NEURAL NETWORK BASED PRONUNCIATION MODELING WITH APPLICATIONS TO SPEECH RECOGNITION

Toshiaki Fukada[†] *Takayoshi Yoshimura*[†‡] *Yoshinori Sagisaka*[†]

[†]ATR Interpreting Telecommunications Research Laboratories, Kyoto, Japan

[‡]Department of Computer Science, Nagoya Institute of Technology, Nagoya, Japan

Email: fukada@itl.atr.co.jp yossie@ics.nitech.ac.jp sagisaka@itl.atr.co.jp

ABSTRACT

We propose a method for automatically generating a pronunciation dictionary based on a pronunciation neural network that can predict plausible pronunciations (*realized pronunciations*) from canonical pronunciations. This method can generate multiple forms of realized pronunciations using the pronunciation network. Experimental results on spontaneous speech show that the automatically-derived pronunciation dictionary gives consistently higher recognition rates than a conventional dictionary.

1. INTRODUCTION

The creation of an appropriate pronunciation dictionary is widely acknowledged to be an important component for a speech recognition system. One of the earliest successful attempts based on phonological rules was made at IBM [1]. Generating a sophisticated pronunciation dictionary is still considered to be quite effective for improving the system performance on large vocabulary continuous speech recognition (LVCSR) tasks [2]. However, constructing a pronunciation dictionary manually or by a rule-based system requires time and expertise. Consequently, research efforts have been directed at constructing a pronunciation dictionary automatically. In the early 1990s, the emergence of phonetically-transcribed (hand-labeled) medium-size databases (e.g., TIMIT and Resource Management) encouraged a lot of researchers to explore pronunciation modeling [3][4]. Although all of these approaches are able to automatically generate pronunciation rules, hand-labeled transcriptions by expert phoneticians are required. As a result, automatic phone transcriptions generated by a phoneme recognizer, which enable one to cope with a large amount of training data, have been used in pronunciation modeling [5][6]. Recently, LVCSR systems have started to treat spontaneous, conversational speech, such as the Switchboard corpus and consequently, pronunciation modeling has become an important topic because word pronunciations vary more here than in read speech [7][8].

In this paper, we propose a method for automatically generating a pronunciation dictionary on the basis of a spontaneous, conversational speech database. Our approach is based on a pronunciation neural network that can predict plausible pronunciations (realized pronunciations) from canonical pronunciations; most other approaches use decision trees for pronunciation modeling [3][6]∼[8].

We define canonical and realized pronunciations as follows.

- *Canonical pronunciations*: Standard phoneme sequences assumed to be pronounced in read speech. Pronunciation variations such as speaker variability, dialect, or coarticulation in conversational speech are not considered.
- *Realized pronunciations*: Actual phoneme sequences pronounced in speech. Various pronunciation variations due to speaker or conversational speech can be included.

2. AUTOMATIC GENERATION OF A PRONUNCIATION DICTIONARY

2.1. Pronunciation network

To predict realized pronunciations from canonical pronunciations, we employ a multilayer perceptron as shown in Figure 1. In this paper, a realized pronunciation $A(m)$ for a canonical pronunciation $L(m)$ is predicted from five phonemes (i.e., quintphone) of the canonical pronunciation $L(m-2), \ldots, L(m+2)$[1].

This raises two questions: (1) how do we train a pronunciation network ? ; and (2) how do we generate multiple realized pronunciations by using the trained pronunciation network ? These questions are answered in the following sections.

2.2. Training procedures

2.2.1. Training data preparation

To train a pronunciation network, we first have to prepare training data, that is, input (canonical pronunciation) and output (realized pronunciation) pairs. The training data can be prepared by transcribing the speech waveform using phoneme recognizer and mapping the recognition result to the canonical pronunciation as follows.

1. Conduct phoneme recognition using speech training data for dictionary generation.
2. Align the canonical pronunciation sequence to the recognition result using a dynamic programming algorithm.

