

An Open Source Prosodic Feature Extraction Tool

Zhongqiang Huang, Lei Chen, Mary Harper

School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907
{huang37, chenl, harper}@ecn.purdue.edu

Abstract

There has been an increasing interest in utilizing a wide variety of knowledge sources in order to perform automatic tagging of speech events, such as sentence boundaries and dialogue acts. In addition to the word spoken, the prosodic content of the speech has been proved quite valuable in a variety of spoken language processing tasks such as sentence segmentation and tagging, disfluency detection, dialog act segmentation and tagging, and speaker recognition. In this paper, we report an open source prosodic feature extraction tool based on *Praat*, with a description of the prosodic features and the implementation details, as well as a discussion of its extension capability. We also evaluate our tool on a sentence boundary detection task and report the system performance on the NIST RT04 CTS data.

1. Introduction

Recently, there has been an increasing interest in utilizing a wide variety of knowledge sources in order to label speech with event tags, such as sentence boundaries and dialogue acts. In addition to the words spoken, the prosodic content of the speech can be quite valuable for accurate event tagging. Shriberg and Stolcke (Shriberg and Stolcke, 2004) have pioneered the “direct modeling” approach to exploit prosodic information in a variety of spoken language processing tasks such as sentence segmentation and tagging (Liu et al., 2004; Liu et al., 2005a), disfluency detection (Liu et al., 2005b), dialog act segmentation and tagging (Ang et al., 2005), and speaker recognition (Sonmez et al., 1998). An advantage of this approach is that no hand segmentation or intermediate labeling of the prosody is required (although if it were available it could be used). Instead the prosodic features are extracted directly from the speech signal given its time alignment to a human generated transcription or to automatic speech recognition (ASR) output. A prosody model can then be trained using these features and combined with a language model to build an event detection system.

Many of the past efforts on speech event detection utilize simple prosodic features such as pause duration (Gotoh and Renals, 2000). By contrast, the above direct modeling efforts utilize a large number of features extracted using a proprietary prosodic feature extraction suite developed at SRI (Ferrer, 2002) to good effect. SRI’s feature extraction tool is Unix script-based, combining *ESPS/Waves* for basic prosodic analysis (e.g., preliminary pitch tracking (*get_F0*) and energy computation (*get_RMS*)) with additional software components, such as a piecewise linear model (Sonmez et al., 1998) for pitch stylization. In this paper, we describe and evaluate an open source prosodic feature extraction tool¹ based on *Praat* (Boersma and Weenink, 1996) to extract a wide variety of prosodic features for event detection tasks that was inspired by the SRI suite. By creating this tool, we hope to provide a framework for building stronger baseline comparisons among systems and to sup-

port more effective sharing of prosodic features.

Figure 1 illustrates how our prosodic feature extraction tool is used by an event detection system. As illustrated in Figure 1 (a), the goal of an event detection system is to determine at a particular point in the speech (usually a word boundary given forced alignment) whether to label it with an event (e.g., sentence unit boundary) or not. Figure 1 (b) shows an event tagging system that combines a language model (LM) with a prosody model, whose features are provided by our prosodic feature extraction system shown in Figure 1 (c). The rest of the paper is organized as follows. We briefly describe the prosodic features implemented in our tool in section 2., and present the implementation details and discuss some potential extension directions in section 3.. In section 4. we evaluate the tool on the sentence boundary detection task using the NIST RT04 CTS data. The last section concludes our work.

2. Feature Description

The features described here have been inspired by SRI’s prosodic feature extraction suite (Ferrer, 2002), however, our implementation differs in that we utilize the functionality of *Praat*. For example, pitch tracking is performed by *Praat*’s built-in function rather than using *ESPS/Waves*, and the pitch stylization is accomplished by *Praat*’s built-in function, rather than the piecewise linear stylization algorithm in SRI’s model. Since currently we mainly focus on various types of boundary detection tasks, all of the prosodic features are extracted around each word boundary.

- **Duration features:** Duration features are obtained based on the word and phone alignments of human transcriptions (or ASR output). Pause duration and its normalization after each word boundary are extracted. We also measure the duration of the last vowel and the last rhyme, as well as their normalizations, for each word preceding a boundary. The duration and the normalized duration of the word are also included as duration features.
- **F₀ features:** *Praat*’s autocorrelation based pitch tracker is used to obtain raw pitch values. The pitch

¹It can be downloaded at <ftp://ftp.ecn.purdue.edu/> harper, along with a manual (Z. Huang, 2006).

