New Grapheme Generation Rules for Two-Stage Model-based Grapheme-to-Phoneme Conversion

Seng Kheang[1], Kouichi Katsurada[1], Yurie Iribe[2] & Tsuneo Nitta[1,3]

[1]Toyohashi University of Technology, 1-1 Tempaku, Toyohashi, Aichi 441-8580, Japan
[2]Aichi Prefectural University, 1522-3 Ibaragabasama, Nagakute, Aichi 480-1198, Japan
[3]Waseda University, 1 Chome-104 Totsukamachi, Shinjuku, Tokyo 169-8050, Japan
Email: kheang@vox.cs.tut.ac.jp

Abstract. The precise conversion of arbitrary text into its corresponding phoneme sequence (grapheme-to-phoneme or G2P conversion) is implemented in speech synthesis and recognition, pronunciation learning software, spoken term detection and spoken document retrieval systems. Because the quality of this module plays an important role in the performance of such systems and many problems regarding G2P conversion have been reported, we propose a novel two-stage model-based approach, which is implemented using an existing weighted finite-state transducer-based G2P conversion framework, to improve the performance of the G2P conversion model. The first-stage model is built for automatic conversion of words to phonemes, while the second-stage model utilizes the input graphemes and output phonemes obtained from the first stage to determine the best final output phoneme sequence. Additionally, we designed new grapheme generation rules, which enable extra detail for the vowel and consonant graphemes appearing within a word. When compared with previous approaches, the evaluation results indicate that our approach using rules focusing on the vowel graphemes slightly improved the accuracy of the out-of-vocabulary dataset and consistently increased the accuracy of the in-vocabulary dataset.

Keywords: *grapheme generation rules (GGR); combined grapheme-phoneme information; two-stage model; grapheme-to-phoneme (G2P); automatic text-to-phonetic transcription.*

1 Introduction

Over the years, literate man has usually used the written word for directly accessing textual information stored in a computer system, but not speech documents, because access to speech documents requires knowledge relating to the process of word reading instead of the orthographic representation of the words. The use of the phonetic transcription of words has also been used in systems concerning the field of natural language processing, such as pronunciation learning software, automatic labeling software for speech recognition, and text-to-speech systems. Therefore, the quality of the conversion of arbitrary text into its corresponding phoneme sequence has a strong influence

Received October 20th, 2014, Revised December 11th, 2014, Accepted for publication December 19th, 2014.
Copyright © 2014 Published by ITB Journal Publisher, ISSN: 2337-5787, DOI: 10.5614/itbj.ict.res.appl.2014.8.2.6

on the performance of speech synthesis systems and on the search results of spoken term detection systems.

Theoretically, the phonetic transcription of a word can be generated using available pronunciation dictionaries for in-vocabulary (IV) words or predicted through a data-driven grapheme-to-phoneme (G2P) conversion for out-of-vocabulary (OOV) words. In order to solve the problems concerning G2P conversion, many approaches (briefly described in Section 2) have been proposed. Recently, the weighted finite-state transducer (WFST)-based G2P conversion method [1] achieved good word accuracy by utilizing a standard joint N-gram model and investigating N-best rescoring with a recurrent neural network language model. However, our two-stage architecture for G2P conversion [2], proposed in 2011, showed the advantage of using phonemic rather than graphemic information to predict the best final output phoneme sequence corresponding to the input word. In addition, our recent paper [3] demonstrated that this two-stage G2P conversion using neural networks is good at identifying single phonemes in a word, but lacks the knowledge for detecting the whole word. As a result, it provides higher phoneme accuracy but lower word accuracy than WFST-based G2P conversion.

Therefore, in this paper, we utilize the existing WFST-based approach to implement a novel two-stage architecture-based G2P conversion. This work investigates a new strategy in which we combine both graphemic and phonemic information as the input sequence for the G2P conversion. Moreover, several new grapheme generation rules for transforming each input word into different representations of grapheme sequences are also introduced in this paper, which enable the addition of extra detail to the vowel and consonant graphemes appearing in a word. In this study, a grapheme could be a single letter or a combination of letters. Most of these rules focusing on the vowel graphemes and can achieve a small but consistent improvement on previous approaches.

The remainder of this paper is organized as follows: in Section 2, we present methods proposed by other researchers to solve the problems concerning G2P conversion. We introduce several newly invented grapheme generation rules in Section 3. Then, we describe the novel two-stage model for G2P conversion in Section 4 and provide the evaluation results in Section 5. The discussion and conclusion are in Section 6 and 7, respectively.

2 Related Work

A set of context-dependent rewrite rules [4], proposed by Chomsky and Halle in 1968, was used in the development of a traditional rule-based approach for automatic translation of text into a string of phonemes. Because of the complexities of English pronunciation rules, many interesting data-driven approaches to automatic phonemic transcription have been proposed.

The complicated relations between letters and phonemes, especially in the case of a language with less regular spelling like English, do not allow the implementation of a one-to-one mapping technique between letters and phonemes for G2P conversion. Thus, a many-to-one mapping technique between letters and phonemes has been integrated in a number of previous approaches. For example, a very well-known data-driven approach using back-propagation neural networks implemented in the NETtalk system was proposed in 1987 [5]. The production of compact rule sets using the Default & Refine algorithm was proposed by Davel and Barnard in 2008 for designing an accurate and efficient pronunciation prediction mechanism for speech processing systems [6]. Furthermore, many other approaches, based on the hidden Markov model [7], inference of rewriting rules [8], pronunciation by analogy [9] and neural networks [10],[11], have also been proposed for dealing with the problems in G2P conversion. However, the integration of a many-to-one mapping technique in those approaches proved unsatisfactory because there is no strict correspondence between letters and phonemes [12].

