Using Prosodic Information for Disambiguation Purposes

Roberto Grettér, Dino Seppi

ITC-IRST, Trento, Italy
{grettér|seppi}@itc.it

Abstract
In this work, we describe how prosodic information can be employed to improve the performance of an Automatic Speech Recognizer (ASR) for specific restricted tasks. The approach exploits additional prosodic information in a post-processing stage. Prosodic features are estimated at word level; this additional information is encoded through a feature extractor and is then modeled using a statistical classifier. To train and test this system we collected an Italian database designed to focus on obtaining large improvements in ASR performance and on specific dialogue problems like ambiguous utterances. The proposed system yields a 69.5% relative word error rate reduction compared to a traditional state-of-the-art recognizer for the task of recognizing sequences of numbers.

1. Introduction
Although it is generally accepted that prosody carries, among other things, linguistic information, it is difficult to include it in an Automatic Speech Recognizer (ASR) to improve its performance. In this paper we address this problem by considering a number of restricted tasks — useful for spoken dialogue systems — where prosody plays an important role. In the future, we plan to extend the proposed approach to more complex and broader situations.

Automatic speech recognition is a task normally performed on a frame-by-frame level: single phonemes are recognized by analyzing short speech frames, typically 20 ms. In most state-of-the-art speech recognizers, energy, cepstral features, and their derivatives are used. This means that among the known prosodic features, only the energy of the signal is generally used as an acoustic feature. Prosodic features such as phoneme lengthening, pause duration, and pitch are discarded or only implicitly considered. This has to do with the fact that prosodic phenomena last many syllables or words, and are thus difficult to include in the acoustic models. Moreover, the information contained in the suprasegmental structure of speech also pertains to paralinguistic events, such as emphasis, emotion, speaker gender, etc., and seems irrelevant or even misleading for the recognition task. Nevertheless, during the last two decades there have been various efforts to apply this source of information for speech understanding purposes and for the improvement of ASR performance. While for the former task this approach has led to several successful systems (see for instance [1]), for the latter task it has so far produced promising but small improvements [2, 3, 4]. The study presented in this paper differs from the previous ones because the emphasis is on obtaining large improvements in ASR performance and on focusing on limited tasks rather than on general purpose ones. With respect to the method, we adopted an approach that is similar to [4] in that prosody is used only after the ASR stage, and to [5] in that the recognition framework is task oriented.

This paper is organized as follows. In Sections 2 and 3 we describe the speech database used and the analyzed tasks. Section 4 presents the architecture of the system. Section 5 reports on the details of the prosodic features and of the classifiers used, together with the classification results. Finally, we compare ASR baseline solutions against our architecture, showing the effectiveness of the proposed approach. Conclusions and future work are presented in Section 6.

2. Speech database
The data collected consist of approximately 5 hours of Italian read speech, acquired at 16 kHz, in a normal office environment. 50 non-professional speakers (25 males and 25 females) read about 70 utterances each. Altogether they constitute a reasonably varied sample of the different Italian accents.

The speakers were asked to read ambiguous utterances with different meanings, which were presented on a computer screen in random order. The meaning of each utterance was clarified either by pictures, text, or both. If necessary the speaker could ask a supervisor for clarification before recording the utterance. The data can be divided into four different parts according to the task performed (Table 1). For this study we focus on the first two tasks.

The NEG task consists in reading pairs of sentences that are very similar from a phonetic point of view, but carry completely opposite meanings:

No voglio andare a Verona
No I want to go to Verona

Even though punctuation is omitted, the two sentences are not ambiguous to a human listener, since they are phonetically different. Nevertheless prosody is often used by people to disambiguate this kind of sentences (in these data after the word no there is always a prosodic boundary), and it could thus be used to help an automatic speech recognizer to discriminate between the two words no and non ("no" and "non"), which are phonetically very similar.

The NUM task refers to sequences of numbers: the vocabulary is therefore limited to about 50 tokens which can be used to

<table>
<thead>
<tr>
<th>Description</th>
<th>ID</th>
<th>Number of sentences</th>
<th>Duration in minutes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Negations</td>
<td>NEG</td>
<td>1175</td>
<td>≈ 79</td>
</tr>
<tr>
<td>Sequences of numbers</td>
<td>NUM</td>
<td>539</td>
<td>≈ 84</td>
</tr>
<tr>
<td>Ambiguous sentences</td>
<td>AMB</td>
<td>1200</td>
<td>≈ 80</td>
</tr>
<tr>
<td>Groups of digits</td>
<td>DIG</td>
<td>626</td>
<td>≈ 55</td>
</tr>
</tbody>
</table>

