the VQ technique. Sautrela is a highly modular and pluggable open source framework focused on speech recognition. It unifies in a single framework almost all the tasks related to pattern recognition such as signal processing, model training and decoding [19].

Until now it wasn't found a Java-based framework for the GMM technique. Then, it was decided to use the PyMix framework, leaving its translation to a near future. The Python Mixture Package (PyMix) is a freely available Python library that implements algorithms and data structures for a wide variety of data mining applications with basic and extended mixture models [20].

*D. Module 4: API*

The speech recognition, speaker verification and speech synthesis engines uses in an application usually are supported by a set of software that hides implementation details from the programmer. The aim of the API module is to offer developers a real time control of all base-modules and input/output audio interfaces.

The development of the API module has not started yet, but the idea is to take as standard the VoiceXML 3.0 definitions. VoiceXML is a standard XML format for specifying interactive voice dialogues between a human and a computer. For more information about VoiceXML see W3C site [21].

*E. Module 5: APP*

The application module includes an abstraction of the possible applications that can be developed with FIVE. In this module, instances of the API can be easily created and inserted in the context of any application.

IV. PARTIAL RESULTS

A database consisting of 20 words recorded by 20 speakers (15 male and 5 female) was generated to test the ASR module. Utterances followed the format 8 kHz, 16 bit, mono and were divided into 75% for training and 25% for testing. Then, MFCC and LPC features were extracted. Table 1 shows the results for each technique applied.

TABLE I. ASR RESULTS

Technique	MFCC	LPC
MLP	80,01%	79,48%
DTW	79,98%	66,81%
SVM	82,22%	85,07%
HMM	91,55%	90,47%

The main parameters used for each technique were: MLP (number of hidden layers: 12, number of interactions: 1000 and minimum error rate: 0.1), DTW (Euclidean distance), SVM (Kernel: linear cost: 1024) and HMM (3 states per phoneme and 10 Gaussian mixtures).

For the TTS module, a survey was conducted through a questionnaire with 20 users. Ten sentences were generated by the FIVE synthesizer using two voices from Mbrola in Brazilian Portuguese. Next, users were asked to transcribe the content of the phrases heard as they understood it. Table 2 shows the results for words and phrases sentences.

TABLE II. TTS RESULTS

	Word	Sentence
Voice 1	72,95%	58,00%
Voice 2	79,21%	62,00%

A database of 20 speakers (15 male, 5 female) recorded passwords and the passwords of other two unknown speakers were used to test the ASV module. Utterances followed the format 8 kHz, 16 bit, mono. Then, MFCC and LPC features were extracted. Table 3 shows results for each technique applied.

TABLE III. ASV RESULTS

Technique	MFCC results		
	FAR	FRR	TSR
VQ	7,22%	4,41%	88,00%
GMM	9,43%	8,32%	91,12%
	LPC results		
	FAR	FRR	TSR
VQ	6,85%	4,58%	86,24%
GMM	8,14%	8,48%	90,20%

The results show the False Acceptance Rate (FAR), False Rejection Rate (FRR) and Total Success Rate (TSR). The main parameters used for each technique were: VQ (number of centroids: 1 and number of iterations: 100) and GMM (64 Gaussian mixtures).

V. CONCLUSIONS

The results of the FIVE can be shown in two ways: quantitatively and qualitatively. Quantitatively, we can evaluate that rates of acceptance of the recognition modules are promising. HMM and GMM techniques obtained values up to 90%. Improvements still to be carried out expecting higher rates: new techniques for Voice Activity Detection can be used to obtain better cuts in utterances; other techniques for features extraction, based on wavelets, can be used to obtain the most representative characteristics of the signals; and classifiers evolutions or hybrid classifiers can be used to obtain better rates.

The TSS module results were considered unsatisfactory. Users had difficulties in understanding sentences and words and several errors were reported as crucial before implementing this module in a production environment. We are working on our own database, recorded in studio with phonetically balanced sentences

and we are studying the feasibility of using the HTS framework in order to improve the quality of generated speech [22].

Qualitatively a production environment test was not possible yet. This test is very important to evaluate the API usability and portability. We intend to apply this test for a group of developers in real applications and draw up a questionnaire arguing the experience of each one.

In general we evaluate the FIVE as an innovative tool that provides an easy way to implement and integrate voice interfaces. Having said this, we believe that voice interfaces may become more popular and open up a new range of applications for speech technologies.

REFERENCES

- [1] Duda, R., et al, 2000. *Pattern Classification*, 2<sup>nd</sup> Edition.
- [2] O'Shaughnessy, D., 2008. *In Invited paper: Automatic speech recognition: History, methods and challenges*, Pattern Recognition 41 (2008) 2965-2979.
- [3] Sakoe, H. and Chiba, S., 1978. *In Dynamic programming algorithm optimization for spoken word recognition*, ICASSP.
- [4] Rabiner, L. R., 1989. *In A Tutorial in HMM and Selected Applications in Speech Recognition*, Readings in Speech Recognition.
- [5] Cristianinni, N., 2000. *In An Introduction To Support Vector Machine*, 1<sup>st</sup> Edition.
- [6] Haykin, Simon, 2000. *Neural Networks – Principles and Practice*, 2<sup>nd</sup> Edition.
- [7] Dutoit, T., 1997. *In High-quality text-to-speech synthesis an overview*, ICASSP.
- [8] Lemmetty, S., 1999. *In Review of Speech Synthesis Technology*, Master's Thesis, Helsinki University of Technology.
- [9] Campbell, J., 1997. *In Speaker Recognition: A Tutorial*, Proceedings of IEEE, 1997.
- [10] Soong, F. K., et al, 1985. *In A Vector Quantization Approach to Speaker Recognition*, ICASSP.
- [11] Reynolds D., R. Rose. . *In IEEE Trans. Speech Audio Process.* Robust text-independent speaker identification using Gaussian mixture speaker models.
- [12] Reynolds, D., T. Quatieri, R. Dunn. 2000. *In Digital Signal Process., DSP.* Speaker verification using adapted Gaussian speaker mixture models.
- [13] Neuroph website. Retrieved January 20, 2010 from <<http://neuroph.sourceforge.net/>>
- [14] Weka website. Retrieved January 20, 2010 from <<http://www.cs.waikato.ac.nz/ml/weka/>>
- [15] Java ML website. Retrieved January 20, 2010 from <<http://java-ml.sourceforge.net/>>
- [16] HTK website. Retrieved January 20, 2010 from <<http://htk.eng.cam.ac.uk/>>
- [17] FurbSpeech website. Retrieved January 20, 2010 from <<http://code.google.com/p/furbspeech/>>
- [18] Mbrola website. Retrieved January 20, 2010 from <<http://tcts.fpms.ac.be/synthesis/mbrola.html>>
- [19] Sautrela website. Retrieved January 20, 2010 from <<http://www.sautrela.es/>>
- [20] PyMix website. Retrieved January 20, 2010 from <<http://algorithmics.molgen.mpg.de/Software/PyMix/>>
- [21] VoiceXML. Retrieved January 20, 2010 from <<http://www.voicexml.org/>>
- [22] HTS website. Retrieved January 20, 2010 from <<http://hts.sp.nitech.ac.jp/>>