A New Prosodic Phrasing Model for Chinese TTS Systems

Weijun Chen, Fuzong Lin, Jianmin Li, Bo Zhang
Department of Computer Science and Technology
Tsinghua University
Haidian District, Beijing, CHINA
cwj@s1000e.cs.tsinghua.edu.cn

Abstract

This paper proposes a new prosodic phrasing model for Chinese text-to-speech systems. First, in contrast to the commonly used CART techniques, we propose a new inductive learning algorithm based on the extension matrix theory. Second, we collected 559 sentences (of approximately 78 min length) from news programs and built a corresponding speech corpus uttered by a professional male announcer. The prosodic boundaries were manually marked and word identification, POS tagging and syntactic analysis were also done on the text. Finally, our model was trained on 371 sentences and tested on 188. We achieved a success rate of about 93%.

1 Introduction

Assigning appropriate phrasing in text-to-speech systems is very important both for naturalness and for intelligibility, particularly in longer sentences. Generally speaking, fluent spoken language is not produced in a smooth, unvarying stream, people tend to group words into phrases and place short pauses between them. Furthermore, variation in phrasing can change the meaning hearers assign to the utterances of a given sentence. Researchers have shown (Ostendorf, 1993) that the relative size and location of prosodic boundaries provides an important cue for resolving syntactic ambiguity.

Traditional research on the location of phrase boundaries has focused primarily on the relationship between prosodic structure and syntactic structure, and uses some sort of syntactic information to predict prosodic boundaries (often in the form of heuristic rules) (Bachenko, 1990; Willemse, 1991). However, such hand-crafted rules systems are notoriously difficult to write, maintain and adapt to new domains and languages. To avoid these problems, recently efforts have been concentrated on acquiring phrasing rules automatically by training self-organizing procedures on large prosodically labeled corpora. For these data-driven methods, the prediction of phrase boundary location is seen as a classification problem, and the main differences between them lie in two aspects: the learning model and the feature set. The primary learning techniques currently used include hidden markov models (Taylor, 1998), neural networks (Muller, 1996), classification and regression trees (CART) (Wang, 1992; Hirschberg, 1996; Lee, 1999) and transformational rule-based learning (TRBL) (Fordyce, 1998).

For Chinese, the precise syntactic analysis of the sentence structure is difficult and large amount of computation required. So most of the previous TTS systems just identify the words in the input texts and no more prosodic information is extracted for further processing. In this paper, we propose a prosodic phrasing model based on a new family of learning algorithms: the extension matrix approach. In contrast to the decision tree in CART algorithm, a new kind of matrix, called an extension matrix, was taken as the mathematical basis. The extension matrix approach was first developed by Hong (1985) in his AE1 algorithm and then redesigned by Wu (1998) in his HCV algorithm. This paper improves upon Wu's work and proposes a new heuristic inductive algorithm HGR (Heuristic Grouping).

2 Original Extension Matrix Algorithms

2.1 Terminology and Notations

Let $E = D_1 \times \ldots \times D_a$ be an event space, here a is the number of attributes $\{ x_1, \ldots, x_a \}$ and D_j

is a finite set of integers. The *events* are in the form of $< x_1 = v_1, \ldots, x_a = v_a >$, where v_j is the *j*th *element* of an event and $v_j \in D_j$. *PE* and *NE* are two subsets of *E*, called the *set of positive events* and the *set of negative events* respectively, and $p = |PE|$, $n = |NE|$. A *selector* is a statement either of the form $[x_j = A_j]$ or $[x_j \neq A_j]$, where $A_j \subseteq D_j$. A *complex* is a conjunction of selectors, and a *cover* is a disjunction of complexes. Let *NE* be expressed by

$$NEM = (e_1^-, \ldots, e_n^-)^T = (r_{ij})_{n*a} \qquad (1)$$

with the *i*-th negative event $e_i^- \ (i = 1, \ldots, n)$ being expressed on the *i*-th row of matrix *NEM* and $NEM(i, j) = r_{ij}$ indicating that the value of e_i^- on attribute x_j is r_{ij}.

Definition 1. A positive event e^+ is covered by a complex *L* against *NE*, if and only if *L* covers e^+ and does not cover any event in *NE*.

Definition 2. Given a positive event $e^+ = < v_1^+, \ldots, v_a^+ >$, the matrix below is the *extension matrix* of e^+ against *NE*

$$EM(e^+) = (r_{ij}^{'})_{n*a} \qquad (2)$$

where

$$r_{ij}^{'} = \begin{cases} * & \text{when } v_j^+ = NEM_{ij} \\ NEM_{ij} & \text{when } v_j^+ \neq NEM_{ij} \end{cases}$$

and '*' denotes a *dead element* that cannot be used to distinguish e^+ from the negative events.