Table 1: The collected prosodic database. In this work only the tasks NEG and NUM have been considered.
utter every number between zero and 999. The difficulty of automatic recognition of numbers is often underestimated because an isolated number is not ambiguous, and only a few syntactic rules govern the way tokens form numbers. On the contrary, a sequence of numbers (e.g. a telephone number) can be ambiguous: this is the reason why ASR systems usually require the user to utter sequences of digits instead. An example from our database is

\[
2100304472
\]

which is uttered as the following sequence of tokens:

\[
\text{due cento trenta quattro quattro cento settanta due two hundred thirty four four hundred seventy two}
\]

A standard ASR usually recognizes this sequence of tokens, utilizing a language model, optionally followed by some post-processing, to form the actual sequence of numbers. This is not straightforward since there are a lot of different valid combinations (60 for the above sequence).

Word and phoneme time markers were automatically generated [6] and manually checked by a professional annotator. Furthermore, the corpora NUM and NEG were prosodically labeled. For numbers sequences (NUM) we selected two types of labels, one for the tokens ending a number (closing), and one for those located in all the other positions in the number (center).

The above example becomes:

\[
\text{due\textsubscript{closing} cento\textsubscript{closing} trenta\textsubscript{closing} quattro\textsubscript{closing} quattro\textsubscript{closing} cento\textsubscript{center} settanta\textsubscript{center} due\textsubscript{center}}
\]

Similarly for task NEG we assigned the end label to each word preceding a prosodic boundary and the neutral label to the other words. In our data all no received an end label, while all non were marked as neutral.

### 3. Tasks description

As previously seen, the tasks considered are: **single word disambiguation and recognition of sequences of numbers.** For the problem of single word disambiguation we focus on two common Italian words used in dialogues: no and non at the beginning of a negation. Although these terms do not usually present any ambiguity problem to a human listener, they are a major source of misunderstanding for dialogue systems, mainly because they often occur during crucial dialogue stages. In such cases recognition errors can completely alter the meaning of the whole turn. In Italian a negative sentence, beginning with the word non looks very similar to an affirmative one if it is preceded by the negation no (see example in Section 2).

For the recognition of sequences of numbers the following issues hold. In Italian, like in English, each number is uttered as a combination of few basic tokens: fortythree is composed by the tokens forty and three. This is unambiguous only if there is the a-priori knowledge that fortythree is a stand-alone word, otherwise it could correspond either to the number “43” or to the sequence “40 3”. Although syntactic constraints do not allow some combination possibilities (four three is not ambiguous), in general the ambiguity grows as the number of tokens increases.

### 4. System architecture

The developed system (henceforth referred to as “prosodic recognizer”) works as shown in Figure 1. The speech signal is fed to two modules working in parallel: a “standard” ASR and a prosody extractor that has first to compute the pitch contour.

When the ASR module has recognized the sentence, it outputs both the tokens (a token is either a word or a sub-word, depending on the task) and the phonemes recognized, with their respective time markers. Then, the prosody extractor computes, for each recognized token, a prosodic feature vector, which is subsequently fed to a classifier that labels each token with a prosodic marker. Finally, a task-dependent set of rules is applied to the classifier’s output stream, to produce the final transcription by possibly modifying the original token sequence of the ASR module. Note that prosody is used after the ASR phase and that the final decision on the token category is taken by a classifier. It follows that this system is highly task dependent and a different classifier has to be trained for each different task.

#### 4.1. Prosodic features

A feature vector was computed for each token to represent prosodic information. The first three features are the number of phonemes in a token, the mean duration of the phonemes in each token normalized over the mean duration of the phonemes of the utterance, and the length of the pause following each token.

We also used features related to the pitch. Trying to keep as much information as possible from the pitch signal, we approximated its shape inside a token by two polynomial curves: a straight line and a cubic. The two curves are then represented by their coefficients. Before the polynomial approximation, the pitch was centered subtracting the (parametric) cubic fit computed on the whole utterance, in order to capture only local movements. Moreover, the pitch approximations of each token were stretched to a fixed length (50 frames).

The fundamental frequency was extracted using two different Pitch Detector Algorithms (PDAs). The first one (PDA 1) is based on the weighted autocorrelation function [7] and has proven to be robust to noise. The second algorithm (PDA 2) is based on the normalized cross correlation function [8] and was used in conjunction with a Viterbi algorithm to detect the best pitch path from all those found. This post-processing is fundamental to improve the performance of the pitch extractor. Finally, the pitch values were converted according to a logarithmic scale.

#### 4.2. The classifier and the rules

Our prosodic classifiers are Classification and Regression Trees (CARTs) [9] trained using k-cross-validation. This allowed to exploit the whole data dividing them in k parts (k = 10), testing on one tenth, and using the remaining parts for training. The process was iterated k times.

