IEICE TRANS. INF. & SYST., VOL.E99–D, NO.4 APRIL 2016

PAPER

Using Reversed Sequences and Grapheme Generation Rules to Extend the Feasibility of a Phoneme Transition Network-Based Grapheme-to-Phoneme Conversion

Seng KHEANG[†a)], ***Nonmember***, **Kouichi KATSURADA**[†], **Yurie IRIBE**[††], ***and*** **Tsuneo NITTA**[†,†††], ***Members***

SUMMARY The automatic transcription of out-of-vocabulary words into their corresponding phoneme strings has been widely adopted for speech synthesis and spoken-term detection systems. By combining various methods in order to meet the challenges of grapheme-to-phoneme (G2P) conversion, this paper proposes a phoneme transition network (PTN)-based architecture for G2P conversion. The proposed method first builds a confusion network using multiple phoneme-sequence hypotheses generated from several G2P methods. It then determines the best final-output phoneme from each block of phonemes in the generated network. Moreover, in order to extend the feasibility and improve the performance of the proposed PTN-based model, we introduce a novel use of right-to-left (reversed) grapheme-phoneme sequences along with grapheme-generation rules. Both techniques are helpful not only for minimizing the number of required methods or source models in the proposed architecture but also for increasing the number of phoneme-sequence hypotheses, without increasing the number of methods. Therefore, the techniques serve to minimize the risk from combining accurate and inaccurate methods that can readily decrease the performance of phoneme prediction. Evaluation results using various pronunciation dictionaries show that the proposed model, when trained using the reversed grapheme-phoneme sequences, often outperformed conventional left-to-right grapheme-phoneme sequences. In addition, the evaluation demonstrates that the proposed PTN-based method for G2P conversion is more accurate than all baseline approaches that were tested.

key words: *reversed sequences, grapheme generation rule (GGR), Grapheme-to-Phoneme (G2P) conversion, combining multiple approaches, phoneme transition network (PTN)*

1. Introduction

Data-driven grapheme-to-phoneme (G2P) conversion is a process used to predict the phoneme strings corresponding to out-of-vocabulary (OOV) words. G2P conversions are commonly implemented for speech synthesis, language-learning software, and spoken-term detection systems.

Often, there is no strict correspondence between letters and phonemes in spoken words, and this is especially true for an orthographically irregular language like English [1]. Thus, researchers have proposed various data-driven methods using many-to-many mapping techniques between graphemes and phonemes. Methods have been proposed based on hidden Markov models (HMMs) [2], [3], artificial neural network (ANNs) [4], joint-sequences [5], margin-infused relaxed algorithms (MIRAs) [6], [7], a weighted finite-state transducer (WFST) [8], an adaptive regularization of weight vectors (AROW) [9], a narrow adaptive regularization of weight vectors (NAROW) [10], and structured soft-margin confidence weighted learning (SSMCW) [11]. Most of these methods, and especially SSMCW-based G2P conversion that is implemented in the Slearp toolkit*, have demonstrated significantly accurate results. However, each of these methods has been designed using specific techniques that address particular challenges faced by G2P conversion. Therefore, any single approach will not suffice when addressing all of the problems encountered by G2P conversion [12]. Considering this, a combination of various approaches using different methods is a reasonable strategy for treating these problems in a flexible manner.

Combining various methods can both lend flexibility to the conversion and improve its predictive performance. Thus, in this paper we present a Phoneme Transition Network (PTN)-based architecture for G2P conversion. Basically, our proposed PTN-based method first converts a target word into multiple phoneme strings using several data-driven methods. Then, it aligns the obtained results—the phoneme-sequence hypotheses—using a dynamic-programming (DP) algorithm, combining them into a confusion network (hereafter referred to as the "PTN"), and determining the best phoneme from each PTN bin—a block of phonemes/transitions between two nodes in the PTN—to represent the final output. The best phoneme selection in this study is based on a voting strategy according to the frequency and maximum confidence score of the occurrences implemented in the Recognizer Output Voting Error Reduction (ROVER) system [13].