Definition 3. In $EM(e^+)$, a set of *n* nondead elements $r_{ij_i}^{'} (i = 1, \ldots, n, \ j_i \in \{1, \ldots, a\})$ that come from the *n* different rows is a *path*. Each path in $EM(e^+)$ corresponds to a conjunctive complex

$$L = \bigwedge_{i=1}^{n} [x_{j_i} \neq r_{ij_i}^{'}] \qquad (3)$$

which covers e^+ against *NE*.

Definition 4. The corresponding nondead elements in the different extension matrices that have the same values are the *common elements* of the matrices; a path that consists only of common elements is a *common path* of the extension matrices involved.

Now, the problem of learning is to obtain a general cover, i.e. a disjunction of complexes, which covers all the positive events against *NE*.

2.2 An Example

Given 8 events of class *T* and *F* in Table 1. Each event is described by three attributes: x_1 {0, 1}, x_2 {0, 1, 2}, and x_3 {0, 1, 2}.

Table 1. Events of class *T* and *F*.

ORDER	x_1	x_2	x_3	**CLASS**
1	0	0	0	T
2	0	0	2	T
3	1	2	0	T
4	1	0	0	T
5	1	0	1	F
6	0	1	0	F
7	1	1	0	F
8	1	1	2	F

In this example, if we were considering for class *T*, the first four events would be the set of positive events, while the last four events would be the set of negative events. A statement such as $[x_1 = 0]$ is a selector, and a rule such as $[x_1 = 0] \wedge [x_2 = 0]$ is a complex.

The extension matrices of the first two positive events e_1 and e_2 are shown in Figure 1.

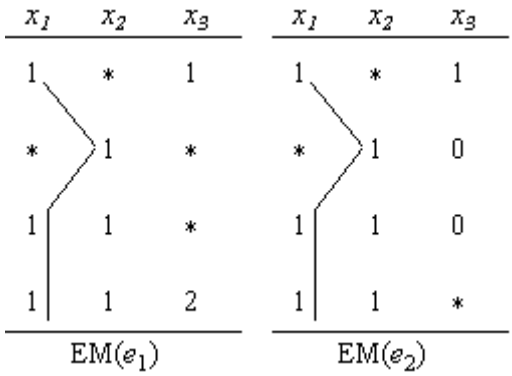

Figure 1. Extension Matrices of e_1 and e_2.

In Figure 1, $\{x_1 = 1, x_2 = 1, x_1 = 1, x_1 = 1\}$ (the segmented line) is a common path of the two extension matrices involved, and it corresponds to a conjunctive complex $[x_1 = 0] \wedge [x_2 = 0]$ which covers both e_1 and e_2.

2.3 AE1 and HCV Algorithms

It is desired that the cover generated contains fewer complexes, and each complex contains fewer selectors. Hence, the following optimization problems arise.

1. MCOMP problem. Generating a complex that has the minimum number of selectors and whose variables take the minimum total number of values; such complexes are called *minimal complexes*.
2. MCV problem. Generating a cover that has the minimum number of complexes.

However, both the MCOMP and the MCV problems had been proved to be NP-hard (Hong, 1985). So we have to stop trying to search the optimal solution, but to resort to heuristic search algorithm to get some approximate approaches.

In AE1, the heuristic approach firstly search the path within every extension matrix, using a simple strategy of starting search from the columns with the most nondead elements, and then get an approximate solution for MCV problem by repeatedly selecting the most common path, i.e. the complex that covers a maximum number of positive events, until all events in *PE* are covered.

In HCV, the disjunction matrix was used as the basis of the algorithm. The disjunction matrix *EMD* of a set of positive events $\{e_1^+, \ldots, e_k^+\}$ is the disjunction matrix of $EM(e_1^+), \ldots, EM(e_k^+)$, i.e., if there exist a $k_1 \in \{1, \ldots, k\}$, $EM_{ij}(e_{k_1}^+) =$ '*', then $EMD_{ij} =$ '*'; otherwise $EMD_{ij} = NEM_{ij}$. Similarly, each path in the *EMD* corresponds to a complex that covers $\{e_1^+, \ldots, e_k^+\}$ against *NE*. The basic idea for HCV algorithm is summarized as follows:

- First, partition the set of positive events *PE* into p' $(p' \le p)$ intersecting *groups* by using a simple greedy algorithm: keep on adding new positive events to a group until there is no path exists in the *EMD* of the group.
- Second, call the heuristic algorithm HFL to find a complex for each intersecting group. Four strategies are applied in turn to make the generated complex as short as possible. In fact, three of these strategies were proved to be complete.
- Last, the final cover is generated by logically ORing all the complexes.