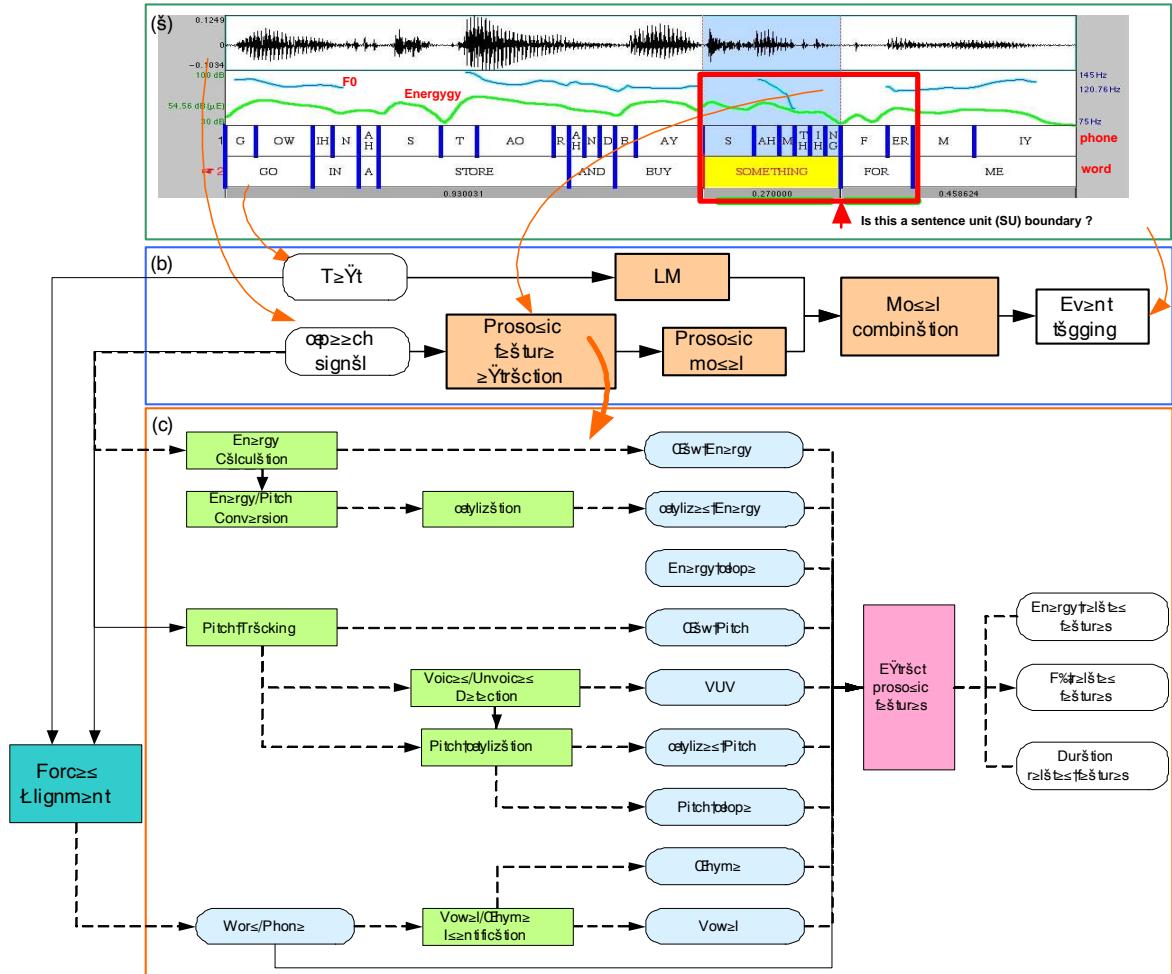


Figure 1: A scheme for combining words and prosodic content for event tagging. (a) A sentence boundary detection task. (b) An integration diagram of a prosodic model and a language model (LM) for event tagging. (c) The procedures of prosodic feature extraction. Note that Forced Alignment is not a part of the tool, and so appears outside of box c.

baseline and topline, as well as the pitch range, are computed based on the mean and variance of the logarithmic F_0 values. Voiced/unvoiced (VUV) regions are identified and the original pitch contour is stylized over each voiced segment. Several different types of F_0 features are computed based on the stylized pitch contour.