Selecting the set of methods used by the proposed architecture is a crucial task. If accurate methods are combined with inaccurate methods, this can considerably degrade the performance of the entire PTN-based G2P conversion model. For example, Schlippe et al. merged five phoneme-sequence hypotheses generated from five different methods to enhance the generation of pronunciation in low-

Manuscript received August 31, 2015.
Manuscript revised November 23, 2015.
Manuscript publicized January 6, 2016.

[†]The authors are with Toyohashi University of Technology, Toyohashi-shi, 441–8580 Japan.
[††]The author is with Aichi Prefectural University, Nagakute-shi, 480–1198 Japan.
[†††]The author is with Waseda University, Tokyo, 169–8050 Japan.
a) E-mail: kheang@vox.cs.tut.ac.jp

DOI: 10.1587/transinf.2015EDP7349

*Slearp: http://osdn.jp/projects/slearp/

resource scenarios [14]. However, they could not demonstrate any significant improvement using this combined approach without the addition of web-derived pronunciation dictionaries. Even so, this improvement deteriorated as the size of the training data increased, especially for a difficult language like English. On the other hand, when the number of phoneme-sequence hypotheses generated from inaccurate models was more than the number of those generated from accurate models, it was difficult to maintain and improve the performance of the PTN-based model [15].

In order to mitigate this risk, we selected a minimum number of methods.[†] We also present a novel use for right-to-left (reversed) grapheme-phoneme (g-p) sequences and grapheme generation rules (GGRs) [12]. In this study, both techniques are especially helpful for extending the feasibility and improving the performance of PTN-based G2P conversion, because they increase the number of phoneme-sequence hypotheses without increasing the number of methods used. By reversing the conventional (left-to-right reading direction) g-p sequence, we can provide context information that differs from conventional sequences during the alignment. This allows each single method to train an additional model, thus producing an additional phoneme-sequence hypothesis. In addition, applying various GGRs[††] to the words (that satisfy the rules) in the source corpus will also generate additional grapheme-sequences and more training samples. This increases the size of training data, enabling a single trained model to produce more than one phoneme-sequence hypothesis. Therefore, this paper proposes two different versions of the PTN-based architecture for G2P conversion. As a result of the reversed g-p sequences, the first architecture uses only three different methods, based on MIRA [7], WFST [8], and SSMCW [11], to train six separated source models in order to generate six phoneme-sequence hypotheses. To reduce the number of methods as well as the number of trained models, we use only a single GGR rule for the second architecture. Consequently, this architecture requires only four models based only on a single method (viz., an SSMCW-based method) to generate the same number of hypotheses.

We evaluated our proposed models against the three baseline methods mentioned in the previous paragraph using multiple datasets and the K-fold cross-validation technique. The results indicate an improvement in both phoneme and word accuracy with respect to OOV words.

The remainder of this paper is organized as follows. In Sect. 2, we describe the three data-driven methods for G2P conversion selected for this study. We then present the PTN-based G2P conversion and its compact version in Sects. 3 and 4, respectively. The evaluation results and discussion are presented in Sects. 5 and 6, respectively. The conclusion is given in Sect. 7.

[†]In previous research, a method/approach has been used to train a single model only, so the terms "method/approach" and "model" might have a similar meaning. Otherwise, here, we differentiate between them because a single selected method in this paper can be used to train more than one model.

[††]In English, the interaction between vowels in a word strongly affects its spelling. Thus, GGRs were originally proposed to add extra-sensitive information to each vowel-grapheme appearing in a word.

2. Different Data-Driven Methods for G2P Conversion

Many data-driven approaches to G2P conversion have been proposed, but the popular joint-sequence or n-gram model-based methods for G2P conversion have been proven to be the most powerful techniques for dealing with OOV words. Because our proposed approach requires the combination of at least three methods, we selected the three most powerful statistical-based methods that differently encode the n-gram model.