HCV has provided a reasonable solution to both the MCOMP and the MCV problems, especially the first one. However there are still two problems. First, the greedy grouping algorithm is too simple. In some cases, a group may fail to cover as many positive events as possible. Second, although the complex generated by the HFL algorithm is very likely to be the shortest one, there is a lack of methods to evaluate the correctness of the complex at induction time. In fact, although all the complexes get a 100% success rate on the training data, some of them may perform poorly on test data, especially when the complex covers fewer examples. The HGR algorithm in the next section is intended to solve these problems.

3 HGR Algorithm

The HGR algorithm consists of two modules: GROUP module and COMP module. Our basic idea is the same as HCV algorithm: first partition *PE* into some intersecting groups using the GROUP module, and then find a complex for each group using the COMP module. In fact, these two modules can be combined with those of HCV separately to generate a new algorithm.

3.1 GROUP Module

A good grouping algorithm is very important for an extension matrix approach; it should be able to distribute more events to fewer groups. More events in a group means a shorter complex might be generated and the complex is more likely to approximate to the truth, and fewer groups means the final cover will contain less number of complexes.

Definition 5. Two positive events e_1^+ and e_2^+ are compatible against *NE* if there exist a common path in $EM(e_1^+)$ and $EM(e_2^+)$, and the matrix below is the compatibility matrix of *PE* against *NE*.

$$CM = (c_{ij})_{p*p} \tag{4}$$

where

$$c_{ij} = \begin{cases} 1 & \text{if } e_i^+ \text{ is compatible with } e_j^+ \\ 0 & \text{otherwise} \end{cases}$$

It is obviously that *CM* is a symmetric matrix and $c_{ii} = 1$. If e_1^+ is compatible with e_2^+, we say e_1^+ is a *friend* of e_2^+.

Theorem 1. The members of a group are compatible with each other.

Proof. Suppose the group is composed of $\{e_1^+, \ldots, e_k^+\}$, then there should exist at least one path L in the disjunction matrix *EMD* of the group, and it is obviously that L is also a common path of $EM(e_1^+), \ldots, EM(e_k^+)$.

The Theorem 1 inspired us to propose the following grouping algorithm:

Procedure GROUP (EM's, CM)
S0: Subset ← Φ, Group ← Φ
S1: add all the positive events that have not been covered to Subset;
S2: Seed ← the event that has largest number of friends in Subset;
S3: EM ← EM (Seed),
Group ← Group ∪ {Seed};
S4: delete Seed and the events that are not friends of Seed from Subset;
S5: Popular ← the event that has largest number of friends in Subset;
S6: EM2 ← EM (Popular);
S7: EM2 ← EM ∪ EM2
S8: if there exist at least one path in EM2 {EM←EM2, Group←Group∪{Popular} Seed ← Popular, go to S4}
S9: return (Group);

The time complexity for GROUP algorithm is $O(p^3 + nap)$.

3.2 COMP Module

Instead of using the HFL algorithm of HCV, we propose a new method to find a complex for each group. Currently we only consider the selectors in form of $[x_j = v]$, where $v \in D_j$. A selector is thought to be "good" if it covers more positive events in the group and rejects more negative events in *NE*. The basic idea of the COMP algorithm is to repeatedly find "good" selectors and use them to generate a conjunctive complex until the accuracy of the complex on training data has reached a specified threshold.

Procedure COMP (Group)
S0: Complex ← Φ;
S1: Size1 ← size of Group, Size2 ← size of *NE*, BestScore← 0, BestSelector← Φ;
S2: find a new selector S that occured in the Group, if fail to do so, go to S7;
S3: Num1 ← the number of events that S covers in Group;
S4: Num2 ← the number of events that S rejects in NE;
S5: Score ← Num1/Size1 + Num2/Size2;
S6: if Score > BestScore then
{ BestScore ← Score, BestSelector ← S, go to S2; }
S7: Complex ← Complex ∧ BestSelector;
S8: delete the events that BestSelector covers and rejects from Group and *NE*;
S9: test Complex on training data, if the accuracy is below a threshold, go to S1;
S10: return (Complex);

The time complexity of COMP is $O(da^2(p+n))$, $d = \max\{|D_1|, \ldots, |D_a|\}$.

