AUTOMATIC GENERATION OF PROSODIC RULES FOR SPEECH SYNTHESIS

Yoichi Yamashita *Riichiro Mizoguchi*
I.S.I.R., Osaka University
8-1, Mihogaoka, Ibaraki-shi, Osaka, 567 JAPAN
yama@ei.sanken.osaka-u.ac.jp

ABSTRACT

This paper describes automatic generation of speech synthesis rules which predict the accent component value (stress level) for the *bunsetsu* in long noun phrases. The rules are inductively inferred from a lot of speech data by using two kinds of tree-based methods, the conventional tree generation and the SBR-tree algorithm. The rule sets automatically generated by two methods have the almost same performance and decrease the prediction error to about 14Hz from 23Hz of the accent component value. The rate of the correct reproduction of the change, that is increase or decrease, for adjacent *bunsetsu* pairs is also used as a measure of evaluation and the generated rule sets correctly reproduce about 80% of the change. Effectiveness of the rule sets is verified through the listening test. And, the SBR-tree methods generate very compact rules which are easy for human experts to interpret and match with the former studies.

1. INTRODUCTION

One of the bottlenecks for realizing speech synthesis of high quality is the construction of a rule set which calculates acoustic or prosodic parameters. In the conventional way of developing rules for speech synthesis, human experts analyze speech data and make rules by hand based on the characteristics of parameters. It is difficult for them to construct a good set of rules taking account of many factors which have an effect on the parameters. Thus, several methods of automatic generation of speech synthesis rules have been tried based on techniques such as inductive learning from a lot of data or estimation of stochastic models[?][?][?][?].

Calculation of parameters in speech synthesis is one of typical regression problems. Since the quantification method usually resolves regression problems in the linear model, it is deficient in capturing non-linearity among factors, which are called items in terms of this model. The neural network has the powerful classification or prediction ability, but the result obtained by this method is like a black box and very difficult to interpret. On the other hand, tree-based methods have the following advantages:

1. They have more descriptive ability than the linear model has.
2. It is easy to introduce some heuristics into the process of learning.
3. It is easy to interpret the results which are represented by trees or rules.

For these reasons, we employ two kinds of tree-based methods, one is the conventional tree generation and the other is the SBR-tree algorithm which has been proposed as an extended algorithm of decision tree[?], for inductively generating prosodic rules from a lot of speech data. In this paper, the parameter to be calculated by rules is the accent component value of the *bunsetsu* in an isolated Japanese long noun phrase, which is a sequence of several *bunsetsus* such as nouns, adjectives, noun-following forms of the inflective words and so on. Here, a *bunsetsu* is a Japanese basic syntactic unit composed of a content word and several function words. Prediction of the f0 contour for the long noun phrase is one of important problems in speech synthesis, because the f0 contour with a constant accent level for *bunsetsus* sounds very monotonous and unnatural.

In general, the prosody generation for phrases or sentences in speech synthesis needs the meaning and the context as well as the syntactic information. However, since it is not easy to determine such high level informations such as the meaning and the context, we namely adopt the syntactic informations of the *bunsetsu* such as part of speech, modification relationship, and so on, as features which are necessary for the prediction of the accent component of the *bunsetsu* in the long noun phrase. Introducing the existence of emphasis for the *bunsetsu* as a feature is also discussed.

2. F0 MODEL AND SPEECH DATA

2.1. F0 Model

The f0 model used in this paper was proposed for Japanese by Hakoda and Satoh[?]. This model represents the f0 contour by accumulating the accent component on the phrase component, shown in Figure ??, and a stress level is represented with a value of the accent component. The frequency is expressed on the logarithmic scale in this paper.

2.2. Speech Data

As speech data for automatic generation and evaluation of rules, 200 long noun phrases from articles in newspapers were uttered once by a professional male speaker.

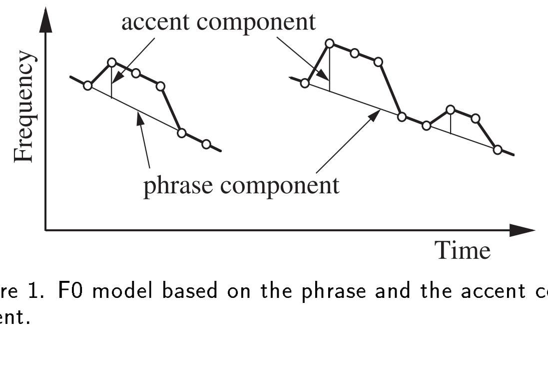

Figure 1. F0 model based on the phrase and the accent component.