2.1 MIRA-Based Method for G2P Conversion (DIRECTL+)

The best-known joint n-gram model-based method for G2P conversion was first proposed in 2008 by Bisani and Ney [5], and it was implemented as a generative system available in the Sequitur toolkit.[†††] In this system, the model is trained using the expectation-maximization algorithm, and the phoneme sequence corresponding to a given word $\varphi(g)$ is predicted through a Bayes' decision rule as follows:

$$\varphi(g) = argmax_{\varphi'} P(g, \varphi') \quad (1)$$

Here, g represents a given grapheme sequence, where φ' is the most likely pronunciation of the grapheme sequence g.

Soon after, Jiampojamarn et al. represented the joint n-grams model for G2P conversion as an online discriminative sequence-prediction model, which used a many-to-many alignment between grapheme and phoneme sequences and a feature vector consisting of n-grams context features, HMM-like transition features, and linear-chain features [6]. For each training iteration, the feature weight vector was updated using the MIRA algorithm; this system is called DirecTL. The updated version of DirecTL is called the DIRECTL+ toolkit,[††††] implemented in 2010, in which the joint n-gram features were integrated [7].

2.2 Rapid WFST-Based G2P Conversion (Phonetisaurus)

A WFST-based method for G2P conversion proposed by Novak et al. [8] has been implemented to develop a rapid and high-quality joint-sequence model-based G2P conversion. First, the training words and their phoneme sequences are provided, and these are aligned using an expectation-maximization training procedure based on the many-to-many aligning technique [5]. The joint-sequence corpus is given as an input for n-gram counting (in which the order or length of the n-grams to count is provided), and then a stan-

[†††]http://www-i6.informatik.rwth-aachen.de/web/Software/g2p.html

[††††]https://code.google.com/p/directl-p/

dard joint n-gram model is trained using the MITLM tookit[†] or the OpenGrm NGram library,[††] and smoothed by Kneser-Ney discounting with interpolation. Then, the trained n-gram model is converted to a WFST-based model, which predicts the phoneme sequences of unknown words using the following decoding function:

$$Phseq_{best} = shortestPath(Project_o(W \circ M)) \quad (2)$$

where "$Phseq_{best}$" refers to the most likely phoneme sequence given the input word "W" under the FSA representation and the n-gram model "M" encoded as FST, "$\circ$" refers to the weighted composition, "$Project_o(.)$" is a projection onto the output symbols, and "$shortestPath(.)$" indicates the shortest-path algorithm.

2.3 SSMCW-Based G2P Conversion (Slearp)

Structured online discriminative learning methods, such as structured AROW [9] and NAROW [10], have been successful at improving performance in G2P conversion. Recently, an SSMCW-based method [11] has been proposed for extending multi-class confidence-weighted learning to structured learning, which softens the marginal errors for hypothesis and update parameters using the N-best hypotheses simultaneously and interdependently for robustness against over-fitting.

The general formulation of a G2P conversion model using a structured learning method is as follows:

$$\hat{y} = argmax_y \omega^T \Phi(x, y) \quad (3)$$

where the parameters x and y represent a given grapheme sequence and its corresponding phoneme sequence, respectively, ω indicates the weight vector for the classifier, and $\Phi(x, y)$ is a feature vector that consists of the frequencies of joint n-gram features on x and y. The predicted phoneme sequence $\hat{y}$ is obtained using a dynamic-programming algorithm. For a detailed discussion of how the parameters in Eq. (3) are determined, please refer to [11].

3. PTN-Based Architecture for G2P Conversion

In this section, we first introduce a novel use of reversed g-p sequences and explain how PTN sequences are generated from multiple phoneme sequences. Then, we describe how to determine the best output phoneme sequence from the PTN sequence using voting techniques.

