Maximum Likelihood Endpoint Detection with Time-Domain Features

Marco Orlandi, Alfiero Santarelli, Daniele Falavigna

ITC-Istituto per la Ricerca Scientifica e Tecnologica
via Sommarive 18, 38050 Trento - Italy
{orlandi,santarel,falavi}@itc.it

Abstract

In this paper we propose an effective, robust and computationally low-cost HMM-based start-endpoint detector for speech recognisers[1]. Our first attempts follow the classical scheme feature extractor-Viterbi classifier (used for voice activity detection), followed by a post-processing stage, but the ultimate goal we pursue is a pure HMM-based architecture capable of performing the endpointing task. The features used for voice activity detection are energy and zero crossing rate, together with AMDF (Average Magnitude Difference Function), which proves to be a valid alternative to energy; further, we study the impact on performance of grammar structures and training conditions. In the end, we set the basis for the investigation of pure HMM-based architectures.

1. Introduction

The endpointing module is a critical part in any spoken dialogue system: weaknesses in the decision process impact directly on the whole chain, since missed speech fragments cannot generally be recovered. Speech endpointing is generally performed using a Voice Activity Detector followed by some postprocessing stage, the latter being introduced to correct front clipping, speech fragmentation and other typical phenomena.

Two main approaches are adopted in developing endpoint detectors for ASR:

- Threshold based: the decision is performed according to one (possibly adaptive) or more thresholds [4].
- Classifier based: a classifier (like a Viterbi decoder or an Artificial Neural Network) substitutes the threshold. This metod relies on general statistics rather than on local information [1].

The first class is the most widespread, as its algorithms are generally simpler and faster to implement. Its major drawback consists in the need of careful tuning of many parameters, something that makes such algorithms sensitive to environmental variations. The second one overcomes this problem at the price of an intensive and careful training of the classifier.

Another big issue of such modules are real-time constraints: the latency introduced by them should be minimal, in order not to introduce delays that can annoy the user.

Our preliminary investigation explores the classifier-based architecture, pursuing an architecture capable of real-time processing, easy to implement and to train, that should also reduce the post-processing stage to a minimum or to completely avoid it. In the first part of this paper we describe the Viterbi decoding scheme, the features adopted for speech/non speech discrimination and the various types of “language models” which can be used to improve certain weak points.

To study the impact of single modifications in the whole architecture, some speech databases formed by international phone calls and field data from real call center services have been used. These databases, together with the evaluation method adopted, will be described in subsection 3.2. In the end, experimental results will be discussed.

2. Viterbi-based endpoint detection

Maximum likelihood classification can be applied, by means of a Viterbi decoder, to feature vectors extracted from the speech signals; if properly chosen, such features can allow a fine discrimination between speech and non-speech.

A previous, deeper work on the subject [1] showed that other types of classifier can guarantee better performance; our choice to adopt Viterbi classification is due to the opportunity of reusing the same reliable and efficient code developed for our speech recogniser. What is more, Viterbi trellis structure is quite amenable to frame-by-frame processing, and this will be useful if we develop a real-time implementation based on such architecture.

It should be noted that the performance measured in this work represents an upper bound for actual values, since Viterbi decoding is performed off-line: as such, the algorithm estimates the maximum-likelihood choice from the whole sequence processed.

In view of real-time implementation, and its computational time issues, we decided to investigate exclusively features extracted in the time domain. Even if modern hardware can make frequency processing more and more appealing, algorithms in the time domain can still be the fastest competitors if suitably designed. Taking inspiration from [2] we focused our attention on energy, zero-crossing rate and average magnitude difference function. All of them share the property of having a bimodal distribution: their values split into two classes, roughly corresponding to speech and non-speech segments.

2.1. Energy

The short-time energy was computed according to the following formula:

$$E_n = \sum_{m=0}^{N-1} [x^2(n+m)] \qquad (1)$$

Even though it is not very robust against noisy backgrounds and impulsive interferences, energy is still a fundamental component in many widely used endpoint detectors.

[1] This work has been partially funded by the European project HOMEY: IST-2001-32434.

2.2. Zero-crossing rate (ZCR)

The short-time average zero-crossing rate is expressed by the following equation:

$$zcr_n = \frac{1}{2N}\sum_{m=0}^{N-1} |sign[x(n+m)] - sign[x(n+m-1)]| \tag{2}$$

While it is not a good index of speech/non speech discrimination alone, it is generally used with some success as a correction term in energy-based voice activity detectors [2].