 © IEEE 1994

1. *juuraino sansyainkeikakuno enerugiikaihatsu*
   ('energy development in the former Sun-Shine project')
2. *nihonnougyouno ninaiteto narubeki seisanseino takai nouka*
   ('farmers who are torchbearers of Japanese agriculture and have the high productivity')
3. *anzende hinshitsuno yoi nousakumotsuno kyoukyuu*
   ('supply of crops which are safe and of high quality')
4. ...

Figure 2. Examples of the long noun phrase.

The length of the long noun phrases is 4.6 *bunsetsus* and 20.1 morae in average and the average speaking rate is 6.8 mora/sec. Examples of the long noun phrase are shown in Figure ??. The phrase components and the accent components of the *bunsetsu* were extracted by visual inspection from the f0 contour in hardcopies of displayed screen.

2.3. Description of Cases

In this paper, speech synthesis rules which predict the accent component value for the *bunsetsu* in the long noun phrase are automatically generated. Consequently, a case for the rule generation corresponds to a *bunsetsu*. Each case is composed of 11 features and an accent component value which is obtained by the analysis in ??. Total number of the *bunsetsu* from 200 long noun phrases is 926. However, 895 cases are used because 31 cases are removed due to difficulties of the f0 analysis such as devocalization.

Table ?? shows 11 features describing the case. Here, B1 denotes the *bunsetsu* corresponding to the case. B0 and B2 denote the *bunsetsu* preceding B1 and following B1, respectively. All features are binary.

Table 1. List of features describing a case. (B1 denotes the *bunsetsu* corresponding to the case. B0 and B2 denote the *bunsetsu* preceding B1 and following B1, respectively.)

- x1: yes if B1 modifies B2 (MOD2)
- x2: yes if B1 is the last *bunsetsu* in the prosodic phrase, no otherwise (LAST)
- x3: yes if B1 contains a noun and a particle '*no*' (of), no otherwise (NNO)
- x4: yes if B0 contains a noun and a particle '*no*' and B1 contains a noun, no otherwise (PRENNO)
- x5: yes if B1 contains a compound noun, no otherwise (COMP)
- x6: yes if the accent type of B0 is flat and those of B1 is not type 1, no otherwise (HLOW)
- x7: yes if B1 contains a superficial verb, such as '*aru*' (exist), '*yoru*' (base), and so on, no otherwise (AUXV)
- x8: yes if B1 contains a superficial noun, such as '*koto*' (thing), '*tokoro*' (place), and so on, no otherwise (AUXN)
- x9: yes if B0, B1, and B2 contain a verb, a noun and a particle '*no*', and a noun, respectively, no otherwise (VNN)
- x10: yes if B0 and B1 contain a particle '*ni*' (to/at) and a verb, respectively, no otherwise (NIV)
- x11: yes if B0 and B1 contain a particle '*wo*' (to/for) and a verb, respectively, no otherwise (WOV)

3. AUTOMATIC LEARNING USING TREE-BASED APPROACHES

The problem we solve in this paper is to predict the accent component value of the *bunsetsu* using features of the *bunsetsu*. This is one of typical regression problems, in which a numerical value results from several features of input data. A tree is usually employed for classification problems by labeling a leaf as the name of a class, [?], however it is easily applicable to regression problems by changing the value of the leaf into a numeric[?]. Furthermore, an evolved method of the decision tree, called SBR-tree (Single Best Rule tree), has been proposed[?] and is also applicable to regression problems.

3.1. Conventional Tree

Algorithm of conventional tree generation takes the following steps:

- Step-1: Assign all training cases to a node, which is the root node.
- Step-2: Select a feature describing the case in order to split the cases into some categories and decompose the node into descendant nodes.
- Step-3: Repeat the step-2 recursively at new nodes until all the nodes satisfy the terminating condition.