3.1 Reversed g-p Sequences

To predict a phoneme sequence corresponding to an input grapheme sequence, most existing approaches use an n-gram model to calculate the likelihood probability that a phoneme (sequence) accurately corresponds to a particular grapheme (sequence) [2], [5], [7]–[11]. This means that only the context from left to right is seen by the model. Thus, the trained model can only learn or cover the relationship between graphemes and phonemes in a single direction.

According to [16], Sutskever et al. reversed the order of input words in all source sentences, but not in the target sentences, and this was done in order to train a machine-translation model using a multi-layered Long Short-Term Memory Recurrent Neural Network (LSTM-RNN). This cross-mapping technique is possible owing to Connectionist Temporal Classification (CTC) [17], which allows the RNNs to be trained without requiring any prior alignment between the source and target sequences. Sutskever et al. demonstrated that this reversed-word model (slightly) outperformed models based on conventional word sequences.

However, this cross-mapping technique is inadequate for statistical-based methods where a prior alignment between input and output sequences is required [2], [5], [7]–[9], [11].[†††] Therefore, in this paper we introduce a new way to use the reversing technique for G2P conversion, such that it avoids alignment problems. Rather than reversing only the input grapheme sequence, we reverse both the input grapheme and the output phoneme sequences, as demonstrated in the following example:

- Conventional g-p sequences: "LURIE"→/L UH R IY/
- Reversed g-p sequences: "EIRUL"→/IY R UH L/

3.2 PTN Generation Using Multiple Phoneme Sequences

Over the last few years, it has proven considerably difficult to improve the performance of a G2P conversion model for OOV words, because each method or approach is uniquely designed using different techniques to address particular challenges. It is seemingly impossible to utilize any single method to deal comprehensively with the host of problems encountered by G2P conversion [12]. Therefore, we designed a PTN-based architecture for G2P conversion that allows many different methods to be applied together, in order to deal broadly with the various problems.

The number of methods used by the PTN-based G2P conversion model, as well as the methods themselves, must be carefully selected, owing to the risk of combining accurate methods with inaccurate ones such that the performance of the entire model is degraded. In order to minimize this risk, only a few accurate methods should be used. By contrast, combining only a minimum number of phoneme-sequence hypotheses will not improve the PTN-based G2P conversion [15].

Therefore, in this study, we propose a novel PTN-based architecture using the three most accurate methods for G2P conversion: the SSMCW-based method (available in the Slearp toolkit), the WFST-based method (available in the Phonetisaurus toolkit), and the MIRA-based method (available in the DIRECTL+ toolkit). As depicted in Fig. 1,

[†] https://code.google.com/p/mitlm/

[††] http://www.openfst.org/twiki/bin/view/GRM/NGramLibrary

[†††] We also conducted tests for G2P conversion, but the results were completely unsuitable, because the grapheme in a left-to-right direction must be aligned to the phoneme in the reversed direction.