For example, if the phoneme recognition result for the canonical pronunciation /a r a y u r u/, is /a w a u r i u/, the correspondence between the canonical pronunciation and the recognition result can be determined as follows:

a	r	a	y	u	r		u	(canonical pron.)
a	w	a		u	r	i	u	(recognition result),

where the second phoneme of the canonical pronunciation, /r/, is substituted by /w/, and /y/ is deleted and /i/ is inserted for the sixth phoneme of the canonical pronunciation, /r/. That is, $L(2) = \mathtt{r}$, $A(2) = \mathtt{w}$, $L(4) = \mathtt{y}$, $A(4) = \mathtt{x}$ (deletion), $L(6) = \mathtt{r}$, and $A(6) = \{\mathtt{r}, \mathtt{i}\}$ (/i/ is an insertion). The correctly recognized phonemes are also treated

[1] This network structure is similar to that employed in NETtalk [9], which can predict an English word pronunciation from its spelling. Note that the pronunciation network is designed to predict realized pronunciations, for the purpose of improving the performance in spontaneous speech recognition, while NETtalk is designed to predict canonical pronunciations for text-to-speech systems.

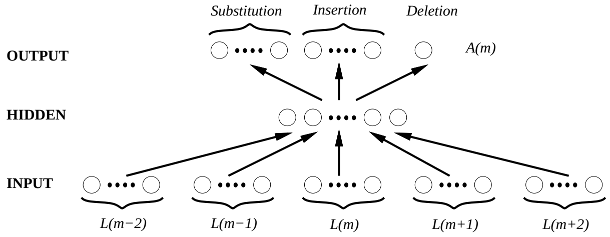

Figure 1. Pronunciation network.

as substitutions (e.g., /a/ is substituted by /a/). Phoneme recognition is conducted using all of the training data and the aligned results are used as the data for input and output, for the pronunciation neural network training (described in the following section). Note that both the phoneme recognition and alignment procedures are not performed for each word but for each utterance.

2.2.2. Structure of pronunciation network

To train a pronunciation network, a context of five phonemes in a canonical pronunciation, $L(m-2)$, ..., $L(m+2)$, is given as the input; $A(m)$ aligned to $L(m)$ is given for the output. A total of 130 units (26 Japanese phoneme sets times five contexts) are used in the input layer. The representation of the realized pronunciation at the output layer is localized, with one unit representing deletion, 26 units representing substitution, and 26 units representing insertion, providing a total of 53 output units[2].

In the previous example, when / /(deletion), which corresponds to the fourth canonical string /y/, is used as $A(m)$, and /r a y u r/ are used as $L(m-2), \ldots, L(m+2)$. Here, 1.0 is given as the output unit for deletion and as the input unit for /r/ in $L(m-2)$, /a/ in $L(m-1)$, /y/ in $L(m)$, /u/ in $L(m+1)$, and /r/ in $L(m+2)$; 0.0 is given for the other input and output units.

2.3. Generation procedures

2.3.1. Realized pronunciation generation

Assume that we want to find the most probable pronunciation for a word W in terms of pronunciation network outputs. Let the canonical pronunciation of W be denoted as $L = [L(1), \ldots, L(|W|)]$, where $|W|$ is the number of phonemes of the canonical pronunciation ($|W| \geq 5$). Realized pronunciation $A = [A(1), \ldots, A(|W|)]$ for L can be obtained in the following steps.

1. Set $i = 3$, $A(1) = L(1)$, and $A(2) = L(2)$.
2. For the quintphone context of the i-th phoneme, $l = [L(i-2), \ldots, L(i+2)]$, input 1.0 in the corresponding input units of the pronunciation network.
3. Find the maximum unit $U1_{out}$ in all of the output units.
   (a) If $U1_{out}$ is found in the substitution units, set $A(i)$ to the phoneme of $U1_{out}$.
   (b) If $U1_{out}$ is found in the insertion units, find another maximum unit $U2_{out}$ in the substitution units. Set $A(i)$ to the phoneme list of $U2_{out}$ and $U1_{out}$, respectively.
   (c) If $U1_{out}$ is the deletion unit, set $A(i) = \mathtt{x}$.
4. Set $i = i + 1$.
5. Repeat step 2 to step 4 until $i = |W| - 1$.
6. Set $A(|W|-1) = L(|W|-1)$ and $A(|W|) = L(|W|)$.