For the NEG task the classifier decision is used by the last module to possibly change no with non, if the no had been labeled as neutral, and vice versa. Instead, for the NUM task, recognized tokens are concatenated to form numbers on the basis of the prosodic label assigned to each of them, and of syntactic
rules for numbers: each token labeled as center is merged with the next one until a token with a label closing is found.

5. Experiments and results

The baseline ASR system used for this work is described in [10]. In this section we compare its performance with that obtained using the prosodic post-processing described so far. First we give the results of the prosodic classifier alone, using various feature subsets. Then we examine word error rate of the “prosodic recognizer” (Figure 1) for the two tasks under analysis.

5.1. Feature classification

Classification results are expressed in terms of mean (RR) and class wise (CL) recognition rate; CL is defined as the mean skewness of the two classes. Results using pitch features depend on the PDA used.

Table 2: Classification results [%] for the NUM task. Single features are in italics while subsets are not. Their related performance is given as recognition rate, RR, and class wise recognition rate, CL. The two values differ because of slight differences in the skewness of the two classes. Results using pitch features depend on the PDA used.

<table>
<thead>
<tr>
<th>Feature</th>
<th>#</th>
<th>RR</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>phoneme number</td>
<td>1</td>
<td>63.4</td>
<td>57.6</td>
</tr>
<tr>
<td>phoneme duration</td>
<td>1</td>
<td>92.3</td>
<td>91.8</td>
</tr>
<tr>
<td>phoneme information</td>
<td>1</td>
<td>94.6</td>
<td>94.2</td>
</tr>
<tr>
<td>pause length</td>
<td>1</td>
<td>89.6</td>
<td>90.4</td>
</tr>
<tr>
<td>pause length + phoneme info.</td>
<td>3</td>
<td>97.1</td>
<td>97.0</td>
</tr>
</tbody>
</table>

PDA 1

<table>
<thead>
<tr>
<th>Feature</th>
<th>#</th>
<th>RR</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear approximation</td>
<td>2</td>
<td>72.0</td>
<td>69.8</td>
</tr>
<tr>
<td>cubic approximation</td>
<td>4</td>
<td>69.9</td>
<td>67.0</td>
</tr>
<tr>
<td>pitch (linear+cubic)</td>
<td>6</td>
<td>75.6</td>
<td>73.4</td>
</tr>
<tr>
<td>pitch + phoneme info.</td>
<td>8</td>
<td>93.9</td>
<td>93.5</td>
</tr>
<tr>
<td>pitch + pause length</td>
<td>7</td>
<td>90.2</td>
<td>90.3</td>
</tr>
<tr>
<td>all</td>
<td>9</td>
<td>97.2</td>
<td>97.1</td>
</tr>
</tbody>
</table>

PDA 2

<table>
<thead>
<tr>
<th>Feature</th>
<th>#</th>
<th>RR</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear approximation</td>
<td>2</td>
<td>74.5</td>
<td>72.6</td>
</tr>
<tr>
<td>cubic approximation</td>
<td>4</td>
<td>68.9</td>
<td>66.5</td>
</tr>
<tr>
<td>pitch (linear+cubic)</td>
<td>6</td>
<td>76.9</td>
<td>75.2</td>
</tr>
<tr>
<td>pitch + phoneme info.</td>
<td>8</td>
<td>94.1</td>
<td>93.7</td>
</tr>
<tr>
<td>pitch + pause info.</td>
<td>7</td>
<td>91.5</td>
<td>91.5</td>
</tr>
<tr>
<td>all</td>
<td>9</td>
<td>97.2</td>
<td>97.0</td>
</tr>
</tbody>
</table>

Table 3 gives the classification results of the NEG task: the two words no and non were classified in terms of the prosodic boundary assigned to them. Results are coherent with those obtained in the previous task. In fact, we tried two different types of training for CARTs: using all the words available in the negation sentences (column “all data”) or only the two words we are interested in (column “no & non”).

From these results we can already observe that, although every group of features is capable of noteworthy classification performance, the features not related to the pitch give the best results. Further, it appears that there is little or no gain at all if the pitch-related features are used in combination with phoneme duration and pause length. This holds for both tasks and for both PDAs.

There are some possible reasons for the usefulness of the features related to the pitch contour. First, the PDAs used are not error free and even the more sophisticated one (PDA 2) has drawbacks, above all in presence of noise or when it is used with “freaky” voices. Secondly, since the pitch is a signal it cannot be lossless coded into a single or few features, like the coefficients of a regression line. The choice of how to code the pitch is not straightforward and always leads to the degradation of the original information. Finally, non-professional speakers are not always capable of properly expressing the tone variation while speaking. As the classification results obtained from the two different PDAs do not significantly differ when all the features are used, we decided to proceed only with PDA 1.