As a consequence, various many-to-many mapping techniques between letters and phonemes have been proposed subsequently, in order to improve the accuracy of G2P conversion. For example, Rama et al. treated the letter-to-sound conversion problems as a phrase-based statistical machine translation problem [13]. The hidden Markov model (HMM)-based approach with context-sensitive observations for G2P conversion [14], proposed by Ogbureke et al., obtained a word accuracy of 79.79% on the Unilex corpora containing the UK English words, but only a maximum of 57.85% for the CMUDict corpus [15] due to the large number of loan words and a few remarkable errors. On the other hand, the two-stage neural network (NN)-based approach for G2P conversion [3] uses both grapheme and phoneme context information to predict the best final output phoneme sequence corresponding to the input word, which could take the performance of the single-stage neural network-based approach for G2P conversion to a higher level. This technique also inspired a letter-to-sound conversion technique using coupled hidden Markov models for lexicon compression [16]. The joint sequence model [17], proposed by Bisani and Ney, is one of the most popular approaches in G2P conversion. The recent WFST-based G2P conversion [1], implemented in the Phonetisaurus toolkit[1], achieved good word accuracy compared to other approaches.

[1] Phonetiaurus toolkit: https://code.google.com/p/phonetisaurus/

Table 1 List of the selected grapheme generation rules.

	Rule	Description			
			(Word	→	Grapheme Sequence)
	GGR1		g_i	→	g_i
			Ex: “OKEECHOBEE”	→	O K E E C H O B E E
	GGR2		g_i	→	$g_i\ g_{i+1}$
			Ex: “OKEECHOBEE”	→	OK KE EE EC CH HO OB BE EE E_
Rules focusing on vowel graphemes	GGR3		$v_1 \ldots v_n$	→	$v_1v_2\ \ v_2v_3 \ldots\ \ v_{n-1}v_n\ \ \ v_n$
			Ex: “OKEECHOBEE”	→	O K **EE** **E** C H O B **EE** **E**
	GGR4	If (n >1):	$v_1 \ldots v_n c_{n+1}$	→	$v_1v_2\ \ v_2v_3\ \ldots\ v_{n-1}v_n\ \ v_nc_{n+1}\ c_{n+1}$
			$v_1 \ldots v_n$ ↵	→	$v_1v_2\ \ v_2v_3\ \ldots\ v_{n-1}v_n\ \ v_n$
		If (n = 1):	g_i	→	g_i
			Ex: “OKEECHOBEE”	→	O K **EE** **EC** C H O B **EE** **E**
	GGR5	If (n > 1):	$v_1 \ldots v_n c_{n+1}$	→	$v_1v_2\ \ v_2v_3\ \ldots\ v_{n-1}v_n\ \ v_nc_{n+1}\ c_{n+1}$
			$v_1 \ldots v_n$ ↵	→	$v_1v_2\ \ v_2v_3\ \ldots\ v_{n-1}v_n\ \ v_n$ _
		If (n = 1):	g_i	→	g_i
			Ex: “OKEECHOBEE”	→	O K **EE** **EC** C H O B **EE** **E_**
	GGR6	If (n >1):	$[c_0]v_1 \ldots v_n c_{n+1}$	→	$[c_0v_1\ \ ]v_1v_2\ \ldots\ v_{n-1}v_n\ \ v_nc_{n+1}\ \ c_{n+1}$
			$[c_0]v_1 \ldots v_n$ ↵	→	$[c_0v_1\ \ ]v_1v_2\ \ldots\ v_{n-1}v_n\ \ v_n$ _
		If (n = 1):	g_i	→	g_i
			Ex: “OKEECHOBEE”	→	O **KE** **EE** **EC** C H O **BE** **EE** **E_**
Rules focusing on consonant	GGR7		$c_1 \ldots c_n$	→	$c_1c_2\ \ c_2c_3 \ldots\ \ c_{n-1}c_n\ \ \ c_n$
			Ex: “OKEECHOBEE”	→	A **PP** **PL** L I C A T I O N
	GGR8	If (n >1):	$c_1 \ldots c_n v_{n+1}$	→	$c_1c_2\ \ c_2c_3\ \ldots\ c_{n-1}c_n\ \ c_nv_{n+1}\ v_{n+1}$
			$c_1 \ldots c_n$ ↵	→	$c_1c_2\ \ c_2c_3 \ldots\ \ c_{n-1}c_n\ \ \ c_n$
		If (n = 1):	g_i	→	g_i
			Ex: “APPLICATION”	→	A **PP** **PL** **LI** I C A T I O N
	GGR9	If (n > 1):	$c_1 \ldots c_n v_{n+1}$	→	$c_1c_2\ \ c_2c_3\ \ldots\ c_{n-1}c_n\ \ c_nv_{n+1}\ v_{n+1}$
			$c_1 \ldots c_n$ ↵	→	$c_1c_2\ \ c_2c_3 \ldots\ \ c_{n-1}c_n\ \ \ c_n$ _
		If (n = 1):	g_i	→	g_i
			Ex: “APPLICATIONS”	→	A **PP** **PL** **LI** I C A T I O **NS** **S_**
	GGR10	If (n >1):	$[v_0]c_1 \ldots c_n v_{n+1}$	→	$[v_0c_1\ \ ]c_1c_2\ \ c_2c_3\ \ldots\ c_{n-1}c_n\ \ c_nv_{n+1}\ v_{n+1}$
			$[v_0]c_1 \ldots c_n$ ↵	→	$[v_0c_1\ \ ]c_1c_2\ \ldots\ c_{n-1}c_n\ \ c_n$ _
		If (n = 1):	g_i	→	g_i
			Ex: “APPLICATIONS”	→	**AP** **PP** **PL** **LI** I C A T I **ON** **NS** **S_**
	GGR11	GGR3 + GGR7			
			Ex: “APPLICATION”	→	A **PP** **PL** L I C A T **IO** O N

$g_i = \{c_i\ , v_i\}$; ‘↵ ’ = End of word; ‘_’ = Empty consonant grapheme; [ ] = Optional parameter

3 New Grapheme Generation Rule (GGR)

The G2P conversion model is usually built as a one-stage architecture for use in predicting phonemes corresponding to input text, especially with OOV words. To improve the model’s performance, this research integrated various newly invented grapheme generation rules into the model.