2.3.2. Multiple pronunciations with likelihoods

Multiple alternative pronunciations can be obtained by finding the N-best candidates based on the output values of the network. Multiple realized pronunciations can be determined by multiplying each normalized output for all possible combinations and choosing the probable candidates. We use a likelihood cut-off threshold for the multiplied normalized output [10].

2.3.3. Integrating the pronunciation likelihood into speech recognition

In conventional speech recognition systems, recognized word sequence $\hat{W}$ given observation O can be obtained by $\hat{W} = \operatorname{argmax}_W P(W|O)$. In this paper, we extend this formula by considering the realized pronunciation Prn for the word W as follows:

$$\hat{W} = \operatorname*{argmax}_{W \in \mathbf{W}} \sum_{Prn \in W} P(Prn, W|O). \tag{1}$$

Using Bayes' Rule, the right-hand side of Eq.(1) can be written as

$$\operatorname*{argmax}_{W \in \mathbf{W}} \sum_{Prn \in W} P(O|Prn, W)\ P(W)\ P(Prn|W). \tag{2}$$

The first term in Eq.(2), $P(O|Prn, W)$, is the probability of a sequence of acoustic observations, conditioned on the pronunciation and word string. This probability can be computed using an acoustic model. The second term in Eq.(2), $P(W)$, is the language model likelihood and can be computed using an n-gram word model. We call the third term in Eq.(2), $P(Prn|W)$, the *pronunciation model*. In this paper, the pronunciation network is used as the pronunciation model.

We consider that multiple realized pronunciations mainly represent the pronunciation variability caused by speaker or context differences. That is, for a certain speaker and in a certain context, only one realized pronunciation can be taken for a word pronunciation. Therefore, we omit the summation in Eq.(2). Furthermore, by applying exponential weighting to the language probability and pronunciation probability, the acoustic observation O can be decoded by the word sequence based on the following equation:

$$\operatorname*{argmax}_{W \in \mathbf{W}, Prn \in W} P(O|Prn, W)\ P(W)^{\alpha}\ P(Prn|W)^{\beta}, \tag{3}$$

where α and β are weighting factors for the language model and the pronunciation model, respectively.

2.3.4. Realized pronunciations for word boundary phonemes

Pronunciation variations for word-boundary phonemes can be taken into account based on language statistics [10]. As language statistics, we employ word bigram models here. Their probabilities are employed to generate realized pronunciations. Because word bigram models give all possible preceding and succeeding words and their frequencies for a certain word, five phoneme contexts (quintphone) of word boundary phonemes are statistically determined.

Consider that we want to find realized pronunciations for the first canonical phoneme $L_{W_C}(1)$ for a word W_C and its canonical pronunciation is $L_{W_C} = [L_{W_C}(1), \ldots, L_{W_C}(|W_C|)]$, where $|W_C|$ is the number of phonemes of the canonical pronunciation. Let a word which can be preceded by W_C be denoted as W_P whose canonical pronunciation is $L_{W_P} = [L_{W_P}(1), \ldots, L_{W_P}(|W_P|)]$, where

[2] In this paper, we do not treat insertions of more than two phonemes, because there are relatively very few of them and the number of weights can be reduced.

