A ROBUST SPEECH COMMAND RECOGNIZER FOR EMBEDDED APPLICATIONS

Alexandre Maciel[1], Arlindo Veiga[1], Cláudio Neves[1], José Lopes[1]
Carla Lopes[1,2], Fernando Perdigão[1,3] and Luís Sá[1,3]
[1]*Instituto de Telecomunicações, Universidade de Coimbra, Portugal*
[2]*Instituto Politécnico de Leiria - ESTG, Portugal*
[3]*Departamento de Engenharia Electrotécnica e de Computadores, FCTUC, Universidade de Coimbra, Portugal*

Keywords: Speech recognition, noise robustness, embedded systems.

Abstract: This paper describes a command-based robust speech recognition system for the Portuguese language. Due to an efficient noise reduction algorithm the system can be operated in adverse noise environments such as in cars or factories. The recognizer was trained and tested with a speech database with 250 commands spoken by 345 speakers in clean and noisy conditions. The system incorporates a user friendly application programming interface and was optimized for embedded platforms with limited computational resources. Performance tests for the recognizer are presented.

1 INTRODUCTION

Environment noise can drastically decrease the performance of automatic speech recognizers as it is the case of vehicle and factory environments. Under such adverse conditions, many speech recognition applications use a reduced vocabulary in order to improve the recognition rate. Even in the case of simple command recognizers, noise reduction techniques should be carefully designed. There are two common approaches to the noise robustness problem: signal processing techniques applied to the input speech or online adaptation of the recognition models to the present noise conditions. The latter generally requires higher computational resources and is not suited to embedded applications. Signal processing techniques for noise reduction include Wiener filtering, histogram equalization (Peinado and Segura, 2006), and others. A powerful feature extraction system for noise robust speech recognition was standardized by ETSI (ETSI, 2003). This system was developed for Distributed Speech Recognition and includes an Advanced Front-End to be implemented in client terminals such as portable devices and cell phones. These terminals send the extracted parameters to a remote server that runs a speech recognition engine. In embedded speech recognition with limited vocabulary, both the feature extraction and the speech recognition tasks are carried out in the terminal devices. Another common requirement for these embedded systems is their ability to run on limited hardware resources.

In this paper we describe a command speech recognizer for European Portuguese that incorporates a new and efficient algorithm to reduce the noise in the acoustical signal and runs in low complexity embedded hardware platforms. The recognizer was trained and tested with a speech database that was recorded under the framework of a cooperation project funded by the Portuguese government (Tecnovoz, 2008).

The paper is organized as follows. In section 2 we describe the algorithms that were used in the extraction of noise robust features. In section 3 we describe the Tecnovoz database and the training of the system. The implementation of the decoder is addressed in section 4. Section 5 describes the application programming interface to operate the recognizer. Performance tests are presented in section 6. Finally, in section 7, we present the conclusions of this paper.

2 NOISE ROBUST FEATURE EXTRACTION

The ETSI noise reduction front-end is based on a two stage Wiener filtering system. The Wiener filter is estimated in the linear frequency domain and is implemented by a time domain convolution. A modified implementation of the ETSI front-end was presented by (Li et al, 2004), where both the filter estimation and operation is carried out in the Mel frequency domain. In our implementation we improved the efficiency of this modified front-end by using the signal processing scheme depicted in Figure 1.

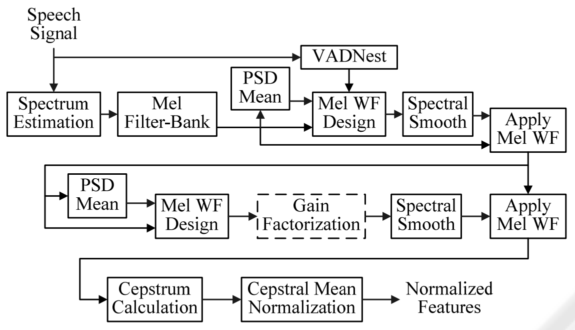

Figure 1: Block diagram of the feature extraction system.

As can be seen the speech signal is processed by a two stage Wiener filter (WF). The estimated signal spectrum is applied to a Mel filter-bank. The signal frames are classified as noise only or speech plus noise by the VADNest block, as required for the Wiener filter design.

A novel characteristic of this front-end compared to others resides on the “Spectral Smooth” block. In our implementation we reduce the operations involved in the smoothing of the Wiener coefficients to a single pre-computed matrix (Neves et al, 2008). The noise reduction system includes a second Wiener filter stage, which, apart from a gain factorization block, is similar to the first one. We found that the gain factorization proposed in (ETSI, 2003), when applied in the Mel frequency domain does not increase the system performance and can be omitted.