The grapheme side does not carry sufficient information or knowledge relating to the phonological interaction [18]. In order to make the graphemic information more sensitive in the G2P conversion, this work designed new rules with respect to the concept of context-dependent models, particularly for generating different grapheme sequences out of the same input word. Theoretically, for each grapheme of a given word, we concatenate it with the graphemes on its left and right contexts. However, in this paper, only the right context information is involved in the rule-making process because we prefer a compact representation for the new grapheme symbols, each of which consists of one or two alphabetical letters (e.g., “A” or “AU”).

Because the interaction between vowels in a word has a strong impact on the spelling process, most of the rules written in this paper were carefully designed to add extra sensitive information to each vowel grapheme appearing in a word. For a few connecting graphemes many rules are possible, but only the rules more related to the vowel graphemes (as listed in Table 1) are taken into account. However, in order to compare the impacts of the vowel and the consonant grapheme in the automatic conversion of a word into its phonetic transcription, we also propose some other rules that mainly focus on the consonant graphemes. As a result, Table 1 shows that most of the newly generated grapheme sequences can make the G2P conversion system easily identify not only the pattern of each vowel but also that of each consonant in a given word. In this table, the parameter g_i refers to the grapheme in index i, while c_i and v_i represent the consonant and vowel graphemes in index i, respectively. Moreover, the parameter n represents the number of vowels.

The first rule (GGR1) represents a unigram model used by most researchers [1]-[11,[13],[14],[16], but it appears not to provide sufficient information concerning each vowel or consonant grapheme. The second rule (GGR2) represents a bigram model, which seems to add too much information to each grapheme because it always combines the consonant grapheme with the vowel grapheme. The other four rules (GGR3, GGR4, GGR5 and GGR6) are designed specifically for adding the information missing in the first rule. For example, the third rule (GGR3) can distinguish the separated vowel – the vowel V that appears in the CVC pattern – from the vowels at the front part of the connecting vowels, i.e. the vowels V1, V2,…, Vn-1 of the V1…Vn pattern. In addition to GGR3, the other three remaining rules (GGR4, GGR5 and GGR6) are capable of distinguishing between the front vowels V1, V2,…, Vn-1 and the last vowel Vn of the V1…VnCn+1 pattern. The use of the empty grapheme “_” in GGR5 and GGR6 permits the recognition of the difference between the last vowel Vn of the C0V1…VnCn+1 pattern and that located at the end of word – the vowel Vn of the C0V1…Vn↵ pattern. Moreover, GGR6 adds more information to the consonant next to the connected vowels (e.g., the graphemes “KE” and “BE”).

In addition, the rules GGR7, GGR8, GGR8-1, GGR9, GGR9-1 and GGR10 are proposed for adding extra detail to the consonant graphemes appearing in the given word, which are designed with respect to GGR3, GGR4, GGR4-1, GGR5, GGR5-1 and GGR6, respectively. Furthermore, another rule that combines GGR3 with GGR7 (GGR11) was created to enable the addition of extra detail for both vowel and consonant graphemes appearing within a word.

4 Two-Stage Model for G2P Conversion

The architecture of the two-stage model-based approach was first proposed in 2011 to address the problem of phoneme conflicts in G2P conversion [2]. This architecture was basically implemented by connecting two different multilayer neural networks in sequence, which improves the accuracy of the ordinary one-stage neural network-based G2P conversion [10, 11]. However, the evaluation results in our recent paper [3] demonstrated that the two-stage model using the Fast Artificial Neural Network (FANN) Library[2] lacks some knowledge for detecting the whole word, so it provides lower word accuracy but higher phoneme accuracy than the WFST-based G2P conversion available in the Phonetisaurus toolkit. Therefore, we used the existing WFST-based approach to employ a novel two-stage model-based approach.

4.1 Prediction using Combined Grapheme-Phoneme (G-P) Information

The phoneme prediction method, in which only the phonemic information is used as input to select the best final output phoneme, was first presented in our previous papers [2],[3]. Its paradigm (Graphemes → Phonemes → Phonemes) shows that this method first converts the input word into phonemic information; then, all the related phonemic information is combined and used to predict the exact output phonemes of the G2P conversion model.

Because only the phonemic information is used in our previous method, we recognized that all of the words producing the same phoneme sequence (or pronunciation) during training in the first-stage are merged together before the second stage. For instance, the words “KOLL,” “KOLLE,” “CAUL,” and “KAHLE” all generated the same phoneme sequence /K AA L/ at the first-stage, so only one sample /K AA L/ → /K AA L/ was used at the second stage. Furthermore, some wrong phoneme sequences may be obtained by accident because it is virtually impossible to obtain a perfectly trained first-stage model. Therefore, some training data could be incorrectly merged or ignored by the

[2] FANN Library: http://leenissen.dk/fann/wp/

second-stage model. For example, the word “COALE” wrongly generates /K AA L/ as its output, while its correct phoneme sequence is /K OW L/. Therefore, it is ignored by the second-stage model. Such a problem reduces the number of training data at the second-stage and negatively affects the performance of the model.

In order to address this problem, we propose a new phoneme prediction method in which the input graphemes and output phonemes obtained from the first stage are combined and used as the new input sequence to determine the best final output phoneme sequence corresponding to the input word. Therefore, our newly proposed method consists of two steps:

1. First step : Graphemes → Phonemes
2. Second step : Combined G-P pairs → Phonemes

4.2 Architecture of the Proposed Model

On the basis of the new phoneme prediction method presented in the previous section, the novel two-stage G2P conversion architecture is built using two main modules (i.e. first-stage and second-stage models) in sequence.

4.2.1 First-Stage Model

The first-stage model, implemented based on the original WFST-based G2P conversion presented in [1] and available in the Phonetisaurus toolkit, is used for the automatic conversion of a word to its corresponding phoneme sequence. As can be seen in Figure 1, this model is trained with pairs of words and their phoneme sequences and each input word must first be generated into a grapheme sequence by using any grapheme generation rule from Table 1. In this context, each grapheme is represented by a single letter (e.g. “A”) or a combination of letters (e.g. “OA”), depending on the rule selected, and they are separated from one another by an empty space. Because it is virtually impossible to acquire a perfectly trained model, some unexpected errors will be produced at this stage.