$|W_P|$ is the number of phonemes of the canonical pronunciation. Then, the quintphone for $L_{W_C}(1)$ is fixed as $[L_{W_P}(|W_P|-1),\ L_{W_P}(|W_P|),\ L_{W_C}(1),\ L_{W_C}(2),\ L_{W_C}(3)]$ and the output values of the pronunciation network can be computed. By computing output values for all possible preceding words for L_{W_C}, the output value of the i-th output unit, $\bar{S}_{W_C,i}(1)$, is statistically computed as

$$\bar{S}_{W_C,i}(1) = \sum_{W_P \in \mathbf{W}} P(W_C|W_P) S_{W_C,W_P,i}(1), \qquad (4)$$

where $\mathbf{W}$ is the set of all possible words, $P(W_C|W_P)$ is the conditional probability of W_C given by the word bigram models, and $S_{W_C,W_P,i}(1)$ is the output of the i-th output units computed by the quintphone input using W_C and W_P. Similarly, the output values for other word boundary phonemes, e.g., $L_{W_C}(2)$, $L_{W_C}(|W_C|-1)$, and $L_{W_C}(|W_C|)$, can be statistically computed. Once the outputs for each output unit are computed, multiple realized pronunciations for W_C can be obtained as described in **2.3.2.**.

3. PRONUNCIATION DICTIONARY FOR SPONTANEOUS SPEECH RECOGNITION

3.1. Conditions

A total of 230 speaker (100 male and 130 female) dialogues were used for the pronunciation network and acoustic model training. A 26-dimensional feature vector (12-dimensional mel-cepstrum + power and their derivatives) was computed using a 25.6 msec window duration and a 10 msec frame period. A set of 26 phonemes was used as the Japanese pronunciation representations.

Shared-state context dependent HMMs (CD-HMMs) with five Gaussian mixture components per state were trained. The total number of states was set to 800. By using the CD-HMMs and Japanese syllabic constraints, phoneme recognition was performed on the training data. The phoneme sequences of the recognition results were taken as the realized pronunciations. For each utterance, these realized pronunciations were aligned to their canonical pronunciations transcribed by human experts.

3.2. Pronunciation network training

Canonical pronunciations with quintphone contexts and their correspondent realized pronunciations (about 120,000 samples in total) were used as the inputs and outputs for the pronunciation network training. The structure of the pronunciation network is shown in Figure 1, where 130 input units, 100 hidden units, and 53 output units are used. There is also a bias that acts as an additional input constantly set to one. The total number of network weights including the biases becomes 18,453 (131×100+101×53). For output and hidden units, the sigmoid function with the mean squared error criterion is used because each output produces a number between 0 and 1 but the sum of all outputs does not sum up to one. The network was trained using 1,000 batch iterations and an intermediate network after 500 iterations was used in the following experiments. The differences in the recognition performance for the number of iterations are discussed in **5.1.**. The phoneme recognition accuracy between the canonical pronunciation and the training data was 81.1%. In order to indicate how the pronunciation network is able to predict pronunciation variations, we evaluated the performance of the pronunciation network by the coincidence rate and by the mean squared error (MSE) for the training data. Figure 2 shows the coincidence rates of target pronunciations and estimated pronunciations (solid line), and the MSE between the targets and the estimates (dotted line) as a function of the number of training iterations. The coincidence rate for the target and canonical pronunciation (shown as *Original Correct* in the figure) is 77.2 %.

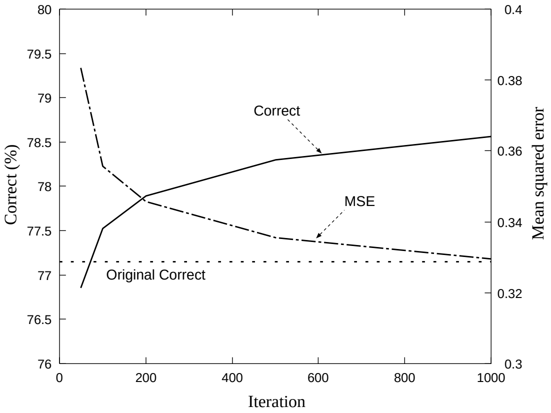

Figure 2. Coincidence rates (solid line) and mean squared error (dotted line) of targets and estimates for training data as a function of the number of training iterations. The coincidence rate for the target and canonical pronunciation (shown as *Original Correct*) is 77.2 %.