The node-splitting rule used in the step-2 is very important, and a node is split with a feature which minimizes the total prediction errors at its descendant nodes. Another important rule is necessary for the step-3 to determine whether a node should be split or not. A node i becomes a leaf if

- the prediction error at the node $i <$ the threshold δ_t,
- the features of the case are exhausted, or
- the depth of the node $i >$ the threshold δ_d.

3.2. SBR-tree

A tree is easily converted into a set of rules. In a tree, each path from the root to a leaf represents a conjunctive rule, and the set of all paths is the disjunction of the rules. The rules formed as a tree are mutually exclusive. This property of the tree rules may increase the complexity of the rule set. The SBR-tree algorithm has been proposed to reduce such complexity and generate simpler rules. We can expect that rules generated by SBR-tree are easy to interpret. However, they should be ordered.

Algorithm of SBR-tree takes the following steps:

- Step-1: Generate a tree using all training cases.
- Step-2: Select a single best rule, which is a best path from the root to a leaf on the tree, based on an appropriate measure.
- Step-3: Remove redundant tests from this single best rule, if any. The condition part of the rule is represented with the conjunction of tests. If a test in the rule is hidden and the rule does not increase the prediction error for the cases covered by the rule, the test is actually removed from the condition part of the rule. Every test in the rule is checked in order.
- Step-4: Remove cases covered by the selected single best rule from the set of training cases and generate a tree using the new set of the training cases.
- Step-5: Repeat the step-2, -3 and -4 until the set of rules covers all cases.

The SBR-tree algorithm needs one more strategy which is definition of the best rule in the step-2. Three criteria are tried in this paper as follows.

- the shortest rule (SBRtree1)
- the rule covering the most cases (SBRtree2)
- the rule which covers the most cases per the depth of the leaf (SBRtree3)

4. EVALUATION

4.1. Performance of Rules

The N-fold cross validation (CV) was utilized for evaluation of rules in order to make use of as many training cases as possible. That is, the case set L is divided into N subsets L_k ($k = 1, 2, \cdots, N$). Rules are inductively generated from the training cases $L - L_1$ and are evaluated for the testing cases L_1. Next this procedure is repeated with other test sets, L_2 to L_N. The total performance of the CV method is the average of N results. The number of the subsets, N, is 20 in this evaluation.

Table ?? shows the best performance of rules generated by the conventional tree and the SBR-tree. Three kinds of SBR-tree were tried with the different measures of the best rule, mentioned in ??. The rules are evaluated on two criteria; one is the prediction error of the accent component and the other is the rate of the correct reproduction of change, that is increase or decrease, of the accent component for adjacent *bunsetsu* pairs. The latter gives evaluation from a viewpoint of relative change of f0.

The definition of the correct reproduction is that the change of the accent component predicted by rules is the same as original one of speech data for the adjacent *bunsetsu* pair. In a preliminary experiment, we investigated the deviation of the accent component for the same utterance using 30 long noun phrases uttered five times by one speaker. In result the deviation was about 6.5Hz. In general, speech utterances contain some fluctuations and one never produces exactly the same speech wave again. It is important to evaluate rules considering these fluctuations. Therefore, the reproduction rate is calculated without adjacent *bunsetsu* pairs which have less difference than the deviation of the accent component.

In this table, CONST means that the accent component value is predicted as a constant value, which is the average over all cases. This table says that automatically generated rules are very effective for predicting the accent component value and that there are little differences of performance among 4 methods.

4.2. Size of Rules

Table ?? compares the size of rule sets generated by 4 methods. The second column is the number of leaves or rules, for the tree rules or the SBR-tree rules, respectively. The last column is the average depth of the leaves or the average number of tests in each rule similarly. The SBR-tree methods generate more compact rule sets than the conventional tree method does.

4.3. An Example of Generated Rule

An entire rule set generated by SBRtree1 is shown in Figure ??. In this figure, the condition part of the rule is enclosed by brackets and the number next to the arrow is the action part of the rule, that is the predicted value of the accent component on the logarithm scale. Enclosed by parentheses is the number of cases which the rule matches with. As the SBR-tree method generates ordered rules, the rule which first matches with each *bunsetsu* is executed to predict the accent component value of the *bunsetsu*. Observing these rules, we can interpret them as follows.