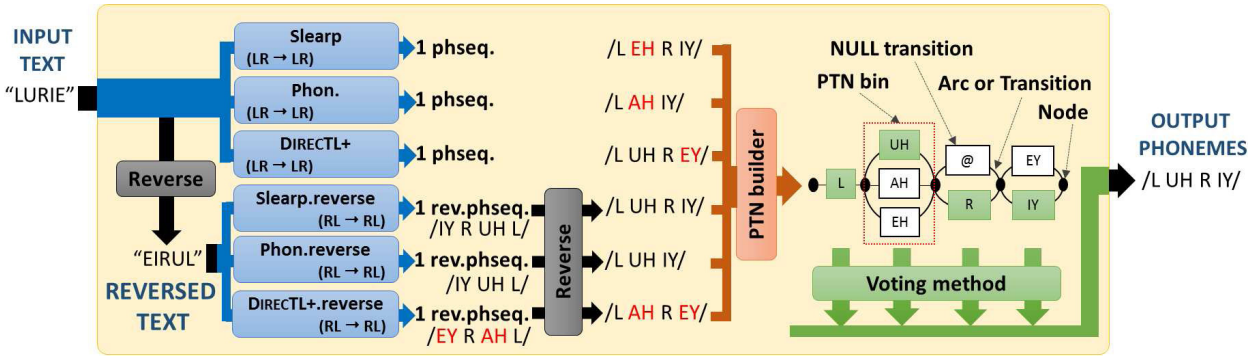

Fig. 1 Architecture for the first proposed PTN-based G2P conversion using six models based on three different methods. (LR→LR) and (RL→RL) represent the models trained using the conventional and reversed g-p sequences, respectively.

by using the conventional g-p sequences as training data, we can generate three phoneme-sequence hypotheses from three source models: Slearp, Phonetisaurus (Phon.), and DIRECTL+. Furthermore, the reversed g-p sequences allow these three methods to produce three additional models: Slearp.reverse, Phonetisaurus.reverse (Phon.reverse), and DIRECTL+.reverse. In total, six phoneme-sequence hypotheses are generated from six models implemented using only three methods.

The ROVER system [13] allows us to align these phoneme sequences using a DP algorithm, and to merge them together in a single confusion network (or PTN), as shown in Fig. 1. In this context, when there is any insertion or deletion problems during the alignment, a null phoneme /@/ is used by the PTN to represent a null transition.

3.3 Determining the Best Output Phoneme

Theoretically, many phoneme sequences can be generated from a PTN, but only a single sequence is needed to represent the best output of the model. In order to determine the best output sequence, we adopted a voting strategy, according to the frequency and maximum confidence score of the occurrences. This voting scheme is provided in the ROVER system [13]. The phoneme-selection function for each PTN bin is based on the following scoring formula:

$$score(ph) = \alpha(N(ph,i)/n) + (1-\alpha)C(ph,i) \quad (4)$$

$$C(ph,i) = MAX(conf_1(ph,i),\ldots,conf_n(ph,i)) \quad (5)$$

where $N(ph,i)$ is the number of occurrences of the phoneme ph in the ith PTN bin, and n denotes the number of phoneme-sequence hypotheses. Furthermore, $C(ph,i)$ represents the confidence score for the phoneme ph in the ith PTN bin, where $conf_1(ph,i),\ldots,conf_n(ph,i)$ is the set of confidence scores for ph in the ith PTN bin that correspond to the various sequence hypotheses. The real value $\alpha = [0\ldots1]$ represents a trade-off between the phoneme frequency and the confidence score in Eq. (5).

Table 1 Examples of GGR_r rules.

GGR_0	$g_i \Rightarrow g_i$ (like unigram = default)
	Ex: "OKEECHOBEE" ⇒ O K E E C H O B E E
GGR_1	$g_i \Rightarrow g_i g_{i+1}$ (like bigram)
	Ex: "OKEECHOBEE" ⇒ OK KE EE EC CH HO OB BE EE E_
GGR_2	If $(n > 1)$: $v_1 \ldots v_n c_{n+1} \Rightarrow v_1 v_2 \;\; v_2 v_3 \;\; \ldots \;\; v_{n-1} v_n \;\; v_n \;\; c_{n+1}$
	If $(n = 1)$: $g_i \Rightarrow g_i$
	Ex: "OKEECHOBEE" ⇒ O K **EE** E C H O B **EE** E

Here, $g_i = \{c_i, v_i\}$: grapheme at index i;
c_i, v_i: consonant and vowel graphemes at index i;
n: number of connecting vowels in a given word.

4. Reducing the Number of Required Source Models

Even if the reversed g-p sequences can make a complementary model that can generate an additional phoneme-sequence hypothesis for each source method, the risk from combining different methods nevertheless remains. Hence, we introduce a novel use of grapheme generation rules (GGRs) [12] to minimize this risk. This allows us to use only a single method for implementing a PTN-based G2P conversion model.