The de-noised frames are converted to cepstral coefficients by a discrete cosine transform (DCT) and their means are normalized (CMN), (Neves et al, 2008), by an online algorithm. As a result, 12 cepstral coefficients plus log energy are produced which, together with their first and second time derivatives, lead to a feature vector with 39 components.

3 MODEL TRAINING

3.1 Database Description

The recognizer is based on hidden Markov models (HMMs) which were created using the speech feature vectors described in the last section. The Tecnovoz speech database was used to train the HMM models. This speech database was designed regarding typical application demands in terms of vocabulary and acoustic environments. The collected speech includes about 250 commands and several phonetically rich sentences. About 30 minutes of spoken content was recorded from each speaker. Three acoustical environments were considered, namely Clean (TVFL), Vehicle (TVV) and Factory (TVF) environments, as indicated in Table 1.

All speech files in the database were automatically classified according to its signal-to-noise ratio (SNR). For training proposes only files with SNR above 15 dB were used in the experiments, totalling 137,237 files. From these, 103,001 (75%) were picked up for training 27,382 (20%) for testing and 6,854 (5%) for development.

Table 1: Audio files and gender distributions.

Distributions	TVFL	TVF	TVV	Total
Audio Files	119975	8633	8629	137237
Gender (F/M)	99/180	19/16	9/22	345

3.2 Training

Three different approaches were used to find the acoustic models which best fit to the task of command recognition: whole-word models, context-free phone models, and context-dependent triphone models. The word models are best suited for a command recognizer but have the disadvantage of being tied to a predefined vocabulary, which is not the case with phone models.

The models were trained using the HTK toolkit (HTK3, 2006) which performs Baum-Welch parameters re-estimation. In order to evaluate the performance of the models in a recognition task, Viterbi algorithm was used and a grammar was defined with all the 254 commands in parallel. Results showing the comparative performance of the three approaches are presented in section 6.

4 DECODER

The decoder task consists in finding the best sequence of commands given the speech features and models. The decoder is based on the Viterbi algorithm applied to a command grammar task and follows the “token passing” paradigm (HTK3, 2006). The core of the decoder was written in C++ programming language, but critical sections were optimized in assembly using Intel SSE and AMD 3DNow! floating-point extensions.

A confidence measure of the recognition results was incorporated in the decoder. This measure is essential to real applications because there are always recognition errors and therefore the recognition results need to be accepted or rejected. Confidence measures can also be used for spotting errors as well as to detect out-of-vocabulary words. To detect out-of-vocabulary (OOV) words, we used a so called “filler model” (Yu et al, 2006). This model was trained with all commands and phrases present in the training database. In order to calculate the confidence measure, a reference model was formed by taking all phone models in parallel. The purpose of this model is to give a score comparison with the main result. For well pronounced words, both the reference model and the result’s models should give almost identical scores. On the contrary, the scores would be very different in the case of misrecognized words.

In a first step the decoder tests whether or not a result is an OOV word using the “filler model”. If the result is not assigned as an OOV word, then a confidence measure is computed using the reference model. This is done by passing the speech feature vectors associated to the result to the reference model. The difference between the scores of the reference and the result’s models is normalized by the number of frames. This value is then applied to a sigmoid function in order to obtain a normalized confidence measure between 0 and 100%.

In integrating the decoder with the feature extraction front-end, a VAD (voice activity detector) is used in order to signal speech endpoints. This detector is essential since it frees the decoder from processing very long sequences that do not correspond to speech.

Due to the optimizations referred to previously, the decoder operates in real time. After the detection of a speech starting point, the decoder immediately starts the recognition process. As soon as the detector signals the speech endpoint, the decoder supplies the API, described in the next section, the best result for this speech segment.

5 APPLICATIONS INTERFACE

In order to use the speech recognizer in practical applications, we developed an application programming interface (API) which allows the real-time control of the speech recognition engine and the audio interface. As shown in Figure 2, the applications interact with the recognition engine through the API. The API hides from the user the low level details of the engine operation, audio drivers and grammar handling.

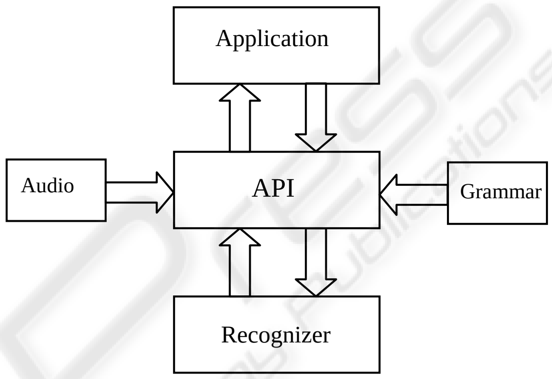

Figure 2: API interaction model.