Table 2. Performance of rule sets.

method	error[Hz]	reproduction rate[%]
Conventional tree	14.2	84.1
SBRtree1	15.1	80.6
SBRtree2	14.4	84.1
SBRtree3	14.4	84.5
CONST	23.2	-

Table 3. Size of rule sets.

method	size	
	leaves/rules	depth/conditions
Conventional tree	49.3	6.97
SBRtree1	13.1	1.35
SBRtree2	20.4	3.30
SBRtree3	21.2	2.83

```
rule1:                  [ COMP=yes ] -> 0.810  (250)
rule2:   [ LAST=yes, AUXV=yes ] -> 0.319  (19)
rule3:                  [ LAST=yes ] -> 0.527  (169)
rule4:                    [ VNN=yes] -> 0.957  (10)
rule5:                  [ AUXN=yes ] -> 0.496  (5)
rule6:    [ MOD2=yes, AUXV=no ] -> 0.771  (317)
rule7:                   [ WOV=yes ] -> 0.397  (3)
rule8:      [ AUXV=no, NIV=yes ] -> 0.420  (2)
rule9:                   [ NNO=yes ] -> 0.746  (24)
rule10:                  [ AUXV=no ] -> 0.633  (19)
rule11:                  [ NIV=yes ] -> 0.350  (4)
rule12:   [ MOD2=yes, HLOW=no ] -> 0.505  (10)
rule13:                 [ MOD2=yes ] -> 0.400  (9)
rule14:                          [ ] -> 0.320  (6)
```

Figure 3. An entire rule set generated by the SBRtree method.

- The compound noun has a very large accent component value (rule1).
- Superficial verbs and nouns have very small accent component values (rule2,5). Since they are tightly related to the preceding *bunsetsu*, the accent component is suppressed.
- The *bunsetsu* directly modifying the next *bunsetsu* has a slightly large accent component value (rule6). This rule matches with the former study conducted by Hakoda and Satoh[?]. They explained that the tightly connecting boundary of the *bunsetsu*, which is defined as the direct modification, suppresses the accent component of the following *bunsetsu*.
- Verbs which are tightly connected to the preceding *bunsetsu* with the particle '*wo*' (to/for) or '*ni*' (to/at) have very small accent component values (rule7,8,11) and nouns followed by the particle '*no*' (of) have large accent component values (rule9). Hakoda also pointed out that the type of the modification relationship is important to the f0 control and the *bunsetsu* connection with the particles '*ni*', '*wo*', and '*no*' are categorized into the tight connection[?].
- Nouns preceded by a verb and followed by a noun have very large accent component values (rule4).

4.4. Listening Test

In order to verify the effectiveness of automatically generated rule sets, the listening test was carried out. 10 long noun phrases, which were taken from 200 phrases, were synthesized with the acoustic parameters obtained by the analysis except for the accent component. The accent component value is constant (CONST), the same as the original analysis result (ORIG), or predicted by the automatically generated rules of SBRtree1 (RULE1) or SBRtree2

Table 4. Result of stimulus comparisons.

		less natural			
		CONST	RULE1	RULE2	ORIG
more natural	CONST	-	47	47	34
	RULE1	63	-	59	52
	RULE2	63	51	-	63
	ORIG	76	58	47	-

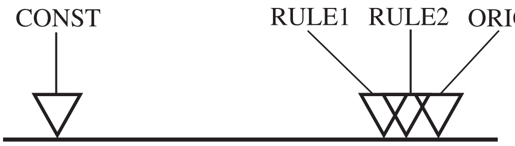

Figure 4. Auditory distance of four kinds of stimuli.

(RULE2). Thus, four kinds of stimuli for each phrase were presented to 11 subjects. The subject judged which has more natural intonation for all combination pairs of four kinds of stimuli. Table ?? shows the result of paired comparisons and four kinds of stimuli are arranged on the line by the Thurstone case V method as shown in Figure ??. This result shows that speech synthesized by using rules is near to the original one in the auditory sense.

4.5. Emphasis of *Bunsetsu* in Utterances

In the experiments mentioned above, we used only syntactic information as features describing cases of the *bunsetsu*. Although we did not give the speaker any instruction related to emphasis, he actually emphasized some *bunsetsus* in the long noun phrases. We investigated the emphasized *bunsetsus* by listening all long noun phrases. The 133 *bunsetsus* were uttered with full emphasis, and the 145 *bunsetsus* with half emphasis.