- **Range features:** These features reflect the pitch range of a single word or a window preceding or following a word boundary. These include the minimum, maximum, mean, and last F_0 values of a specific region (i.e. within a word or window) relative to each word boundary. These features are also normalized by the baseline F_0 values, the topline F_0 values, and the pitch range using linear difference and log difference.
- **Movement features:** These features measure the movement of the F_0 contour for the voiced regions of the word or window preceding and the word or window following a boundary. The minimum, maximum, mean, the first, and the last

stylized F_0 values are computed and compared to that of the following word or window, using log difference and log ratio.

– **Slope features:** Pitch slope is generated from the stylized pitch values. The last slope value of the word preceding a boundary and the first slope value of the word following a boundary are computed. We also include the slope difference and dynamic patterns (i.e. *falling*, *rising*, and *unvoiced*) across a boundary as slope features, since a continuous trajectory is more likely to correlate with non-boundaries; whereas, a broken trajectory tends to indicate a boundary of some type.

- **Energy features:** The energy features are computed based on the intensity contour produced by *Praat*. Similar to the F_0 features, a variety of energy related range features, movement features, and slope features are computed, using various normalization methods.
- **Other features:** We add the gender type to our feature set. Currently the gender information is provided in

a metadata file, rather than obtaining it via automatic gender detection.

3. Implementation and Extension

We have implemented the prosodic feature extraction tool using *Praat*'s programmable scripting language (Boersma and Weenink, 1996). A very important reason that we chose *Praat* as our platform is that it provides an existing suite of high quality speech analysis routines, such as pitch tracking. Additional reasons for using *Praat* include:

1. *Praat* is a public domain, widely used speech analysis toolkit that is supported on a variety of platforms (e.g., Windows, Macintosh, Linux, and Solaris).
2. It provides a variety of value data structures, such as *TextGrid*, *PitchTier*, and *Table*, to represent various types of information used for extracting prosodic features.
3. It provides a built-in programmable scripting language for calling *Praat*'s commands and extending its capability.
4. Additions to *Praat* functionality can be immediately adopted into the tool. This is especially useful for incorporating new prosodic features.

Hence, *Praat* is an ideal platform for building a public domain prosodic feature extraction tool that can be used and extended by a wide variety of researchers. In the rest of this section we describe our tool implementation and discuss some potential ways in which the tool could be extended.

3.1. Implementation

Given a corpus with audio and time aligned words and phones as input, our tool first extracts a set of basic elements (e.g., raw pitch, stylized pitch, VUV) representing duration, F_0 , and energy information, as is shown in Figure 1 (c). Then a set of duration statistics (e.g., the means and variances of pause duration, phone duration, and last rhyme duration), F_0 related statistics (e.g., the mean and variance of logarithmic F_0 values), and energy related statistics are calculated. Given the duration, F_0 , and energy information, as well as the statistics, it is straightforward to extract the prosodic features at each word boundary, according to the definition of features in (Ferrer, 2002) and our tool documentation (Z. Huang, 2006). We describe below how to obtain and represent these basic elements in *Praat*. Table 1 summarizes their use in the computation of prosodic features.

- **Word and Phone Alignments:** A forced alignment system² is used to determine the starting and ending times of words and phones. In our tool these alignments are represented in *TextGrid IntervalTiers*.

²This is separate from our feature extraction tool. Researchers can choose from a variety of systems, such as Aligner (Wightman and Talkin, 1997), ISIP ASR (Sundaram et al., 2000), and SONIC (Pellom, 2001).

Table 1: The use of raw files for extracting various features

	Duration Features	F_0 Features	Energy Features
Word	✓	✓	✓
Phone	✓	✗	✗
Vowel	✓	✗	✗
Rhyme	✓	✗	✗
VUV	✗	✓	✗
Raw Pitch	✗	✓	✗
Stylized Pitch	✗	✓	✗
Pitch Slope	✗	✓	✗
Raw Energy	✗	✗	✓
Stylized Energy	✗	✗	✓
Energy Slope	✗	✗	✓