For example, after training three words with almost the same pronunciation (e.g., “KOLL”→/K AA L/, “KOLLE”→/K AA L/, and “COALE”→/K OW L/), Figure 1 demonstrates that the word “COALE” generates “C OA A L E” as its grapheme sequence and then produces /K AA L/ as its output phoneme sequence with one error phoneme /AA/. Supposing that the other two words produce correct phoneme sequences, these three words all output the same phoneme sequence, /K AA L/.

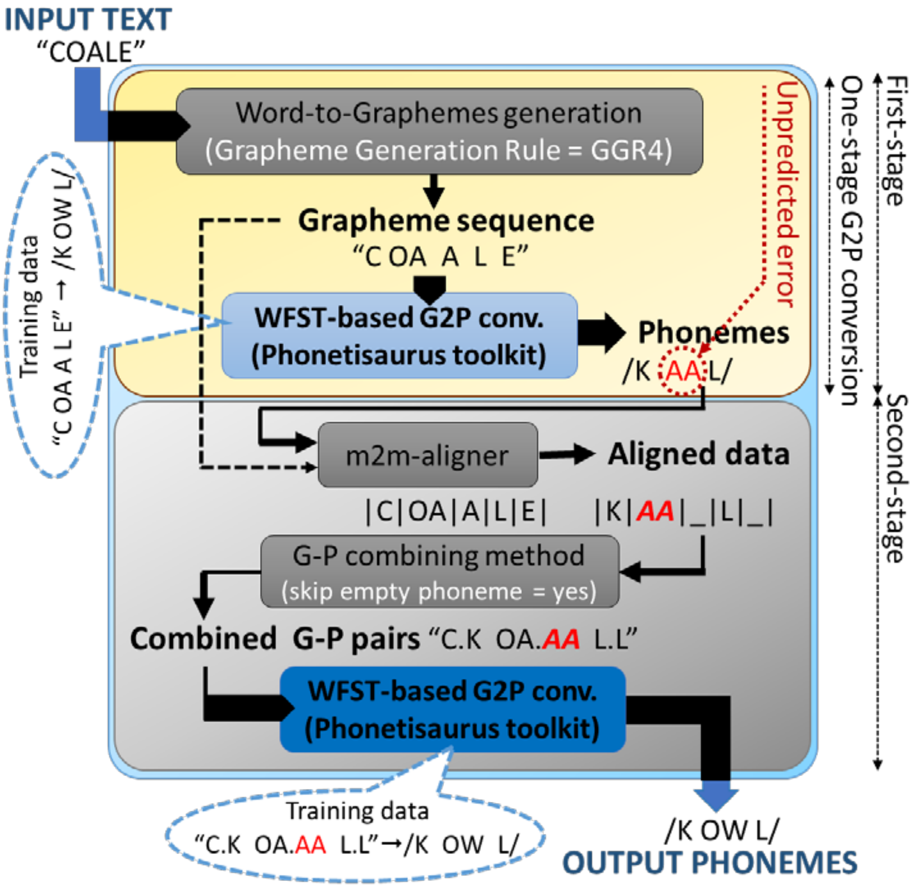


Figure 1 Architecture of the novel two-stage model-based G2P conversion.

4.2.2 Second-Stage Model

The second-stage model is built similarly to the first-stage model, with the exception that it combines both the input grapheme and the output phoneme sequences obtained from the first stage and utilizes that combined sequence as a new input to determine the best final output phoneme sequence corresponding to the original input word. In this paper, that new input sequence is called "a sequence of combined G-P pairs" hereafter. As both the graphemic information and the preliminary phonemic information have already been obtained before the final phoneme prediction, some errors occurring at the output level of the first-stage model can be fixed at the second stage. Therefore, our novel two-stage model for G2P conversion seems to provide a better performance.

According to Figure 1, this conversion requires two additional sub-modules for utilizing the grapheme and phoneme sequences of the first-stage model as input for the second-stage model. The first sub-module is created using the m2m-aligner software[3] for aligning the grapheme and phoneme sequences. The second sub-module automatically transforms the aligned data into a new sequence of combined G-P pairs to be used as input for the second stage; we also implemented a default option to ignore all the G-P pairs in which the grapheme is mapped to an empty phoneme (i.e., /_ /).

For the previous example, three aligned sequences such as "|K|O|L|L|→ |K|AA|L|_|," "|K|O|L|L|E|→ |K|AA|L|_|_|," and "|C|OA|A|L|E|→ |K|*AA*|L|_|_|" are generated after the alignment process. After passing all of them through the second sub-module, three sequences of combined G-P pairs are made, which include two unique sequences "K.K O.AA L.L" and another sequence "C.K OA.*AA* L.L". Hence, only two new training data (e.g., "K.K O.AA L.L"→ /K AA L/ and "C.K OA.*AA* L.L"→ /K OW L/) are created. Finally, the error phoneme /AA/ can be fixed at the second-stage.

5 Evaluation

In this section, we first describe the data preparation. Then, we present different proposed test sets including two baseline approaches and sixteen other approaches. The performance metrics are explained after that, which is followed by the experimental results of all the proposed test sets.

5.1 Data Preparation

The performance of our proposed approach was evaluated against two baseline approaches. We conducted experiments on the American English words-based pronunciation dictionary (CMUDict corpus [15]) used in our previous papers [2],[3], except that each word and its phoneme sequence used in this paper were unaligned (i.e. absence of the empty grapheme '_' and empty phoneme /_ /). Thus, the training dataset contained a total of 100,713 IV words, while the testing dataset contained 11,188 OOV words. Although we used the same CMUDict corpus as [1],[17], the selected words in our datasets were different from those used in [1],[17]. The dataset preparation is fully described in our previous paper [2].

