A User-Friendly Interface for Text-Independent Phoneme Segmentation

Guido Aversano\textsuperscript{1,3}, Anna Esposito\textsuperscript{2,3}
\textsuperscript{1} Dipartimento di Fisica “E.R. Caianiello”, Università di Salerno, Italy
e-mail: guido.aversano@sa.infn.it
\textsuperscript{2} Department of Computer Science and Engineering, Wright State University,
Dayton, Ohio, USA, e-mail: anna@cs.wright.edu
\textsuperscript{3} International Institute for Advanced Scientific Studies (IIASS),
Vietri sul Mare (SA), Italy

I. Introduction
The speech segmentation interface discussed in this paper is a first step towards the implementation of an original speech recognition system based on the classification of previously segmented phonetic units. The main advantage of this approach is that it might work also in a multi-lingual context since phonetic transcriptions can be obtained without the need of language’s models and dictionaries. Language-dependent constraints can be added to the system, as a further processing phase, depending on the user’s demand.

Unlike most of the segmentation methods reported in the literature, an algorithm aimed to the above objective cannot rely on an externally supplied phoneme transcription for detecting phoneme boundaries. To overcome this limitation our research has been focused on implementing a linguistically unconstrained, or text-independent, segmentation algorithm (see Pellom and Hansen \cite{1} for details on the differences between linguistically constrained and unconstrained segmentation methods).

Moreover, the proposed algorithm will be embedded in a computer-aided system for segmentation and labeling of large speech corpora, to serve as a research tool for speech scientists involved in training ASR models or building synthesis applications.

The remainder of the paper is organized as follows. An exhaustive description of the segmentation algorithm, as well as a methodology for fixing algorithm’s parameters, has been provided in recently published papers \cite{2}, \cite{3} and will be shortly summarized in the next two sections.

Section 4 is devoted to analyze algorithm’s performance on noisy data. Section 5 will approach the issue of segmenting new datasets using a previously trained system. Section 6 will describe the main features of a user-friendly experimental interface, designed for speech segmentation and analysis. Conclusions and perspectives are given in section 7.

II. Preprocessing and Segmentation
The segmentation task, as performed by the proposed system, is carried out in two fundamental steps: speech preprocessing and phoneme boundary detection. The first step provides the encoding of the speech signal into a time-sequence of acoustic features. Any “short-time” representation of the signal can be used in order to obtain a sequence of vectors, every vector encoding a small segment of the speech wave (frame).

The phoneme boundary detection is performed by a heuristic algorithm \cite{2}, \cite{3} regulated by three parameters, namely \(a\), \(b\) and \(c\). The \(a\) and \(c\) parameters are integers, representing the number of speech frames involved in different phases of the algorithm’s implementation. The \(b\) parameter is a real number in the \([0, 1)\) interval that can be pictorially described as a particular threshold level used to reject “candidates” to the role of phonemic boundary.

Up to now, several speech encoding schemes, among which Mel-frequency Cepstral Coefficients (MFCC) \cite{4}, Linear Predictive Coefficients (LPC) \cite{5}, Perceptual Linear Prediction (PLP) \cite{6}, RASTA-PLP \cite{7}, Perceptual Critical Band Features (PCBF) \cite{2} and Mel-frequency Bank of filters (MelBank) have been tested with the proposed segmentation algorithm.

To select the encoding scheme that best suits the heuristics defined for the segmentation process, it is necessary a fine tuning of the algorithm’s parameters. Optimization of the parameters has to be made for each encoding scheme and, to this aim, the current system’s architecture includes an automatic optimization procedure \cite{3}. The procedure significantly reduces the involvement of the experimenter in this time-consuming task.

III. Experimental Results on Clean Speech
The search for a suitable encoding scheme was carried out on a collection of 480 sentences, extracted from the American-English DARPA-TIMIT database. These sentences are pronounced by 48 speakers (24 females and 24 males). Each speech wave, sampled at 16 kHz, has an associated labeling file, which contains the “true segmentation”, i.e. the actual positions (in sample index) of the phoneme boundaries manually detected by an expert phonetician.