- **Vowel and Rhyme:** The starting and ending times of vowels and rhymes are determined from the phone alignments. As to rhyme, we only consider the last rhyme, which is defined to be the sequence of phones starting from the last vowel and covering all the remaining phones in a word. Vowels and rhymes are also represented in *TextGrid IntervalTiers*.
- **F_0 :** We rely on *Praat*'s autocorrelation based pitch tracking algorithm to extract raw pitch values, using gender dependent pitch range. The raw pitch contour is smoothed and the voiced/unvoiced regions are determined and stored in a *TextGrid IntervalTier*. *Praat*'s pitch stylization function is used to stylize raw F_0 values over each voiced region. Both raw F_0 values and stylized F_0 values are represented in *PitchTiers*. The pitch slope values are generated based on the stylized pitch contour, and are stored in a *TextGrid IntervalTier*.
- **Energy:** Intensity values are computed for each frame and stored in an *IntensityTier*. Since there is no intensity stylization function in *Praat*, we choose to represent intensity values in a *PitchTier*, and apply the pitch stylization function to stylize the intensity contour. Note that stylization is performed on the entire intensity contour, in contrast to the pitch case for which this applies only in voiced regions. The raw and stylized intensity values are stored in *PitchTiers*, and the slope values are stored in a *TextGrid IntervalTier*.

3.2. Extension

As we discussed above, the major advantage of building a prosodic model based on *Praat* is the capability of taking advantage of *Praat*'s existing built-in speech analysis algorithms and other *Praat* scripts that have been written as extensions, as well as those that would be added to *Praat* in the future. Although features we have implemented have been used for a variety of event detection tasks, they are not necessarily equally effective for all tasks. Hence, it is important to have flexibility to easily add new features into the system.

Take jitter and shimmer for example, jitter is the cycle-to-cycle variability in frequency of vocal fold vibration,

and shimmer is the cycle-to-cycle variability in amplitude of the vocal fold vibration. These two measurements are well known as voice quality measurements for pathological speech. Liu (Liu, 2003) used prosodic features (i.e. duration, pitch, and energy features) and some voice quality measures for word fragment identification in conversational speech, and found that jitter was the most frequently queried voice quality feature in the decision tree. There are existing algorithms in *Praat* for computing both jitter and shimmer, therefore it would be a simple matter to implement a jitter and shimmer extraction script and add it into our tool. As a matter of fact, Liu obtained the jitter values right in *Praat*.

Take Fujisaki analysis for another example. The Fujisaki model (Fujisaki and Kawai, 1988) is a parametric superpositional model widely used in the area of speech synthesis. In this model, the fundamental frequency contour is decomposed into phrase components and accent components over the base frequency. One can imagine that the phrase components are informative of phrase or sentence boundaries, thus could be used in some boundary detection tasks. Although Fujisaki analysis has yet to be implemented in *Praat*, when it does, it would be a simple matter to add a Fujisaki analysis into our tool in order to provide an alternative set of features.

4. Evaluation

4.1. Setup

In order to examine the effectiveness of our tool, i.e. the performance of the features extracted by the tool on event tagging tasks, we evaluated it on SU detection task using NIST RT04 CTS data. The RT04 CTS data has a training set drawn from the Switchboard corpus, two development sets with one drawn from both Switchboard and Fisher and another from Fisher, and a eval set drawn from Fisher. To evaluate the in-domain and out-of-domain effects, we have chosen to train our prosody model on the entire training set and evaluate on the Switchboard part of the first development set and the Fisher data in the eval set. Table 2 summarizes these data used in our experiments. SONIC’s aligner was used to produce the word and phone alignments given the word boundaries in the official NIST RT04 RTTM files as reference. Pronunciations were created for all out of vocabulary (OOV) words.

	SUs/Words	Source
train	64K/480K	Switchboard
dev	6K/35K	Switchboard
eval	5K/34K	Fisher

Table 2: The size and source of experiment data.