After the data analysis, the grapheme "X" is sometimes mapped to three phonemes /EH K S/ (e.g., "VISX"→/V IH S *EH K S*/). To this end, we replaced

[3] m2m-aligner software: https://code.google.com/p/m2m-aligner/

the connected phonemes /K S/ and /K SH/ with two other phonemes /X/ and /XH/, respectively, for words where “X” produces /K S/ and /K SH/.

5.2 Proposed Test Sets

In this research, we designed and separately utilized eighteen different test sets, as listed in Table 2. According to [9], the WFST-based approach proved to outperform other well-established approaches such as Sequitur [17], direcTL+ [19], therefore we chose only the WFST-based approach to represent our baseline approach. As a result, we first propose two baseline approaches (i.e. Baseline1 & Baseline1-0) using GRR1, which refers to the original WFST-based approach [1].

Table 2 Configurations of the eighteen proposed test sets.

Proposed test set	**Configuration**		
	G-P mapping	**/K S/→/X/; /K SH/→/XH/**	**Grapheme Generation Rule**
Baseline 1	2-2	No	GGR1
Baseline 1-0	1-2	No	GGR1
Approach 1	2-2	Yes	GGR1
Approach 1-0	1-2	Yes	GGR1
Approach 2	1-2	No	GGR2
Approach 3	1-2	No	GGR3
Approach 4	1-2	No	GGR5
Approach 4-1	1-2	No	GRR4
Approach 5	1-2	Yes	GRR5
Approach 5-1	1-2	Yes	GRR4
Approach 6	1-2	No	GRR6
Approach 7	1-2	No	GGR7
Approach 8	1-2	No	GGR9
Approach 8-1	1-2	No	GGR8
Approach 9	1-2	Yes	GGR9
Approach 9-1	1-2	Yes	GGR8
Approach 10	1-2	No	GGR10
Approach 11	1-2	No	GGR11

Next, two similar approaches (Approach1 and Approach1-0) were designed with respect to both baseline approaches, with the exception that they were evaluated using the datasets where the connecting phonemes /K S/ and /K SH/ were manually replaced by /X/ and /XH/, respectively.

In order to show the effect of our proposed grapheme generation rules on the performance of the G2P conversion, especially on the word accuracy of the

OOV dataset, we designed the remaining approaches (as listed in Table 2) by assigning each of them different rules and configurations.

In the Phonetisaurus toolkit, the relationship between graphemes and phonemes can be many-to-many, but the best results were obtained when it was set to (1-2) or (2-2) [1]. Otherwise, whenever new grapheme generation rules (except for GRR1) were applied, our experimental results showed that the relationship (1-2) provided the best results. Therefore, in Table 2, we show only the approaches where the relationship (1-2) was used.

5.3 Performance Measuring Metrics

We evaluated the performance of the approaches listed in Table 2 in terms of phoneme accuracy (PAcc) and word accuracy (WAcc) using the NIST Sclite scoring toolkit[4]. The calculation of PAcc and WAcc are written as follows:

$$\mathrm{PAcc} = 1 - \mathrm{PER} = 1 - ((S_{ph} + D_{ph} + I_{ph}) / N_{ph}) \quad (1)$$

$$\mathrm{WAcc} = 1 - \mathrm{WER} = 1 - (S_{w} / N_{w}) \quad (2)$$

where PER and WER are known as phoneme error rate and word error rate, respectively; S_{ph}, D_{ph}, I_{ph} and N_{ph} are the number of phoneme substitutions, phoneme deletions, phoneme insertions, and total phonemes in the reference, respectively. Since only isolated words were used in our experiments, the value of WER was exactly equal to the number of word substitutions (S_w) divided by the total number of words in the reference (N_w). Because this research was more aimed at the WAcc result, we only report results related to this goal.

5.4 Experimental Results

The approaches listed in Table 2 used the CMUDict corpus to evaluate the model's performance. Since the selected words in both training and testing datasets were different from those used in [1],[17], the accuracy of the baseline approaches presented in this paper was lower than that shown in both previously mentioned papers. In terms of word accuracy (WAcc) of the OOV dataset, Figure 2 and Figure 3 indicate that most of the approaches using rules related to the vowel graphemes (i.e. Approach3, Approach4, Approach4-1, Approach5 and Approach5-1) provided better performance than those using rules related to the consonant graphemes (i.e. Approach2, Approach6, Approach7, Approach8, Approach8-1, Approach9, Approach9-1, Approach10 and Approach11); they also provided a slightly higher word accuracy than both baseline approaches at the first stage; however, there was no improvement

[4] NIST Sclite scoring toolkit: http://www.nist.gov/speech/tools/

between the one-stage and two-stage architecture. Conversely, in terms of the WAcc of the IV dataset, all approaches provided almost the same results (98.19% ~ 98.39%) when built as a one-stage model, but they improved when implemented as a two-stage model.

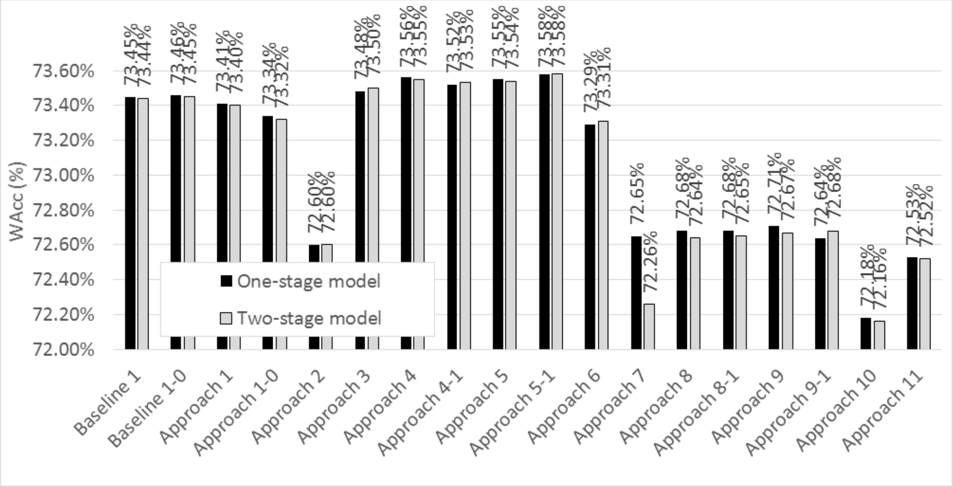


Figure 2 WAcc of different proposed test sets measured based on OOV dataset.