4.1 Grapheme Generation Rules (GGRs)

Textual information does not supply a sufficient amount of information relating to the phonological interaction [18]. In orthographically complex languages such as English, the interaction between vowels in a word significantly affects the spelling. Hence, a technique for generating new grapheme sequences from the same input text (known as GGRs) has been proposed for adding extra-sensitive information to each vowel-grapheme appearing in a word [12]. Suppose that a grapheme sequence $g = g_1 g_2 \ldots g_n$ is provided as an input. The new grapheme sequence $\hat{g}_r = \bar{g}_1 \sqcup \bar{g}_2 \sqcup \ldots \sqcup \bar{g}_n$, in which an empty space is used as a separator, can be generated with respect to a rule GGR_r, formulated as follows:

$$\hat{g}_r = GGR_r(g) \quad (6)$$

A list of few rules, which is selected from [12] and designed to tackle the connecting vowels in the English lan-

Table 2 Example of a newly generated dataset when various GGR_r rules are applied. Here, the g-p sequences in the source dataset are selected from the CMUDict_noisy corpus.

	Grapheme sequence	→Phoneme sequence	GGR_r
Source (S)	NEWLY	→N UW L IY	
	CREATIVE	→K R IY EY T IH V	
	IDEA	→AY D IY AH	
GGR$_0$(S) ($\hat{S}_0$)	N E W L Y	→N UW L IY	GGR_0
	C R E A T I V E	→K R IY EY T IH V	GGR_0
	I D E A	→AY D IY AH	GGR_0
GGR$_2$(S) ($\hat{S}_2$)	N E W L Y	→N UW L IY	GGR_2
	C R **EA** A T I V E	→K R IY EY T IH V	GGR_2
	I D **EA** A	→AY D IY AH	GGR_2
$\hat{S}_0 \cup \hat{S}_2$ ($\hat{S}$)	N E W L Y	→N UW L IY	GGR_0 or GGR_2
	C R E A T I V E	→K R IY EY T IH V	GGR_0
	C R **EA** A T I V E	→K R IY EY T IH V	GGR_2 (+)
	I D E A	→AY D IY AH	GGR_0
	I D **EA** A	→AY D IY AH	GGR_2 (+)

guage, is provided in Table 1. In this study, we selected only the rules GGR_0 and GGR_2 for our first-time experiments because we wanted to investigate the difference between the baseline rule GGR_0 and the best rule GGR_2 from [12] when used within the PTN-based G2P conversion. The rule GGR_0 is equivalent to the conventional grapheme sequence (where the space is ignored), but GGR_2 can distinguish the separated vowel v in the *cvc* pattern from the connecting vowels $v_1, v_2, \ldots, v_{n-1}$ in the $v_1 v_2 \ldots v_n$ pattern.

4.2 PTN-Based G2P Conversion Using Only One Method

According to Fig. 1, after using the reversed g-p sequences, only three different methods are required for generating six phoneme-sequence hypotheses used in the PTN-based G2P conversion. However, the source/trained models remain the same six models. Hence, the integration of GGRs into the source models is especially helpful.

Rather than using only the original word-pronunciation pairs from the source corpus, we applied several GGRs to all the words, in order to generate additional g-p sequences. These were then added to the dataset; the redundant g-p sequences were omitted. According to the example in Table 2, we suppose that a source dataset (i.e., $S = \{(g,p)_1, (g,p)_2, \ldots, (g,p)_N\} = \bigcup_{k=1}^{N}(g,p)_k$) consists of N pairs of g-p sequences. Then, a set of R rules is applied, and the newly generated dataset $\hat{S}$ is formulated as follows:

$$\hat{S} = \bigcup_{r=1}^{R} \hat{S}_r = \bigcup_{r=1}^{R} GGR_r(S) = \bigcup_{r=1}^{R}\bigcup_{k=1}^{N}(GGR_r(g), p)_k$$
$$\Rightarrow \hat{S} = \bigcup_{r=1}^{R}\bigcup_{k=1}^{N}(\hat{g}_r, p)_k \quad (7)$$