3.3 Preliminary Experiments

A serial of experiments have been conducted to evaluate the performance of our algorithm. All the databases used in the experiments were copied from the University of California at Irvine machine learning database repository (Murphy, 1995), the famous database repository for machine learning community. The majority of these databases were obtained from real world and are noisy. The performance comparisons of HGR with other induction algorithms are summarized in Table 2. C4.5 is the recent successor of the famous ID3-like algorithms and CN2 is a successor of the AQ family algorithms. The best result for each problem is highlighted with **boldface** font in the table.

Table 2. Experimental results of four algorithms.

Database	C4.5	CN2	HCV	HGR
Audiology	80.8	80.8	73.1	**84.6**
Hayes-roth	89.3	71.4	89.3	**92.9**
Monk1	78	98.6	**100**	**100**
Monk2	65.3	75.7	**85.9**	78.5
Monk3	92.6	90.7	87	**93.1**
Tic-tac-toe	93.7	**98.4**	97.2	97.8
Vote	94.8	95.6	97	**97.8**
Mux6	**100**	**100**	**100**	**100**
Vote-irvine	**95.2**	**95.2**	93.8	**95.2**
Average	87.74	89.6	91.48	**93.32**

Table 2 shows that HGR obtained the best results on more databases than any other algorithms.

4 Prosodic Phrasing

4.1 Corpus

For a data-driven method, the scale and quality of the corpus is important. Since there is no suitable corpus available for modeling Chinese prosody, we collected 559 sentences (of approximately 78 min length) from TV news program and built a corresponding speech corpus uttered by a famous male announcer. We found that the transcription of punctuation did not seem to exactly match the original written text, so we collected the original script of each piece of news from newspapers and other sources. And then, the annotation of this data for analysis was conducted in the following steps:

First, although a Chinese word is composed of one to several characters, a Chinese sentence is in fact a string of characters without blanks to mark the word boundaries. Therefore the first step is to identify the words in the text corpus. This task was accomplished using a simple segmentation algorithm and the errors were corrected.

Second, the part-of-speech information of each word was obtained with a Chinese POS tagger, whose output had been modified to adapt to the characteristic of prosody. We designed 72 POS's which belong to 20 word classes.

Third, the syntactic analysis was done manually. To describe the structure of a sentence, we propose a simplified Chinese grammar. The grammatical functions we considered include subject, predicate, object, modifier and complement. The syntactic phrasal units include noun phrases, prepositional phrases, adjectival phrases, etc.

Lastly, we labeled the speech prosodically by hand, noting location and type of boundaries and accents. We labeled two levels of boundary, major and minor; in the analysis presented below, however, these are collapsed to a single category.

4.2 Features

First of all, we define the problem of prosodic phrasing as follows: the input sentence consists of a sequence of words and between each pair of adjacent words is a word juncture $< w_i , w_{i+1} >$, where w_i represents the word to the left of the juncture and w_{i+1} represents the word to the right. There are a number of juncture types and the task of a phrasing model is to assign the most appropriate type to each juncture. The experiments in this paper use two types of juncture, boundary and non-boundary.

A good feature set can help to improve the prediction accuracy, however finding a good feature set is work intensive. In fact, which parameters are relevant to prosody phrasing still remains an open research question. Part-of-speech (POS) is the most commonly used feature because POS information is relatively independent of language and easy to obtain. Hirose (1986) found that prosodic boundaries tend to bisect longer units, and Fach's study (1999) showed that there is a strong correspondence between syntactic phrasing and prosodic phrasing. So we selected the following 17 features which seem to affect Chinese prosody:

- *Pos*{1-4}: A part-of-speech window of four around the juncture, $<w_{i-1}$, w_i, w_{i+1}, $w_{i+2}>$.
- *Sbs*: The smallest syntactic constituent dominating both w_i and w_{i+1};
- *Sll*: The largest constituent dominating w_i, but not w_{i+1};
- *Srl*: The largest constituent dominating w_{i+1}, but not w_i;
- *Ltw, Ltc*: Total words and characters in the sentence;
- *Lsw, Lsc*: Distance from start to w_i, in words and characters;
- *Lew, Lec*: Distance from w_{i+1} to end, in words and characters;
- *Lllw, Lllc*: The size of *Sll*;
- *Lrlw, Lrlc*: The size of *Srl*.