2.3. Average Magnitude Difference Function (AMDF)

The expression that describes the short-time average magnitude difference function is

$$AMDF(k) = \sum_{m=0}^{N-1} |x(n+m) - x(n+m-k)| \tag{3}$$

Though this feature is usually exploited in pitch estimation, it can be observed that it also has a bimodal distribution, with the two classes corresponding to speech presence and absence. In fact, it can be shown [2] that every coefficient $AMDF(k)$ is related to $\hat{R}_n(0)$ and $\hat{R}_n(k)$, being $\hat{R}_n$ the short-time autocorrelation function: both $\hat{R}_n(0)$ (which is equivalent to energy) and autocorrelation are good discrimination features.

Further, in order to avoid fluctuations of a single AMDF coefficient, typically due to voiced segments (AMDF should tend to zero for $k = nP$, where P is the pitch period), we averaged five equally spaced coefficients, obtaining a ”box approximation” of the AMDF area.

2.4. Normalization

To achieve equalization of long-term channel effects, both energy and AMDF have been normalized by their maximum value over a 10 seconds window before taking their logarithm. This implies that their values lie in the negative range.

3. Performance evaluation

3.1. Evaluation method

Deciding whether an endpointing module works better than another can be a somewhat tricky task. Many performance measures have been used in literature, dealing with alignment and markers placement or simply giving frame classification figures. For the sake of simplicity we adopted a measure used in VAD evaluation, based on the following four parameters expressing misclassification [3]:

- FEC (Front End Clipping): clipping introduced in passing from noise to speeh activity;
- MSC (Mid Speech Clipping): clipping due to speech misclassified as noise;
- OVER: noise interpreted as speech due to the EPD remaining active after the end of speech period;
- NDS: Noise Detected as Speech (also known as “false alarm”).

Of the four, the first two are computed as a ratio between the number of samples misclassified and the total number of speech samples, while the second two are computed with respect to the number of noise samples. This achieves a sort of normalization according to the speech/non-speech ratio, yielding more significant figures: for example, if speech amounts only to a small ratio of the total, and all speech is uncorrectly misclassified as noise, the FEC parameter will sum to 100% anyway.

For our purpose, MSC and NDS are indeed the most important parameters to consider: the other two can easily be corrected by a simple hangover scheme, for example taking the previous and the following two frames around a speech segment.

As our target is the improval of a speech recogniser, recognition performance could be another good evaluation parameter; however, the data used in these experiments did not allow us to define a proper recognition task.

3.2. Speech data

Our endpointing module is mainly intended to work with telephone dialogue systems: it should overcome some typical problems of threshold-based endpointers we currently use [4]. As a consequence, our experiments were performed on telephone data, sampled at 8kHz (it should be noted, though, that the features considered can be made independent of operating frequency). Further work will investigate the impact of such techniques on other kinds of speech data (e.g. in the car).

In our first experiments we worked on the CallHome international telephone calls corpus [5], which seemed particularly suitable for our purposes, as it consists of long sequences of speech and background noise, rather than utterances with short background sequences in the beginning and end, which is typical of many speech databases: it is worth noting that we are interested in the ability of our algorithms to adapt themselves to varying conditions and events such as laughter, breaths, impulsive noises and so on.

Later on we considered field data acquired by commercial services based on our ASR system, and decided to include them (we could not drop CallHome recordings since new data were not sufficient for model training). These new data consist mainly of short utterances; however, they are closer to a real situation since they present a number of phenomena such as echo cancelling, line distortion, street noise, as well as those non-standard behaviours typical of voice interface usage (hesitations, background speech and so on). One group of data (named “Real1”) was taken by a menu access service, so it only contains isolated digits; the other (“Real2”) comes from a credit card entering service, that is to say, it consists of connected digits sequences.

To complete our data set we examined early recordings from the COralRom project, aimed at the acquisition of a European corpus of interactions with a speech interface [6]. COralRom corpus seems to be suitable to our purposes, and possibly it will become the sole data set for our future experiments; for the time being we just included a few acquisitions.