The API was designed for the embedded Windows operating systems, such as XPe and CE. The grammars are specified in XML, according to W3C SRGS format.

From a programming point of view, the API consists of a single class referred to as SREngine. This class exposes to the applications a set of methods and properties that are described in Table 2.

Table 2: Main API methods.

Method/Event	Basic Description
InitSREngine	Method to initialize events, audio stream and the engine.
LoadGrammar	Method to load a grammar.
SetRuleState	Method to activate/deactivate a grammar rule.
StartRecognition	Method to start the recognition process.
StopRecognition	Method to stop the recognition engine.
OnSRecognition	Event that occurs whenever a recognition result is available with an associated confidence measure.

With the limited set of methods presented in Table 2 it is easy to build compact Windows speech recognition applications.

6 RESULTS

In section 2 we described a new noise robust feature extraction front-end based on the ETSI standard. In order to assess the merit of our front-end, we compared both front-ends using the Tecnovoz database. Whole word models were trained and tested as described in section 3, using both front-ends. The models are described by a 10 component Gaussian mixture. As it can be seen in Table 3, our front-end outperforms the ETSI front-end by a figure of 2% in recognition rate.

Table 3: Front-end comparison results.

Front-end	Recognition rate
ETSI front-end	94.88%
New front-end	96.88%

In section 3 we described 3 possible approaches for the recognition units: whole-word models, context-free phone models, and context-dependent triphone models. The training and testing datasets were the same as in the previous experiment and the results are presented in Table 4.

Table 4: Results obtained for Tecnovoz database.

Acoustic Model	Recognition rate
Whole-word	96.88% (10 mixtures)
Phone model	91.41% (16 mixtures)
Triphone model	97.13% (10 mixtures)
Triphone model	97.50% (16 mixtures)

With the whole-word model set the recognition rate is 96.88%. This model set has 46.7k Gaussians (about 3.6M parameters).

For the context-free phone model set we used a multiple pronunciation dictionary but, despite of this, we obtained a recognition rate of only 91.41% even with a larger number of 16 mixtures. As the main concern is word recognition performance, the silence/pause models are simply ignored in the recognition evaluation. This low performance value is obviously due to the lack of parameters: only 1,888 Gaussians (150k parameters).

For the triphone model set, no multiple pronunciation dictionary was used and, as before, the silence/pause models were ignored. The result of 97.50%, was obtained for 16 mixtures with 32,208 Gaussians (about 2.5M parameters) and 846 physical triphone models. The result for 10 mixtures is also shown and compares favourably with the whole-word case.

7 CONCLUSIONS

A speech command recognizer for the Portuguese language was presented in this paper. It incorporates a new noise robust front-end and a Viterbi decoder optimized for real time operation in embedded applications. A compact programming interface for application development was also presented. Several recognition units were discussed and evaluated.

Results with the presented recognizer show that the models based on triphone units are higher than whole-word or context-free phone models. A possible explanation for this result could be the fact that, on average, there are much more occurrences of triphones than whole-words, which leads to a better parameter estimation. The use of triphones is also a better solution, as it combines fewer parameters with higher recognition rate.

The noise robustness of our front-end was also evaluated, showing an increased performance when compared to the ETSI front-end. This result suggests that the ETSI front-end may be biased towards the database used in its development.

REFERENCES

ETSI, 2003. ETSI ES 202 050 v1.1.3. *Speech Processing, Transmission and Quality Aspects (STQ); Distributed Speech Recognition; Advanced Front-end Feature Extraction Algorithm; Compression Algorithms.* Technical Report ETSI ES 202 050, ETSI.

HTK3, 2006. *The HTK book (for HTK version 3.4).* Technical report, Cambridge University. England. http://htk.eng.cam.ac.uk/.

Li, J.-Y., Liu, B., Wang, R.-H., and Dai L.-R., 2004. *A Complexity Reduction of ETSI Advanced Front-end for DSR.* In proc. of ICASSP'2004, vol. I, pp. 61-64. Montreal, Canada.

Neves, C., Veiga, A., Sá, L., and Perdigão, F., 2008. *Efficient Noise-Robust Speech Recognition Front-end Based on the ETSI Standard*. Submitted to INTERSPEECH'2008. Brisbane, Australia.

Peinado, A., and Segura, J., 2006. *Speech Recognition over Digital Channels: Robustness and Standards*, John Wiley & Sons, Ltd. England.

Tecnovoz, 2008. http://www.tecnovoz.pt/web/home.asp.

Yu, D., Ju, Y., Wang, Y.-Y., and Alex, W., 2006. *N-Gram Based Filler Model for Robust Grammar Authoring.* In proc. of ICASSP'2006, vol. I, pp. 565-568. Toulouse, France.