The sentence segmentation obtained by the algorithm is compared with the true segmentation: a phoneme boundary identified by the algorithm is assumed to be “correct” if it is placed within a range of ±20 ms (or ±320 samples, in our case) from a true segmentation point.

An index for expressing the algorithm’s performance is the percentage of correctly detected phoneme boundaries,
defined as:

\[ P_c = 100 \cdot \frac{S_c}{S_t}, \]  

(1)

where \( S_t \) is the total number of true segmentation points (\( S_t \)) contained in the database (17,930), and \( S_c \) the number of phoneme boundaries correctly detected by our algorithm.

It is easy to prove that such index alone does not give a reliable measure of the quality of the performed segmentation. In fact, the algorithm could output a huge number of detected boundaries, fictitiously incrementing the probability of detecting true segmentation points, with the drawback of having inserted a large number of extra erroneous segmentation points. This phenomenon is known as over-segmentation and is quantified by an over-segmentation measure \( D \), defined as the difference between the total number of segmentation points detected by the algorithm \( S_d \) and the total number of true segmentation points \( S_t \):

\[ D = S_d - S_t. \]  

(2)

There is an alternative measure of over-segmentation, \( D' \), that can be useful to assess segmentation results from different databases. This measure is expressed as the percent ratio of \( D \) to the number of points \( S_t \) contained in the database [8]:

\[ D' = 100 \cdot \frac{(S_d/S_t - 1)}{S_t}. \]  

(3)

The percentage of correctly detected boundaries and the over-segmentation are strictly interdependent. Higher values of \( P_c \) can be obtained at expense of a higher over-segmentation. Therefore, in quantifying the absolute performance of the algorithm on different encoding schemes and for different parameter values, it’s advisable to fix one of the indices at a common value. The most natural choice [3] is to assume an over-segmentation \( D = 0 \). All the experimental results reported in this paper, unless otherwise stated, refer to \( D = 0 \).

Among the tested encoding schemes, MelBank analysis gave the best segmentation performances on the above selected database. For this particular encoding, the speech wave is represented by the output of a bank of filters spanning the whole frequency-range or a part of it. The filters’ shape and distribution along the frequency axis are not uniform, trying to reproduce the spectral resolution of the human ear. A further transformation of the filters’ output generates the Mel-frequency Cepstral Coefficients (MFCC representation [4]).

The best percentage of correct detection obtained by the proposed algorithm, using an 8-MelBank encoding and fixing the values of the free parameters at \( a = 2 \), \( b = 0.22365 \), \( c = 7 \) was \( P_c = 76.53\% \). The numeral preceding the name of the encoding method indicates the number of filters used for the analysis.

IV. Experimental Results on Noisy Speech

The speech database was then corrupted with additive traffic noise, randomly selected from a noise database. The algorithm performance was tested for signal-to-noise ratios ranging from 30dB to 0dB, using the 8-MelBank encoding. Results showed a surprising behavior, as evidenced in Figure 1.

As expected, for very noisy data the performance tends to decrease, but surprisingly an improvement, with respect to the clean data, was observed for signal-to-noise ratio (SNR) greater than 10dB SNR. The best percentage of correct detection, \( P_c = 78.06\% \), was found at 20dB SNR, when \( a = 3 \), \( b = 0.25413 \), \( c = 6 \).

Although additional tests should be performed on other noisy sources to assess the algorithm performance on noisy data, the above results indicate a satisfactory behavior of the proposed system. A comprehensive explanation for the improvements observed for slightly noisy data was not found yet. It can be supposed that the small amount of noise may enhance transition cues in the encoded speech.

V. Algorithm Generalization Properties

Once the parameters of the algorithm have been fixed to get the maximum performance on a particular set of speech data (the training set), its generalization ability was tested on a different dataset (the testing set). To this purpose a new set of TIMIT sentences was selected, containing 320 sentences pronounced by 32 speakers (16 females and 16 males) which were not included in the previous dataset. The total number of phoneme boundaries contained in the testing set was \( S_t = 11,947 \).