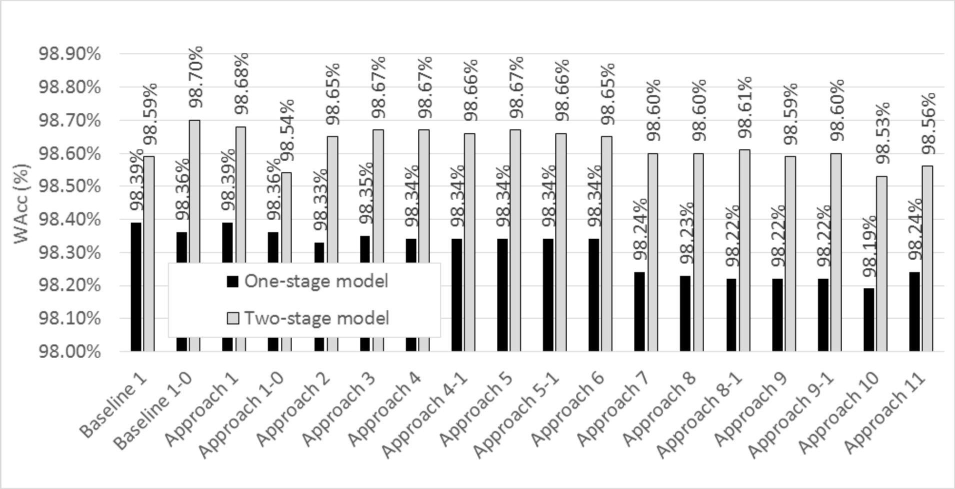
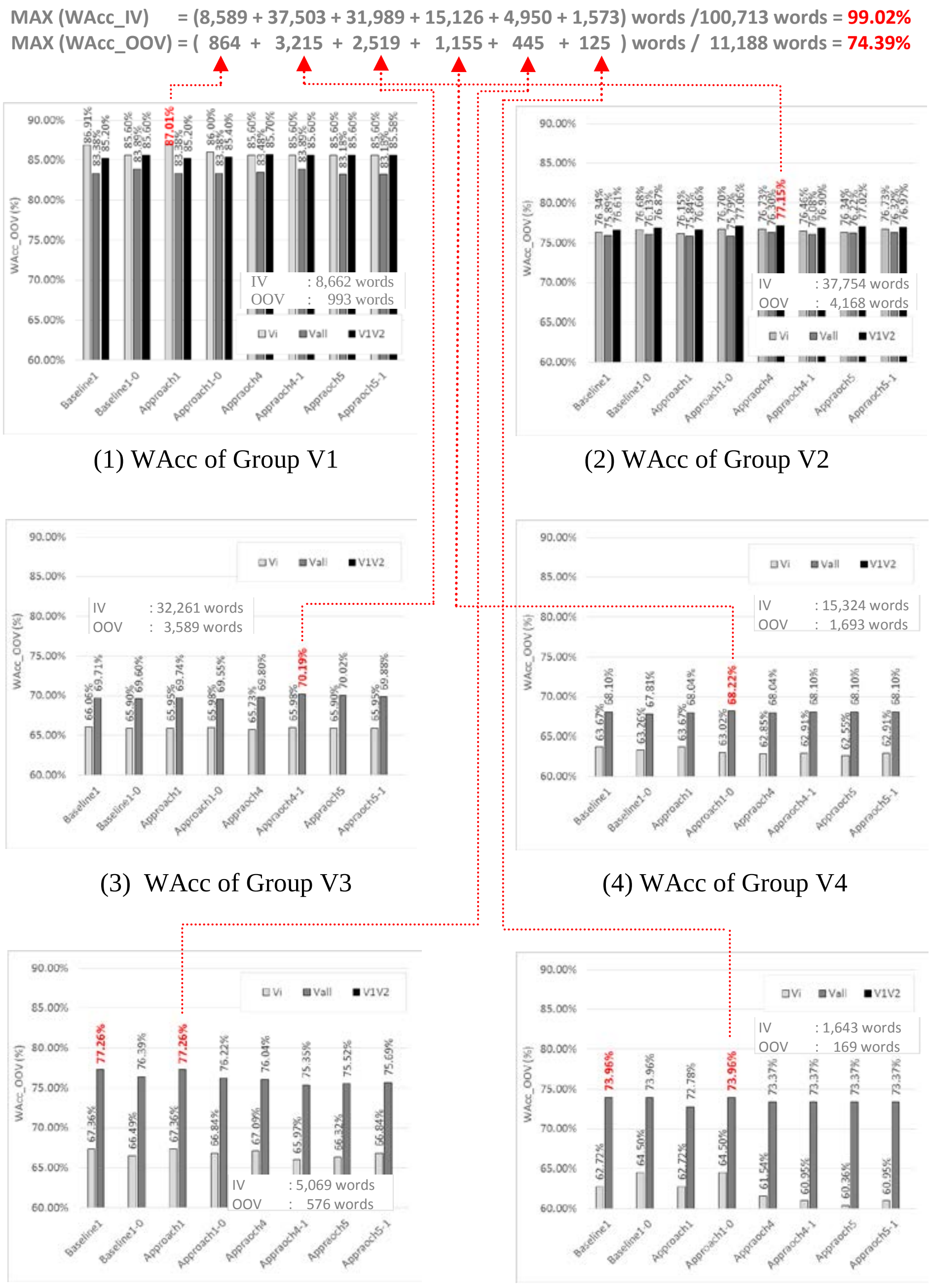


Figure 3 WAcc of different proposed test sets measured based on IV dataset.