3.3. Generation of realized pronunciation dictionary

We applied the trained pronunciation network to a Japanese pronunciation dictionary with 7,484 word entries [3] developed for spontaneous speech recognition on a travel arrangement task. The dictionary was constructed by human experts considering pronunciation variabilities such as successive voicings [4], insertion and substitution of phonemes occurring in spontaneous speech, and possible insertions of a pause. Multiple pronunciaitons with 42,103 entries were obtained for the 7,484 word entries by setting a cut-off threshold, which controlled the number of realized pronunciations, to 0.4 for the multiplied normalized outputs. The multiple realized pronunciations obtained from the pronunciation network for a word /w a z u k a/ are shown in Table 1.

Table 1. Examples of realized pronunciations with normalized likelihoods for /w a z u k a/.

pronunciation	normalized likelihood
w a z u k a	1.0
a z u k a	0.896
w a z u t a	0.662
a z u t a	0.593

4. SPONTANEOUS SPEECH RECOGNITION EXPERIMENTS

To investigate the relative effectiveness of the proposed dictionary generated in **3.**, we conducted continuous speech recognition experiments on a Japanese spontaneous speech database.

4.1. Experimental conditions

The same training data, front-end, and acoustic model described in **3.1.** were used. For the open test set, 42 speaker

[3] Multi-words, which were automatically generated by the language modeling, were also included in the entries.

[4] Some Japanese word pronunciations change when a compound word is formed. For example, the conjunction of /k o d o m o/ (*child*) and /h e y a/ (*room*) is pronounced /k o d o m o $\underline{\text{b}}$ e y a/.

(17 male and 25 female) dialogues were used. Variable-order n-grams were used as the language model. A multi-pass beam search technique was used for decoding . The language and pronunciation probability weights, α and β in Eq.(3), were equally set.

4.2. Recognition results

Recognition results in the word error rate (WER) (%) for the simple dictionary are shown in Table 2. We can see from this table that the proposed dictionary achieved about a more than 9% error reduction compared to the baseline performance.

Table 2. Recognition results.

dictionary	WER (%)
Baseline	29.0
Proposed	26.4

5. DISCUSSION

5.1. Number of iterations for NN training

The WER and the total number of realized pronunciations as functions of neural network training iterations (50, 100, 200, 500, and 1,000) are shown in Fig. 3. The experimental conditions were the same as those described in **4.1.**, except that the threshold for the normalized likelihood was set to 0.5. The baseline expert dictionary was used for generating the realized pronunciations. No pronunciation likelihoods or language statistics were used in this experiment. From these results, it can be seen that the WER was reduced up to 500 iterations and then saturated, while the realized pronunciations kept increasing. Note that all created dictionaries outperformed the baseline dictionary.

5.2. Application to another recognition system

To see the effectiveness of the obtained pronunciation dictionary on other recognition systems, is an interesting topic, since we do not know whether the proposed method generates a universal dictionary able to be effective in other systems. To construct another system, we used Janus Recognition Toolkit (JRTk) [11]. Although the same training and test sets were used, the front-end, acoustic modeling, language modeling, and decoder were totally different from those in the previous experiment. Unfortunately, no significant improvement was observed (the WER slightly increased by 0.2%). We therefore suspect that a pronunciation dictionary generated based on phone recognition results, i.e., the proposed method or other similar approaches [5] ∼ [8], is difficult to use as a universal dictionary unless the inappropriate pronunciations caused by the phone recognizer (i.e., recognition errors) are filtered out.