A standard CART style decision tree was used to train the prosody models (an example tree is given in Figure 2). In order to address the imbalanced data problem (since there are fewer SU events than non-events at interword boundaries; overall only 13.3% word boundaries in the training set are also sentence boundaries). we used a down-sampled training set in which SU and non-SU classes had

equal prior probabilities. Additionally, we employed ensemble bagging to reduce the variance of the prosodic classifiers. Using this method, several random downsampled training sets were generated, and each was resampled multiple times and corresponding classifiers were combined via bagging³. This has been found to improve the performance of the prosody model on the CTS SU task (Liu et al., 2005a; Liu, 2004).

```

PAUSE_DUR < 3.5:
| FOK_WIN_DIFF_LOLO_NG < 0.0093418:
| | WORD_DUR < 35.5:
| | | FOK_WORD_DIFF_LOLO_NG < 0.0070514:
| | | | LAST_RHYME_NORM_DUR_PH_ND < -14.353: 1
| | | | LAST_RHYME_NORM_DUR_PH_ND >= -14.353:
| | | | | AVG_PHONE_DUR_Z < 0.033767: 0
| | | | | AVG_PHONE_DUR_Z >= 0.033767:
| | | | | WORD_DUR < 19.5: 0
| | | | | WORD_DUR >= 19.5: 1
| | | | FOK_WORD_DIFF_LOLO_NG >= 0.0070514: 0
| | WORD_DUR >= 35.5:
| | | FOK_WORD_DIFF_LOLO_NG < 0.0071925: 1
| | | FOK_WORD_DIFF_LOLO_NG >= 0.0071925:
| | | | FOK_WIN_DIFF_LOHL_N < -0.22393: 1
| | | | FOK_WIN_DIFF_LOHL_N >= -0.22393:
| | | | | FOK_WORD_DIFF_ENDBEG < 0.047089:
| | | | | AVG_VOWEL_DUR_ZSP < 1.512: 0
| | | | | AVG_VOWEL_DUR_ZSP >= 1.512: 1
| | | | FOK_WORD_DIFF_ENDBEG >= 0.047089: 1
| | | FOK_WIN_DIFF_LOLO_NG >= 0.0093418:
| | | | LAST_RHYME_NORM_DUR_PH_ND < -15.931: 0
| | | | LAST_RHYME_NORM_DUR_PH_ND >= -15.931: 1
PAUSE_DUR >= 3.5:
| PAUSE_DUR < 42.5:
| | WORD_DUR < 28.5:
| | | FOK_WORD_DIFF_HIHI_NG < 0.0092524:
| | | | FOK_DIFF_LAST_KBASELN < 99.307:
| | | | | FOK_DIFF_MEAN_KBASELN < 14.179: 1
| | | | | FOK_DIFF_MEAN_KBASELN >= 14.179: 0
| | | | FOK_DIFF_LAST_KBASELN >= 99.307: 1
| | | | FOK_WORD_DIFF_HIHI_NG >= 0.0092524: 1
| | WORD_DUR >= 28.5: 1
| PAUSE_DUR >= 42.5: 1

```

Figure 2: An example of a decision tree for SU detection. Each line represents a node in the tree, with the associated question regarding a particular feature (e.g., PAUSE_DUR is the pause duration after a word boundary, FOK_WIN_DIFF_LOLO_NG is the log ratio between the minimum of the stylized F_0 within a window before a word boundary and the minimum of the stylized F_0 within a window after a word boundary, normalized by the pitch range). The ‘0’ (non-SU) or ‘1’ (SU) at the end of a line (representing a leaf note) denotes the decision at the corresponding node.

To measure the performance of our models, we used the *Error Rate* metric defined by NIST for the DARPA EARS metadata evaluation for comparison with the literature. The *Insertion Rate* and *Deletion Rate* are also provided to determine whether there are different patterns of insertions and deletions among different feature sets. The three metrics are defined below:

1. *Insertion Rate* = #Insertions / #SUs in the reference

³The posterior probabilities provided by decision tree models are normalized to reflect the distribution of SU/non-SU boundaries in the training set.

2. *Deletion Rate* = #Deletions / #SUs in the reference
3. *Error Rate* = (#Deletions + #Insertions) / #SUs in the reference

4.2. Results and Discussion

We started with all of the prosodic features (more than two hundred) extracted by our tool, and compared in Table 3 the performance of the decision tree model trained using these features to a simple baseline that always selected the majority class (i.e., non-SU). We also included for comparison the performance of using only pause duration (pause_dur) after a word boundary. As can be observed, using prosodic features improves performance considerably over the chance baseline, and also reduces overall error rate compared to using pause duration alone. Although the improvement on the eval set using the full feature set against pause duration alone is greater than that on the dev set, the prosody model seems overfitted to the training data which was drawn from the Switchboard corpus since it performs significantly worse on the eval set than on the dev set. This may be caused by the difference between the Switchboard corpus and the Fisher corpus. For example, the former was created at an earlier time, and was carefully transcribed; whereas, the latter was transcribed using a faster but less accurate transcription guideline.