As a result, for each input word (refers to the conventional grapheme sequence g), Fig. 2 shows that it is possible to generate more than one phoneme sequence from a trained model in which the newly generated dataset $\hat{S}$ is used (e.g., Slearp.GGR$_{0+2}$), given the different representations of its grapheme sequence (e.g., the generated

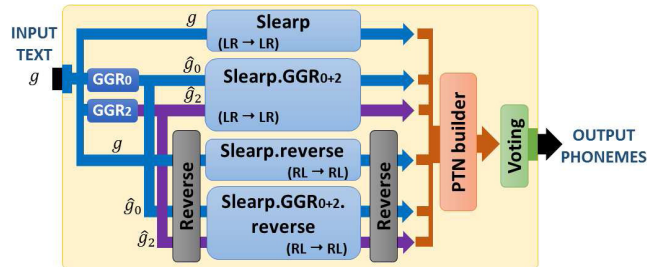

Fig. 2 Architecture for the second proposed PTN-based G2P conversion using four models based on only a single method from the Slearp toolkit.

grapheme sequences $\hat{g}_0 = GGR_0(g)$ and $\hat{g}_2 = GGR_2(g)$ seen in Table 2). By using both reversed g-p sequences and various GGRs, the number of generated hypotheses Nb_{hyps} can be calculated using the following formula:

$$Nb_{hyps} = \begin{cases} 2 * Nb_{GGRs}, & \text{if the reversed seq}_s \text{ are used} \\ Nb_{GGRs}, & \text{otherwise} \end{cases} \quad (8)$$

where Nb_{GGRs} indicates the number of applied rules.

The novel use of GGRs in G2P conversion allows us to use only one method to train one or several models combined at the PTN level. In this study, we compared the performance among the models using GGRs with those using conventional and reversed g-p sequences. Therefore, as seen in Fig. 2, the second proposed PTN-based architecture for G2P conversion combines six hypotheses generated from four models implemented using only a single method (i.e., the most accurate SSMCW-based method for G2P conversion available in the Slearp toolkit). The Slearp and Slearp.reverse models are trained using the original dataset S, and thus producing only two phoneme-sequence hypotheses. The Slearp.GGR$_{0+2}$ and Slearp.GGR$_{0+2}$.reverse models are trained using the newly generated dataset $\hat{S}$, and thus possibly generating four phoneme-sequence hypotheses. Although the input grapheme sequences g and $\hat{g}_0$ are equivalent, two different phoneme-sequence hypotheses might be produced owing to the different source models.

5. Evaluation

In this section, we describe the data-preparation process and the experimental setup. Subsequently, we report the experimental results.

5.1 Data Preparation

The performance of our two proposed approaches was evaluated relative the baseline models discussed in Sect. 2. We conducted experiments using four different pronunciation dictionaries (three in English and one in French), as listed in Table 3. The NETtalk, Brulex and CMUdict datasets were obtained from the Pascal Letter-to-Phoneme Conversion Challenge website†. A noisy CMUdict dataset (CMU-

†http://pascallin.ecs.soton.ac.uk /Challenges/PRONALSYL/Datasets

Table 4 Phoneme (PAcc) and word accuracy (WAcc) for all baseline and PTN-based G2P conversion models. The italicized text indicates the highest accuracy among the baseline models. The text is bold where a PTN provided a better result than all the baselines, and the background is gray when the PTN(1+...+6) outperformed both PTN(1+2+3) and PTN(4+5+6).