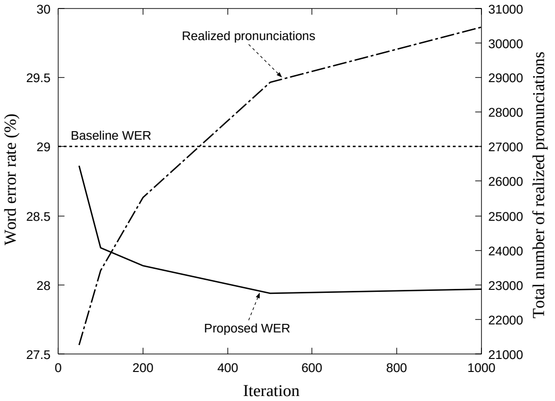

Figure 3. Word error rate and total number of realized pronunciations as functions of neural network training iterations.

6. CONCLUSION

In this paper, a method for automatically generating a pronunciation dictionary based on a pronunciation neural network has been proposed. Experimental results on spontaneous speech recognition show that the automatically-derived pronunciation dictionary gives higher recognition rates than the conventional dictionary. In this paper, only a quintphone context is used for predicting pronunciation variations, i.e., words whose quintphone contexts are the same have the same pronunciation variations. However, other factors (e.g., part-of-speech) can easily be incorporated into the pronunciation network by having additional units for these factors. Although the proposed method requires a fixed input window (i.e., a context of five phonemes), this requirement can be relaxed by adding word boundary phones (pad phones) to the beginning and ending of the word. In addition, we expect the multiple pronunciation dictionary to be a useful resource for acoustic model retraining by realigning the training data[7].

ACKNOWLEDGMENT

The experiment described in **5.2.** was performed at Carnegie Mellon University (CMU). We would like to thank Detlef Koll from CMU for his support in the experiment.

REFERENCES

[1] L. Bahl, J. Baker, P. Cohen, F. Jelinek, B. Lewis and R. Mercer: "Recognition of a continuously read natural corpus," *Proc. ICASSP-78*, pp. 422–424, 1978.

[2] L. Lamel and G. Adda: "On designing pronunciation lexicons for large vocabulary, continuous speech recognition," *Proc. ICSLP-96*, pp. 6–9, 1996.

[3] M. Riley: "A statistical model for generating pronunciation networks," *Proc. ICASSP-91*, pp. 737–740, 1991.

[4] C. Wooters and A. Stolcke: "Multiple-pronuncition lexical modeling in a speaker independent speech understanding system," *Proc. ICSLP-94*, pp. 1363–1366, 1994.

[5] P. Schmid, R. Cole and M. Fanty: "Automatically generated word pronunciations from phoneme classifier output," *Proc. ICASSP-93*, pp. II-223–II-226, 1993.

[6] J. Humphries: "Accent modelling and adaptation in automatic speech recognition," PhD thesis, University of Cambridge, 1997.

[7] E. Fosler, M. Weintraub, S. Wegmann, Y.-H. Kao, S. Khudanpur, C. Galles and M. Saraclar: "Automatic learning of word pronunciation from data," *Proc. ICSLP-96*, pp. 28–29 (addendum), 1996.

[8] B. Byrne, M. Finke, S. Khudanpur, J. McDonough, H. Nock, M. Riley, M. Saraclar, C. Wooters and G. Zavaliagkos: "Pronunciation modelling using a hand-labelled corpus for conversational speech recognition," *Proc. ICASSP-98*, pp. 313–316, 1998.

[9] T. Sejnowski and C. Rosenberg: "NETtalk: a parallel network that learns to read aloud," The Johns Hopkins Univ. Electrical Engineering and Computer Science Tech. Report JHU/EECS-86/01, 1986.

[10] T. Fukada, T. Yoshimura and Y. Sagisaka: "Automatic generation of multiple pronunciations based on neural networks and language statistics," *Proc. ESCA workshop on Modeling pronunciation variation for automatic speech recognition*, pp. 41–46, 1998.

[11] M. Finke, P. Geutner, H. Hild, T. Kemp, K. Ries and M. Westphal: "The Karlsruhe-Vermobil speech recognition engine," *Proc. ICASSP-97*, pp. 83–86, 1997.