MAX (WAcc_IV) = (8,589 + 37,503 + 31,989 + 15,126 + 4,950 + 1,573) words /100,713 words = 99.02%
MAX (WAcc_OOV) = (864 + 3,215 + 2,519 + 1,155 + 445 + 125) words / 11,188 words = 74.39%

(1) WAcc of Group V1

(2) WAcc of Group V2

(3) WAcc of Group V3

(4) WAcc of Group V4

(5) WAcc of Group V5

(6) WAcc of Group V6

Figure 4 Results of the WAcc obtained from the two-stage model-based G2P conversion and separately measured based on different groups of OOV datasets.

Among the proposed approaches that use rules related to the vowel graphemes, Approach2, Approach6 and Approach11 provided lower word accuracy than the others, even including both baselines, so we excluded both of them from the next analysis process. Appraoch3 was also eliminated, because its word accuracy was lower than that of the other approaches, especially Approach4; moreover, GGR3 appeared less effective than the rule used in Approach4. The other approaches, such as Approach7, Approach8, Approach8-1, Approach9, Approach9-1, and Approach10, which use rules focusing on the consonant graphemes rather than the vowel graphemes, were also eliminated because they provided much poorer accuracy compared to the other approaches.

Problems in spelling English words mostly occur when a word has many vowels. Therefore, in order to thoroughly analyze the experimental results, the words in both the training and the testing dataset were classified into six different groups (V_1, V_2, V_3, V_4, V_5, and V_6) depending on the total number of vowels found in each word. The group of words without vowels (V_0) was merged with group V_1, while group V6 included all the remaining groups (V_7, V_8, etc.). The IV and OOV data at the bottom part of Figure 4 show that V2 was the largest group, while V6 was the smallest.

To understand the effects of a different number of vowels in a word and the effects of using different sizes of datasets in the training process, we conducted two different evaluations. First, we trained and evaluated each group of datasets ($V_i = V_{1,} V_2, \dots, V_6$) separately. Second, we used the pre-trained model in Table 2 ($V_{all} = V_1 + \dots + V_6$) to evaluate each group of datasets (V_i) one by one. The evaluation results given by the different approaches are depicted in Figure 4. It shows that the highest values of WAcc for groups V_1 and V_2 (i.e. 87.01% and 76.73%) were obtained using the V_i trained model, while those for the remaining groups were obtained using the V_{all} trained model. This demonstrates that the pronunciation rules in words with zero, one and two vowels are more consistent than those in words with more vowels. In addition, in the largest group V_2, only 10% of the words consisted of VVC syllables. Based on these facts, we conducted another experiment, where we trained the model using a combined V_1 V_2 training dataset (i.e. V_1+V_2) and then evaluated each group V_1 and V_2 separately. As a result, the WAcc of V_2 increased from 76.73% to 77.15%.

We also conducted some experiments in which we kept the G-P pairs with the grapheme mapped to the empty phoneme (e.g. “A._” or “E._” as shown in Figure 1), however we did not report those results in this paper because there was not much difference between the absence and presence of the empty phoneme in the G-P combining method.

6 Discussion

The experimental results in Figure 2, Figure 3 and Figure 4 clearly show that our newly proposed rules (GGR3, GGR4, and GGR5) were more effective than the rules representing unigram and bigram models (GGR1 and GGR2) since they could help improve the model's performance. However, the results given by Approach6 allow us to assume that the strongest rule, such as in this case GGR6, does not always lead to the highest performance because it increases the complexity of the training datasets. In addition, the rules that are designed to enable extra detail for the consonant graphemes (i.e., GGR7, …, GGR11) were not helpful in tackling the problem concerning G2P conversion at all and also degraded the model's performance.

In the one-stage model-based G2P conversion, even though the most effective rules were applied, the WAcc of the IV datasets was very difficult to improve, since it was already very high (for Baseline1, WAcc= 98.39%). However, it could still be improved by adding the second stage. As a result, the two-stage model-based G2P conversion appears to keep almost the same WAcc for the OOV datasets and boosts the WAcc of the IV dataset (i.e. +0.2% ~ +0.3% in WAcc= 200~300 words difference). Therefore, we believe that our proposed approach also can improve the WAcc of the OOV dataset if we select the OOV words carefully, as other researchers have done [20]-[22]. According to an extra experiment, the newly prepared training and testing datasets (which consist of 100,564 and 11,125 words, respectively) selected only words with grapheme-phoneme pairs appearing at least four times in both datasets. The newly obtained results based on the one-stage architecture prove that our proposed approach using GGR5 (Approach4) outperformed the baseline approach (Baseline 1-0) ($p < 0.05$), while obtaining 73.89% and 73.54% as WAcc of the OOV dataset using Approach4 and Baseline 1-0, respectively.

Figure 4 shows that the highest accuracy for each group of OOV datasets (V1…V6) was obtained using different approaches, which means that it appears to be very difficult to use only one approach to solve all the problems associated with G2P conversion. Therefore, this experiment demonstrates that at least five different approaches are required to reach the maximum value of WAcc related to the OOV datasets. After selecting only the trained models providing a maximum value of WAcc for each group of OOV datasets, we obtained 74.39% and 99.02% as the WAcc of the OOV and IV datasets, respectively. These results show that, if we are able to correctly pick the best output phonemes from several results given by different models, then this combined technique could outperform the baseline approaches (i.e. 0.94% = 108 words difference for the OOV dataset and 0.63% = 634 words difference for the IV datasets).

7 Conclusion

It has been shown in this paper that using new grapheme generation rules that are designed to enable extra detail for vowel graphemes can improve the performance of G2P conversion. The new phoneme prediction method allows the second-stage model to learn the pronunciation rules more easily than the first-stage model because both the grapheme sequences and the preliminary phoneme sequences have already been identified at the input level. Moreover, we have shown that using a single-stage approach is not sufficient to deal with all the problems associated with G2P conversion, because each approach is designed using different technique to address different challenges and therefore, using various approaches proves very helpful in solving different specific problems.