The optimal parameters’ values found for the clean training set \( (a = 2, b = 0.22365, c = 7) \) were also used for segmenting the testing set in both noisy and noiseless conditions. Speech waves were in both cases encoded using the 8-MelBank analysis. Detection results are reported in Table 1. The first row of Table 1 reports the segmentation performance on the clean training set (that was also employed to determine the optimal values for the free parameters). The subsequent rows show the performance on the testing set for clean and increasingly noisy conditions.

On the clean testing set, segmentation accuracy was almost the same as that obtained on the training set and the over-segmentation was very low (close to zero). As ex-
Table 1.
Segmentation performances when the algorithm is trained on a clean dataset and tested on the testing set for clean and increasingly noisy conditions.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$P_c$</th>
<th>$D'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>clean training set</td>
<td>76.53%</td>
<td>0%</td>
</tr>
<tr>
<td>clean testing set</td>
<td>77.19%</td>
<td>0.85%</td>
</tr>
<tr>
<td>noisy testing set (20dB SNR)</td>
<td>80.08%</td>
<td>7.26%</td>
</tr>
<tr>
<td>noisy testing set (15dB SNR)</td>
<td>80.97%</td>
<td>11.44%</td>
</tr>
</tbody>
</table>

Table 2.
Segmentation performances when the algorithm is trained on a noisy dataset and tested on the testing set for clean and increasingly noisy conditions.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$P_c$</th>
<th>$D'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>noisy testing set (20dB SNR)</td>
<td>78.06%</td>
<td>0%</td>
</tr>
<tr>
<td>clean testing set</td>
<td>74.45%</td>
<td>-4.96%</td>
</tr>
<tr>
<td>noisy testing set (20dB SNR)</td>
<td>78.19%</td>
<td>1.00%</td>
</tr>
<tr>
<td>clean testing set (15dB SNR)</td>
<td>78.54%</td>
<td>3.44%</td>
</tr>
</tbody>
</table>

expected, a slightly higher over-segmentation occurs when the testing set is corrupted by noise. For several applications a low over-segmentation value can be tolerated, since it could be eliminated by post-processing the output of the system, either manually as in the case of semi-automatic speech segmentation [9], or in an automatic way, as when using a text-to-speech system to generate, from the transcription, a synthetic signal to be aligned with the real one [10], [11].

One methodological concern that could be raised at this point is that the improvement obtained on the noisy testing sets could be attributed to the higher over-segmentation. However, it can be shown that it is mostly due to the effect of noise, as discussed in the previous section. To prove this observation, the equivalent percentage of correct detection, equivalent-$P_c$, was computed for the clean training set by fixing the over-segmentation value equal to that obtained for the noisy testing sets, i.e. either $D' = 7.26\%$ (when the SNR was 20dB) or $D' = 11.44\%$ (when the SNR was 15dB). It was found that this equivalent-$P_c$, calculated on the training set was 78.85\% for $D' = 7.26$ and 79.95\% for $D' = 11.44$. In both cases the performance obtained on the noisy data was higher than that obtained on the clean data.

The idea of training the system on noisy data was also explored. The results of such additional experiments are shown in Table 2.

It can be seen that, if trained with noisy data, the system performance on the clean testing set is lowered. Moreover, the over-segmentation value is negative, indicating that the number of phoneme boundaries found by the algorithm is less than the total number of boundaries in the dataset. Furthermore, the correct detection rates obtained on the noisy testing sets are lower than those obtained by training the system with clean data. However, this lowering in performance is not significant, since it’s accompanied by a lower over-segmentation value, suggesting that the efficacy of our algorithm is not impaired by a small amount of noise in the training data.

VI. The Graphical Interface
There were several reasons that brought to the idea of embedding the proposed algorithm and the optimization procedure in a graphical interface [2], [3]. The most important was to allow speech researchers to exploit an analysis tool where several encoding schemes as well as several new acoustic features of the speech wave could be easily identified from an acoustical analysis of the processed signal. Among the other reasons, there was the desire to reduce the time for manual segmentation, to study the signal and its several representations, and to provide a tool for tutorial activities on this subject.