Evaluation on dev set			
	Deletion	Insertion	Error
Baseline	100	0	100
pause_dur	41.72	17.75	59.47
full set	51.45	5.88	57.33

Evaluation on eval set			
	Deletion	Insertion	Error
Baseline	100	0	100
pause_dur	41.93	27.10	69.03
full set	56.42	6.17	62.59

Table 3: SU detection results using the full set of prosodic features, pause duration only, and a baseline obtained by considering all word boundaries to be non-events.

As pointed out in (Shriberg et al., 2000), the full feature set contains a high degree of feature redundancy, and may result in a suboptimal tree due to the greedy nature of the decision tree. We found that many unnormalized F_0 and energy features were rarely queried in the final decision trees. In the second experiment, we started with about 100 features comprised of most of the duration features, the normalized F_0 features, the normalized energy features, and gender type, and tested different combinations of these features (with gender type always included). The results are provided in Table 4.

The first observation is that these relatively small subsets of features perform comparably to or better than using the full feature set with respect to the overall *Error Rate*. Among the performance of these different combinations, a lower overall *Error Rate* is largely contributed by a lower *Deletion Rate*. Combining F_0 features with the duration features could slightly reduce the *Error Rate* ($p < 0.061$ in sign test); whereas, combining energy features with the

Evaluation on dev set			
	Deletion	Insertion	Error
Baseline	100	0	100
dur	49.41	6.71	56.13
dur+ F_0	47.08	8.14	55.23
dur+energy	51.94	6.01	57.94
dur+ F_0 +energy	49.96	6.60	56.56

Evaluation on eval set			
	Deletion	Insertion	Error
Baseline	100	0	100
dur	49.98	9.87	59.85
dur+ F_0	47.52	11.56	59.08
dur+energy	52.80	8.01	60.82
dur+ F_0 +energy	51.31	8.88	60.19

Table 4: SU detection results using different combinations of prosodic features.

duration features impairs the performance somewhat ($p < 0.02$ in sign test), and the performance degradation of using dur+energy compared to using dur+ F_0 is even more significant ($p < 0.0003$ in sign test). However, it is incorrect to conclude that certain features such as the energy related features are not useful at all for several reasons. First, as explained by Shriberg and Stolcke (Shriberg et al., 2000), these features may be useful on their own, but less useful when combined with other features. Second, the effectiveness of a certain feature varies across different corpora and tasks. Shriberg and Stolcke reported in (Shriberg et al., 2000) that, F_0 features contributed 11% feature usage⁴ in sentence segmentation using Broadcast News data and 36% for topic segmentation; whereas, F_0 features were not among the feature set identified by their feature selection algorithm in the sentence segmentation task using Switchboard data. Liu reported in (Liu, 2003) that both energy and F_0 features were effective for word fragment detection. Third, there is variance in training and testing, and different decision tree training algorithms and different statistical models may use these features differently.

Our system has around a 60% *Error Rate* on the RT04 evaluation set, which is comparable to the best result for the RT03 evaluation set reported by Liu (Liu, 2004) using SRI's prosodic feature extraction suite. Rather than being used alone, a prosody model is usually integrated with a language model for better performance. To complete this evaluation, the best prosody model (PM) in Table 4 is combined with a language model (PM) trained on the training set of RT04 CTS data. Results are given in Table 5. The integration of the prosody model with the language model could significantly reduce the *Insertion Rate* and thus the overall *Error Rate*, compared to the respective prosody and language models alone. It was also shown by Chen (Chen et al., 2004) that adding gesture and gaze features could further improve system performance on the sentence unit detection task.

⁴Feature usage (Shriberg et al., 2000) is a measure computed as the relative frequency with which that feature or feature class is queried in the decision tree.