4.3 Results

We divided the corpus into training data (371 sentences, 8170 word junctures) and test data (188 sentences, 4084 junctures). We did not consider the boundaries between sentences because virtually there is always a boundary at the end of a sentence. When the training procedure completed, we got a set of well understood rules. And then these rules were applied to the test data. The accuracy is defined as:

$$Accuracy = \frac{N_{corr}}{N_{total}} \quad (5)$$

where N_{corr} is the number of data which had been correctly predicted, and N_{total} is the total amount of the test data. In order to provide a baseline, accuracy using the initial rule that assigns "non-boundary" to each word juncture is included. And four other algorithms are also used in our experiments for comparison. The experimental results are summarized in Table 3.

Table 3. Performance of Different Methods.

Algorithm	Accuracy
All no boundary	77.6%
CART	91.4%
TRBL	89.5%
C4.5	92.6%
CN2	90.1%
HGR	92.9%

Table 3 shows that HGR algorithm achieves better performance than commonly used CART techniques and is competitive with C4.5. The confusion matrix for the HGR algorithm experiment is given in Table 4.

Table 4. Confusion Matrix for HGR Algorithm.

Truth	Predicted		
	No boundary	Boun-dary	Percent correct
No boundary	3074	80	97.5%
Boundary	210	720	77.4%

5 Conclusion

In this paper, we have described a new prosodic phrasing model for Chinese TTS systems. The problem of prosodic phrasing can be seen as a machine learning problem, so generally speaking, any machine learning algorithm can be used to solve this problem, and the rule induction algorithms seem to be more suitable than CART because of their higher accuracy and more straightforward interpretation of the prediction rule. Furthermore, the problem of efficiency is an important issue for a real time TTS, and it is very convenient for our algorithm to train the prosodic phrasing rules off-line and then incorporate them into the system.

For ongoing research to further improve the performance, we are experimenting in three directions: (1) Increase the size of the training data. (2) Incorporate the rules generated by our algorithm into a real TTS system where we have to depend upon an imperfect segmenter and parser to provide us with the features we need, and (3) Investigate the effectiveness of other learning techniques.

References

Bachenko, J. and Fitzpatrick, E. 1990. A Computational Grammar of Discourse-Neutral Prosodic Phrasing in English. *Computational Linguistics* 16: 155-170.

Fach, M. L. 1999. A Comparison Between Syntactic and Prosodic Phrasing. *Proceedings of Eurospeech99*: 527-530.

Fordyce, C. S. and Ostendorf, M. 1998. Prosody Prediction for Speech Synthesis Using Transformational Rule-Based Learning. *Proceedings of the International Conference on Spoken Language Processing (ICSLP-98)*: 682-685.

Hirose, K.; Fujisaki, H. and Kawai, H. 1986. Generatation of Prosodic Symbols for Rule-Synthesis of Connected Speech of Japanese. *Proceedings of the IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP-86)*: 2415-2418.

Hirschberg, J. and Prieto, P. 1996. Training Intonational Phrasing Rules Automatically for English and Spanish Text-to-Speech. *Speech Communication* 18: 281-290.

Hong, J. R. 1985. AE1: An Extension Matrix Approximate Method for the General Covering Problem. *International Journal of Computer and Information Science* 14: 421-437.

Lee, S. and Oh, Y. H. 1999. Tree-Based Modeling of Prosodic Phrasing and Segmental Duration for Korean TTS Systems. *Speech Communication* 28: 283-300.

Muller, A. F.; Zimmermann, H. G. and Neuneier, R. 1996. Robust Generation of Symbolic Prosody by a Neural Classifier Based on Autoassociators. *Proceedings of the IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP-96)*: 1285-1288.

Murphy, P. M. and Aha, A. W. 1995. UCI Repository of Machine Learning Databases, Machine-Readable Data Repository, Department of Computer Science, University of California, Irvine, CA.

Ostendorf, M. and Wightman, C. W. 1993. Parse Scoring with Prosodic Information: an Analysis/Synthesis Approach. *Computer Speech and Language* 7: 193-210.

Taylor, P. and Black, A. W. 1998. Assigning Phrase Breaks from Part-of-Speech Sequences. *Computer Speech and Language* 12: 99-117.

Wang, M. Q. and Hirschberg, J. 1992. Automatic Classification of Intonational Phrase Boundaries. *Computer Speech and Language* 6: 175-196.

Willemse, R. and Boves, L. 1991. Context Free Wild Card Parsing in a Text-to-Speech System. *Proceedings of the IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP-91)*: 757-760.

Wu X. D. 1998. Rule Induction with Extension Matrices. *Journal of the American Society for Information Science* 49: 435-454.