In order to make the interface available to a wide number of speech researchers from several fields, the implementation was made on a Personal Computer and the code was written in the JAVA programming language [12] making it easily portable to Windows, UNIX, LINUX, Mac-OS, and other platforms. On the other hand, the underlying algorithms are coded in C++, allowing maximum speed in running the segmentation process.

The interface allows to load and display a speech waveform at several levels of detail. Afterwards the speech signal can be represented through different encoding schemes and the obtained sequence of acoustic features can be visualized through a pseudo-spectrogram representation (time on the $x$-axis, frequency bands or any other acoustic feature on the $y$-axis, and different tonalities of color for rendering the features’ weights). If a phonetic transcription of the wave already exists, it can be displayed below the waveform. A typical window of the interface is displayed in Figure 2.

![Figure 2. A typical window of the interface, displaying the waveform (top), the manual segmentation read from the TIMIT database (center), and the segmentation performed by the proposed algorithm (bottom).](image-url)
architecture, making easy the development of plug-ins for extending the interface capabilities without the need of re-compiling the entire code. Through plug-ins, the user can define new encoding schemes and segmentation algorithms, can load his own phonetic transcriptions, or, simply, add procedures for reading new or not included file formats.

VII. Conclusions and Perspectives
In the light of the experimental results presented above and the reported advancement towards the development of a user-friendly interface, the possibility of using phoneme segmentation as a basis for language-independent speech recognition systems becomes more concrete. At this point our research will move towards the definition of a communication protocol between the segmentation and classification phases. This cooperation is expected to improve the respective performances of the two phases.

Furthermore, the availability of a tool for visualizing the several phases (or outputs) composing the segmentation process can significantly speed-up the analysis and the solution of open issues such as how to further improve the segmentation performance. New encoding methods can be introduced, and evaluated through the comparison with those already tested, and both their performance and the details of their functioning can be assessed.

By exploiting graphical comparisons of the segmentation obtained through different encoding schemes, it will be possible to modify the preprocessing, or even the segmentation algorithm, in order to improve the system performance, especially for specific classes of phonemes whose detection is not so accurate.

In addition, the unclear role of noise in the performance improvement can be better understood with the help of graphics. Further works will be devoted to identifying a strategy for minimizing the size of the training set, and to study an alternative method for optimizing the free parameters on datasets not accurately labeled as the TIMIT database.

References

Acknowledgements: The authors would like to thank Prof. Maria Marinaro and Antonietta Esposito for useful suggestions and collaboration. This work has been supported by the NSF KDI program, Grant No. BCS-9980054 “Cross-modal analysis of speech signal and sense: multimedia corpora and tools for gesture, speech, and gaze research” and by NSF Grant No. 9906340 “Speech driven facial animation”.

Abstract: This work represents a further advance towards the implementation of an automatic system for text-independent phoneme segmentation. A user-friendly interface is required to make available to a wide public an already developed phoneme segmentation algorithm. The adoption of a flexible design both for the interface and the system’s architecture is presented and justified through the paper. In addition, in the light of new experimental results, it is advanced the possibility of exploiting the developed phoneme segmentation algorithm as a basis for language-independent speech recognition systems.

Sadržaj: Ovaj rad prikazuje progres u realizaciji sistema za segmentaciju glasa po fonemima, nezavisno od teksta. Jednostavan interface, za ranije razrađen algoritam segmentacije, potreban je kako bi isti algoritam bio šire dostupan. Zato je u radu prikazano i obrazloženo usvajanje fleksibilnog dizajna za interface i arhitekturu sistema. Osim toga, u svetlu nedavnih oglednih rezultata, predočena je mogućnost upotrebe segmentacije po fonemima kao osnov za prepoznavanje glasa nezavisno od jezika.

JEDNOSTAVAN INTERFACE ZA SEGMENTACIJU GLASA PO FONEMIMA NEZAVISNO OD TEKSTA, Guido Aversano, Anna Esposito