The whole amount of data was manually segmented, and divided into two sets as follows:

Training data

- CallHome: 5 male and 5 female voice recordings of 2 minutes each, for a total of 20 minutes;
- Real1: 63 recordings for a total of 16’23”; they include male and female voices and pure background noise;
- Real2: 11 recordings for a total of 3’30”; they include male voices and pure background noise;

- COralRom: a single sequence made of multiple sessions from both male and female speakers, with a total duration of 16’43”.

Test data

- CallHome: 3 female speakers, 4 minutes per recording, total 12 minutes;
- Real1: 28 recordings, male, female and background noise, total 7’1”;
- Real2: 3 recordings (all male), 57”;
- COralRom: a single multisession sequence with only female speakers, 6’30”.

Even if these data appear to be heterogeneous and of little significance from a statistic point of view, they allow us to show the benefit of multicondition training and the robustness of such a system even with a small amount of training data.

3.3. Segmentation of data

We had to adopt a certain number of rules in order to have consistent and significant speech-non speech segmentation of our data.

To start, we decided to set a minimum of 300 ms for non-speech segments (shorter segments can be breathing pauses within speech). It is likely that in the future we will increase this threshold to a couple of seconds, which is a reasonable endpointer behaviour; in the present work our main concern was accuracy in classification.

Secondly, a general classification of events was decided:

- *speech* includes speech, breaths directly connected to speech activity, hesitations and all “vowel-like” sounds (sounds like “oh!”, “er...”);
- *non-speech* includes all sorts of noise, isolated breaths, hisses, laughs, coughs, and also background speech and echo cancellation residuals.

4. Experimental results

Speech and noise have been represented as a single-state continuous Hidden Markov Model, with distributions modelled by a mixture of Gaussian functions having diagonal covariance matrices. Training has been performed via the standard Baum-Welch procedure. To test the system, a loop grammar with equiprobable transitions has been used; the whole logical structure can be represented as a finite state automaton, as depicted in Fig. 1

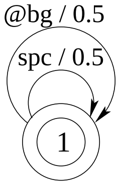

Figure 1: *Endpoint model based on loop grammar.*

First an “ideal” analysis window size was investigated. Setting a the number of Gaussian mixtures to N=8, and using energy as the only feature, we tried to vary the window size from 20 to 320 ms. The results are reported in Table 1.

	20 ms	40 ms	80 ms	160 ms	320 ms
FEC (%)	1.68	2.02	2.48	3.65	6.12
MSC (%)	11.85	8.38	6.16	3.46	1.26
OVER (%)	4.88	5.18	5.24	5.68	6.92
NDS (%)	22.69	22.01	19.98	18.84	17.85

Table 1: *Impact of variation of window size.*

From the table we observe that, by increasing the size of the analysis window, FEC and OVER deteriorate while MSC and NDS improve. On second thoughts, this means that a wider window makes the algorithm less accurate (the decision has to be made over a longer interval) but more robust (small “insertions” due to feature fluctuations are avoided). According to Table 1, a good choice for the window is 160 ms, so the parameter will be set to this value from now on.

The next cycle of experiments explores the impact of the number of Gaussian mixtures, still using energy as the sole feature (results are presented in Table 2).

	N=2	N=4	N=8	N=16	N=32
FEC (%)	3.74	3.67	3.65	3.65	3.65
MSC (%)	3.42	3.48	3.46	3.45	3.35
OVER (%)	5.62	5.64	5.68	5.74	5.74
NDS (%)	18.73	18.69	18.84	18.89	18.89

Table 2: *Impact of variation of Gaussian mixture dimension.*

Increasing the number of mixtures does not seem to be beneficial: in fact, an histogram of parameter distributions shows that the latter are quite compact and almost Gaussian in shape. Similar experiments were repeated with different feature vectors, but the results were generally similar.

As a consequence of such indications, the value N=8 was chosen as optimal for the following experiments.

Another issue of investigation was the composition of the feature vector. AMDF behaves in a similar way to energy, so it is worth understanding whether it can give advantages over it or together with it. We tried the combinations shown in Table 3.

By observing the results, AMDF appears to be almost equivalent to energy. It receives similar benefits from being coupled with zero-crossing rate, whereas the combination of the two features gives worse performance than each of them. Bad performance of ZCR alone is also confirmed.

Energy and ZCR, as well as AMDF and ZCR, can both be considered good feature vectors for endpoint detection. Note that overall performance is good, but indeed the algorithm lacks of robustness, especially for what concerns “insertions” (false alarms and mid-speech clipping).