Evaluation on dev set			
	Deletion	Insertion	Error
Baseline	100	0	100
PM	47.08	8.14	55.23
LM	31.21	12.63	43.84
PM + LM	31.58	4.42	36.00
Evaluation on eval set			
	Deletion	Insertion	Error
Baseline	100	0	100
PM	47.52	11.56	59.08
LM	30.83	14.65	45.48
PM + LM	31.94	5.95	37.89

Table 5: SU detection results combining a prosodic model with a language model

5. Conclusion

In this paper, we have reported our open source prosodic feature extraction tool built based on *Praat*. This tool is capable of extracting a variety of duration features, F_0 features, and energy features given an audio recording and its time aligned words and phones sequences. Implementation details are described and extension capabilities are discussed. Evaluation of the prosody model derived from this tool and its integration with a language model has demonstrated the effectiveness of our tool on the sentence boundary detection task using NIST RT04 CTS data. In the future, we will apply this tool to other event tagging applications, and will also keep developing this tool and make it easily extensible for research communities.

6. Acknowledgments

This work was supported in part by ARDA under contract number MDA904-03-C-1788. Any opinions, findings, and conclusions expressed in this paper are those of the authors and do not necessarily reflect the views of ARDA. We would also like to acknowledge Yang Liu for sharing her experts regarding prosodic features and evaluation experiments.

7. References

- Ang, J., Y. Liu, and E. Shriberg, 2005. Automatic dialog act segmentation and classification in multiparty meetings. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*. Philadelphia, PA.
- Boersma, P. and D. Weeninck, 1996. Praat, a system for doing phonetics by computer. Technical Report 132, University of Amsterdam, Inst. of Phonetic Sc.
- Chen, L., Y. Liu, M. Harper, and E. Shriberg, 2004. Multimodal model integration for sentence unit detection. In *Proc. of Int. Conf. on Multimodal Interface (ICMI)*. University Park, PA.
- Ferrer, L., 2002. Prosodic features extraction. Technical report, SRI.
- Fujisaki, H. and H. Kawai, 1988. Realization of linguistic information in the voice fundamental frequency contour of the spoken Japanese. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 1.
- Gotoh, Y. and S. Renals, 2000. Sentence boundary detection in broadcast speech transcript. In *Proc. of the Intl. Speech Communication Association (ISCA) Workshop: Automatic Speech Recognition: Challenges for the new Millennium ASR-2000*.
- Liu, Y., 2004. *Structural Event Detection for Rich Transcription of Speech*. Ph.D. thesis, Purdue University.
- Liu, Y., N. Chawla, E. Shriberg, A. Stolcke, and M. Harper, 2005a. Resampling techniques for sentence boundary detection: A case study in machine learning from imbalanced data for spoken language processing. *To appear in Computer Speech and Language*.
- Liu, Y., E. Shriberg, A. Stolcke, and M. Harper, 2005b. Comparing HMM, maximum entropy, and conditional random fields for disfluency detection. In *INTERSPEECH*. Lisbon Spain.
- Liu, Y., A. Stolcke, E. Shriberg, and M. Harper, 2004. Comparing and combining generative and posterior probability models: Some advances in sentence boundary detection in speech. In *Proceedings of the Empirical Methods in Natural Language Processing*.
- Liu, Yang, 2003. Word fragment identification using acoustic-prosodic features in conversational speech. In *HLT/NAACL 2003 Student Workshop*.
- Pellom, B., 2001. SONIC: The University of Colorado continuous speech recognizer. Technical Report TR-CSLR-2001-01, University of Colorado.
- Shriberg, E. and A. Stolcke, 2004. Direct modeling of prosody: An overview of applications in automatic speech processing. In *International Conference on Speech Prosody*.
- Shriberg, E., A. Stolcke, D. Hakkani-Tur, and G. Tur, 2000. Prosody-based automatic segmentation of speech into sentences and topics. *Speech Communication*:127–154.
- Sonmez, K., E. Shriberg, L. Heck, and M. Weintraub, 1998. Modeling dynamic prosodic variation for speaker verification. In *Proc. of Int. Conf. on Spoken Language Processing (ICSLP)*.
- Sundaram, R., A. Ganapathiraju, J. Hamaker, and J. Picone, 2000. ISIP 2000 conversational speech evaluation system. In *Speech Transcription Workshop 2001*. College Park, Maryland.
- Wightman, C. and D. Talkin, 1997. *The Aligner*. Entropic.
- Z. Huang, M. Harper, L. Chen, 2006. *Purdue Prosodic Feature Extraction Toolkit on Praat*. Spoken Language Processing Lab, Purdue University, <ftp://ftp.ecn.purdue.edu/harper/>.