In order to predict the accent component value using semantic information, we added one feature into those shown in Table ?? and retried the rule generation by the SBR1 method. The new feature is related to emphasis of the *bunsetsu* and takes one of three values; full, half, and no emphasis. Introduction of this semantic feature lead to the drastic improvement. The prediction error is reduced to 12.2Hz from 15.1Hz and the correct reproduction rate is increased to 92.3% from 80.6%.

Since the emphasized *bunsetsus* have usually the large accent component, it is rather natural that the new feature of the emphasis reduces the prediction error. This result made it clear that the semantic information about the output message to be converted into the speech is essential to the speech synthesis of high quality. We believe that it is important to consider the representation scheme of the output message and the dialog processing in the speech interface system in order to add semantic features to the message, especially for spoken dialog synthesis[?].

5. CONCLUSIONS

This paper described methods of the automatic generation of speech synthesis rules which predict the accent component value. The construction of a rule base is one of the most important and difficult problems in speech synthesis by rule. Automatic generation of speech synthesis rules based on a lot of speech data is a promising paradigm which overcomes this bottleneck of speech synthesis. The methods proposed are based on two kinds of tree-based inductive learning algorithms, the conventional tree generation and SBR-tree.

The generated rule sets were evaluated by using the prediction error for the *bunsetsu* and the rate of the correct reproduction of the change for adjacent *bunsetsu* pairs. Effectiveness of the rule set were verified through the listening test. Two methods do not differ very much to each other with regard to the performance, but SBR-tree generates more compact rule sets than the conventional tree does. Particularly, the SBR-tree method with the definition of the best rule as the shortest path drastically reduces the complexity of the rule set. The rule set by the SBR-tree method is quite easy for human experts to interpret and matches with the former study. In a supplementary experiment, introduction of a feature about emphasis improved the performance of the rules.

Although speech data used in this paper is restricted to the long noun phrase, there is no limitation of the range which the tree-based method is applicable to. These methods can easily apply to the rule generation for sentences.

REFERENCES

[1] Yamashita, Y., Taniguchi, K., Mizoguchi, R., Yanagida, M. and Kakusho, O., "A Support System for Constructing Rules Base for Speech Synthesis by Rule –Automatic Extraction of Synthesis Rules–", *Trans. of IEICE*, J71-D, 6, pp.1012–1019, 1988 (in Japanese). (also appeared in *Systems and Computers in Japan*, 21, 4, pp.15–25, 1990)

[2] Sagisaka, Y., "On the Prediction of Global F0 Shape for Japanese Text-to-Speech", *Proc. of ICASSP '90*, S6a.9, pp.325–328, 1990.

[3] Riley, R.D., "Tree-Based Modelling of Segmental Durations", *Talking Machines: Theories, Models, and Designs*, North–Holland, pp.265–273, 1992.

[4] Kaiki, N., Mimura, K. and Sagisaka, Y., "Statistical Modeling of Segmental Duration and Power Control for Japanese", *Proc. of Eurospeech '91*, pp.625–628, 1991.

[5] Hakoda, K. and Satoh, H., "Prosodic Rules in Connected Speech Synthesis", *Trans. of IECEJ*, J63-D, 9, pp.715–722, 1980 (in Japanese).

[6] Hakoda, K., Nakajima, S. and Hirokawa, T., "A Rule for Deciding Tone Control Symbols in Text-To-Speech Conversion", *IEICE Tech. Rep.*, SP89-5, pp.33–38, 1989 (in Japanese).

[7] Quinlan, J.R., "Induction of Decision Trees", *Machine Learning*, 1, pp.81–106 1986.

[8] Breiman, L., Friedman, J.H., Olshen, R.A. and Stone, C.J., "Classification and Regression Trees", Wadsworth & Brooks, 1984.

[9] Indurkhya, N. and Weiss, S.M., "Iterative Rule Induction Methods", *Applied Intelligence*, 1, pp.43–54, 1991.

[10] Yamashita, Y., Mizutani, N., Kakusho, O. and Mizoguchi, R., "Speech Synthesis from Concept Representation in General Speech Output Interface", *Trans. of IEICE*, J76-DII, 3, pp.415–426, 1993 (in Japanese). (also to be appeared in *Systems and Computers in Japan*)