Therefore, we tried to exploit all the possibilities of the Hidden Markov Model architecture, inserting other “language models” with different state connections and different transition weights.

The first structure considered (see Fig. 2) is a two-state network with a smaller weight on speech (labelled as “spc”) to non-speech (“@bg”) transitions (and viceversa), which should return longer sequences with less insertions.

The second network is more complex (Fig. 3), and tries to introduce some duration models. There is a high probability to enter the three-state path, which will return a unique decision for a sequence equal or longer than three observations (480 ms

	FEC (%)	MSC (%)	OVER (%)	NDS (%)
energy	3.65	3.46	5.68	18.84
ZCR	5.45	6.79	15.77	29.38
AMDF	4.11	3.62	5.96	19.26
energy+ZCR	4.10	2.80	5.62	16.78
AMDF+ZCR	4.55	3.10	5.80	17.16
energy+AMDF	3.53	3.80	5.33	21.37
energy+AMDF+ZCR	3.69	3.76	6.43	18.76

Table 3: *Feature vectors for endpoint detection.*

	FEC (%)	MSC (%)	OVER (%)	NDS (%)
energy + ZCR				
baseline	4.10	2.80	5.62	16.78
two-state	4.71	1.51	6.74	15.60
three-state chains	6.98	1.14	8.79	14.44
AMDF + ZCR				
baseline	4.55	3.10	5.80	17.16
two-state	5.50	1.46	6.85	15.60
three-state chains	8.15	1.13	8.45	14.53

Table 4: *Impact of grammars on performance.*

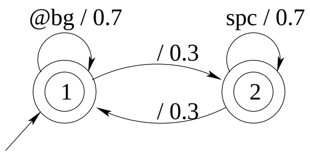

Figure 2: *Endpoint model based on two-state grammar.*

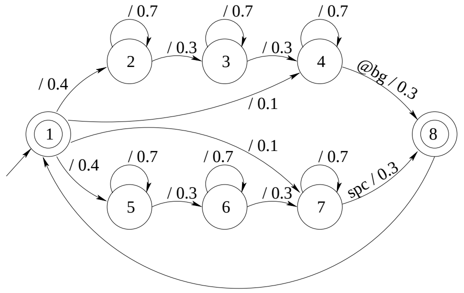

Figure 3: *Endpoint model based on three-state chains grammar.*

in the experiments presented). There is a skip transition which models shorter speech events.

Table 4 describes the effect on performance of such structures; as above, window size was set to 160 ms and the number of Gaussian mixtures to 8. MSC and NDS are reduced, while the other two parameters increase: it is similar to what was obtained by increasing the window size. Both effects cannot be pushed too far, but a good combination of the two should result to be effective.

5. Conclusions and future work

The above described experiments aimed at exploring the impact of certain parameters on general performance. The data we have at our disposition at the moment do not allow us to draw definitive conclusions on the best feature vector to use.

A good strategy for this architecture appears to be the combination of the Hidden Markov Model with a language model. In our future investigation we are going to embed duration models on state transitions, in order to reduce fragmentation in speech sequences which are passed to the recogniser.

We also intend to develop strategies for real-time decision on the Viterbi trellis. The impact on performance is still to be seen.

Finally, a better definition of training and test data sets will help us to evaluate performance of different architectures, allowing us also to compare this system with other endpointing modules.

6. Acknowledgements

The authors would like to thank Fabio Brugnara and Marco Matassoni for some valuable suggestions.

7. References

[1] J. Stadermann, V. Stahl, G. Rose, “Voice activity detection in noisy environments”, Proc. of Eurospeech 2001

[2] Lawrence R. Rabiner and Ronald W. Schafer, Digital Processing of Speech Signals, Prentice-Hall, 1978.

[3] F. Beritelli, S. Casale, G. Ruggeri, “Performance evaluation and comparison of ITU-T/ETSI voice activity detection”, Proc. of ICASSP 2001.

[4] G. Carli, R. Gretter, “A start-end point detection algorithm for a real-time acoustic front-end based on DSP32C VME board”, ICSPAT’92.

[5] Site of Linguistic Data Consortium http://www.ldc.upenn.edu/

[6] E. Cresti, M. Moneglia et al., “The C-ORAL-ROM Project. New methods for spoken language archives in a multilingual romance corpus”, Proceedings of LREC 2002.