5.2. Word recognition

In order to set a reasonable upper bound for the NUM task, we decided to measure human performance by having three different subjects transcribe all the utterances recorded for this task. As expected human performance is very high: we counted only one disagreement over 2577 numbers, in only one of the three listeners. As far as the NEG task is concerned, the additional phonetic difference between the words lets us assume optimal human disambiguation performance.

As explained in Section 4, once a prosodic label had been associated to a token or word transcription, we could compute the output words using task-dependent rules. In doing so we were able to determine the actual word stream of the system.

Table 3: Classification results [%] for the NEG task. Columns report results using two different training sets: in the “all data” column, all of the words in the sentences were used; in the “no & non” column, only the words no and non were used. The feature phoneme number has been discarded because it alone can determine the word class.
This same task can be afforded with a standard ASR, without explicitly considering prosodic information, but instead trying to modify the speech recognition grammars in order to allow or deny the presence of pauses in certain positions. For comparison purposes we used two grammars, G1 and G2: G1 is the baseline grammar of each task (which allows pauses before and after each word/token), G2 is a tuned grammar in which the presence of pauses was carefully imposed.

The results for the NUM task are plotted in Table 4, in terms of token and number error rates. The columns refer to three different systems: G1 and G2 do not use prosodic information, while G1+Prosody uses the most general grammar with prosodic information. As we are primarily interested in understanding what the contribution of the prosodic information, we added a second pair of rows which refer to a reduced test set. This test set contains only utterances without token recognition errors. While G1 gives very poor results in terms of number error rate (basically there is no information at all on how to connect tokens), forcing pauses reduces number error rate to 17.4%. Using explicitly prosodic information leads to 5.3%.

Likewise, for the NEG task (see Table 5) we tested two different non-prosodic systems corresponding to two different grammars G1 and G2. Then we measured the performance of two different prosodic systems, one for each different set of CARTs (see Table 3). Also for this task the introduction of prosodic information highly increases the recognition performance. The usage of all available data for training yields better performance results than using CARTs trained only on the two words no and non.

<table>
<thead>
<tr>
<th>#</th>
<th>G1</th>
<th>G2</th>
<th>G1+Prosody</th>
</tr>
</thead>
<tbody>
<tr>
<td>tokens</td>
<td>4123</td>
<td>4.0</td>
<td>4.3</td>
</tr>
<tr>
<td>numbers</td>
<td>2587</td>
<td>68.7</td>
<td>24.2</td>
</tr>
<tr>
<td>tokens</td>
<td>3187</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>numbers</td>
<td>2008</td>
<td>66.2</td>
<td>17.4</td>
</tr>
</tbody>
</table>

Table 4: Recognition results [%]: NUM task. Results are expressed as token error rate and number error rate. The first two rows refer to all the utterances (539), the last two refer to the subset that does not contain errors in terms of tokens (430 utterances). Prosodic information use is indicated by “+Prosody”.

Through prosodic word disambiguation. Prosodic information has been coded into different CARTs according to the task under analysis. The importance of different prosodic features was inspected by single features classification trials. We observed that pitch features, if coded as a polynomial curve, are difficult to handle to gain additional improvements. Experiments show that this use of prosody outperforms task-driven grammars and finally permitted to reduce the relative error rate from 42.9% (NEG) to 69.5% (NUM). We expect that further improvements could be obtained by choosing or modifying the features used, especially pitch features and pitch extraction methods. Future efforts will try to extend this approach to a more general case. Other architectures will be considered, exploiting richer ASR outputs like word graphs instead of single best, and adding a rescoring phase.

We thank Silvia Rocchi and Andrea Facco for assistance in preparing the data.

6. Conclusions

In this work we described a system that exploits prosodic information for disambiguation purposes. In particular we focused on restricted but crucial tasks for speech recognition and disambiguation within a dialogue system. We extracted prosody after the ASR stage and applied it to the output token stream to modify words transcription or to change word-boundary positions. Higher word recognition accuracy is thus obtained through prosodic word disambiguation. Prosodic information has been coded into different CARTs according to the task under analysis. The importance of different prosodic features was inspected by single features classification trials. We observed that pitch features, if coded as a polynomial curve, are difficult to handle to gain additional improvements. Experiments show that this use of prosody outperforms task-driven grammars and finally permitted to reduce the relative error rate from 42.9% (NEG) to 69.5% (NUM). We expect that further improvements could be obtained by choosing or modifying the features used, especially pitch features and pitch extraction methods. Future efforts will try to extend this approach to a more general case. Other architectures will be considered, exploiting richer ASR outputs like word graphs instead of single best, and adding a rescoring phase.

We thank Silvia Rocchi and Andrea Facco for assistance in preparing the data.

7. References