In the future, we plan to design more effective rules to reduce the complexity of pronunciation in both training and testing datasets. This can potentially boost the word accuracy of our proposed approach to a higher level. The method using pseudo-phonemes presented in [6] will also be helpful for further improvement of our approach. Furthermore, we will integrate a confusion network data structure [23] and the voting schemes implemented in the NIST Recognizer Output Voting Error Reduction (ROVER) system [24] into the proposed method. This integration will allow us to design an accurate architecture that is able to combine different approaches for tackling different problems concerning G2P conversion.

Acknowledgements

This work is supported by a Grant-in-Aid for Young Scientists (B) 24700167 2012 from MEXT, Japan.

References

[1] Novak, J.R., Dixon, P.R., Minematsu, N., Hirose, K., Hori, C. & Kashioka, H., *Improving WFST-based G2P Conversion with Alignment Constraints and RNNLM N-best Rescoring*, in Proc. of Interspeech, Portland, Oregon, USA, September 9-13 2012.

[2] Kheang, S., Iribe, Y. & Nitta, T., *Letter-To-Phoneme Conversion based on Two-Stage Neural Network focusing on Letter and Phoneme Contexts*, in Proc. of Interspeech, pp. 1885-1888, Firenze Fiera, Florence, Italy, 2011.

[3] Kheang, S., Katsurada, K., Iribe, Y. & Nitta, T., *Solving the phoneme conflict in Grapheme-To-Phoneme Conversion using a Two-Stage Neural Network-based Approach*, The Journal of the Institute of Electronics,

Information and Communication Engineers, E97-D, 4, **8**(4), pp. 901-910, 2014.

[4] Chomsky, N. & Halle, M., *The Sound Pattern of English*, New York: NY: Harper and Row, 1968.

[5] Sejnowski, T.J. & Rosenberg, C.R., *Parallel Networks that Learn to Pronounce English Text*, Complex Systems **1**, pp. 145-168, 1987.

[6] Davel, M. & Barnard, E., *Pronunciation Prediction with Default & Refine*, Computer Speech and Language, **22** Science Direct, Elsevier, pp. 374-393, January 2008.

[7] Taylor, P., *Hidden Markov Models for Grapheme to Phoneme Conversion*, in Proc. of Interspeech, Centro Cultural de Belém, Lisbon, Portugal, September 4-8 2005.

[8] Claveau, V., *Letter-to-phoneme Conversion by Inference of Rewriting Rules*, in Proc. of Interspeech, pp. 1299-1302, Brighton, UK, September 6-10 2009.

[9] Marchand, Y. & Robert, I.D., *A Multi-strategy Approach to Improving Pronunciation by Analogy*, Journal of Computational Linguistics, **26**(2), pp. 195-219, 2000.

[10] Bilcu, E.B., *Text-To-Phoneme Mapping Using Neural Networks*, PhD dissertation, Tampere University of Technology, October 2008.

[11] Marks, J.E. & Fabio, A., *Neural Networks for Text-to-Speech Phoneme Recognition*, IEEE International Conference on Systems, Man & Cyberbetics, pp. 3582-3587, 2000.

[12] Miller, G.A., *Language and Speech*, W.H. Freeman and Company, San Francisco, 1981.

[13] Rama, T., Singh, A.K. & Kolachina, S., *Modeling Letter-to-Phoneme Conversion as a Phrase Based Statistical Machine Translation Problem with Minimum Error Rate Training*, in Proc. of the NAACL HLT Student Research Workshop and Doctoral Consortium, Boulder, Colorado, USA, June 2009.

[14] Ogbureke, K.U., Cahill, P. & Berndsen, J.C., *Hidden Markov Models with Context-Sensitive Observations for Grapheme-to-Phoneme Conversion*, in Proc. of Interspeech, Japan, 2010.

[15] Auto-aligned CMUDict corpus, *Letter-to-Phoneme Conversion Challenge: 10 folds datasets*, http://pascallin.ecs.soton.ac.uk/Challenges /PRONALSYL/Datasets/ (November 2012).

[16] Che, H., Tao, J. & Pan, S., *Letter-To-Sound Conversion using Coupled Hidden Markov Models for Lexicon Compression*, in Proc. of International Conference on Speech Database and Assessments (Oriental COCOSDA), Macau, pp. 141-144, December 2012.

[17] Bisani, M. & Ney, H., *Joint-Sequence Models for Grapheme-to-Phoneme Conversion*, Speech Communication, **50**(5), pp. 434-451, 2008.

[18] Jiampojamarn, S., Kondrak, G. & Sherif, T., *Applying Many-to-Many Alignments and Hidden Markov Models to Letter-to-Phoneme Conversion*, in the Conference of the North American Chapter of the Association for Computational Linguistics and Human Lanauge Technology (NAACL HLT), Rochester, New York, April 2007.

[19] Jiampojamarn, S. & Kondrak, G., *Letter-Phoneme Alignment: An Exploration*, in Proc. of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, July 2010.

[20] Hahn, S., Vozila, P. & Bisani, M., *Comparison of Grapheme-to-Phoneme Methods on Large Pronuciation Dictionaries and LVCSR Tasks*, in Proc. of Interspeech, Portland, Oregon, USA, September 9-13 2012.

[21] Libossek, M. & Schiel, F., *Syllable-based Text-to-Phoneme Conversion for German*, in Proc. of the Sixth International Conference on Spoken Language Processing (ICSLP), pp. 283-286, Beijing, China, 2000.

[22] Lehnen, P., Hahn, S., Guta, V.A. & Ney, H., *Hidden Conditional Random Fields with M-to-N Alignments for Grapheme-to-Phoneme Conversion*, in Proc. of Interspeech, Portland, Oregon, USA, September 9-13 2012.

[23] Furuya, Y., Natori, S., Nishizaki, H. & Sekiguchi, Y., *Introduction of False Detection Control Parameters in Spoken Term Detection*, in APSIPA ASC, Hollowood, CA, 2012.

[24] Fiscus, J.G., *A Post-Processing System to Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER)*, in Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding, Canada, pp. 347-354, Santa Barbara, CA, December 1997.