	NETtalk		Brulex		CMUdict		CMUdict_noisy	
	PAcc	WAcc	PAcc	WAcc	PAcc	WAcc	PAcc	WAcc
1_Slearp	93.66%	70.99%	*99.15%*	*95.65%*	93.60%	73.12%	93.83%	73.55%
2_Phon.	92.89%	68.56%	98.95%	94.52%	93.25%	72.39%	93.48%	72.71%
3_DirecTL+	93.75%	71.31%	98.20%	92.54%	92.61%	70.91%	92.37%	70.11%
4_Slearp.reverse	*93.79%*	*71.93%*	99.14%	*95.55%*	*93.74%*	*73.91%*	*93.84%*	*73.96%*
5_Phon.reverse	93.07%	69.15%	98.93%	94.43%	93.30%	72.53%	93.54%	73.10%
6_DirecTL+.reverse	93.65%	70.89%	98.20%	92.55%	92.19%	69.88%	91.91%	68.92%
PTN(1+2+3)	**94.10%**	**72.73%**	**99.20%**	**95.89%**	93.11%	73.87%	**94.28%**	**75.20%**
PTN(4+5+6)	**94.16%**	**73.14%**	**99.20%**	**95.82%**	**94.06%**	**74.96%**	**94.23%**	**75.25%**
PTN(1+2+3+4+5+6)	**94.23%**	**73.45%**	**99.22%**	**95.98%**	**94.13%**	**75.17%**	**94.28%**	**75.30%**

Table 3 Datasets or corpora used in the experiments.

Dataset	Vocabulary size (words)			
	Train	Dev.	Test	K-fold
NETalk (English)	17,508	1,000	1,500	10
Brulex (French)	23,955	1,000	2,500	10
CMUdict (English)	95,286	6,000	11,000	8
CMUdict_noisy (English)	107,438	5,939	11,998	1
CMUdict_noisy_GGR$_{0+2}$ ($\tilde{S}$)	*130,533*	*7,787*	*15,372*	1

dict_noisy) containing words with multiple pronunciations (i.e., heteronyms) is available in the Phonetisaurus package. In this study, we used the NETtalk corpus to tune the parameters for each method and the ROVER system.

We subdivided each corpus into training, development, and testing datasets. The NETtalk, Brulex, and CMUdict datasets each originally consisted of ten separated folds. Thus, for each trial of cross-validation, one fold was used as the testing data, some data in another fold was randomly selected for the development data, and the eight remaining folds, along with the leftover data from the fold used for the development data, were extracted and combined for use as training data. By contrast, the source of the CMUdict_noisy dataset originally consisted of only two parts (training and testing datasets). Thus, development data was randomly extracted from the training dataset. In order to conduct a fair evaluation, when the same word appeared multiple times with different phoneme sequences in the development or testing dataset, we retained only a single pair.

Owing to the fact that the GGRs in this paper were designed exclusively for English words and the CMUdict_noisy corpus were used in many previous studies [5], [7], [8], we used only this corpus to evaluate our second PTN-based G2P conversion (see Sect. 4). Equation (7) was applied to increase the size of the training, development, and testing datasets, after GGR_0 and GGR_2 were applied, the details for which are provided in Tables 2 and 3. Here, GGR_0 was used to convert the format of the original grapheme sequence by adding a space between two connected graphemes.

5.2 Experimental Setup

5.2.1 Proposed Test Sets

In our experiments, we employed the three original models using the conventional g-p sequences as baseline models—viz., Slearp (*1_Slearp* in Table 4), Phonetisaurus (*2_Phon.*), and DIRECTL+ (*3_DIRECTL+*), presented in Sect. 2.

To see the advantages of using the reversed g-p sequences for G2P conversion, we proposed three additional models (*4_Slearp.reverse, 5_Phon.reverse and 6_DIRECTL+.reverse*) in which the reversed g-p sequences were used in place of the conventional sequences.

As listed in Table 4, in order to compare the performance between G2P conversion based on a single model with G2P conversion based on multiple models, we proposed three PTN-based G2P conversion models. In this case, all six separated models mentioned in the previous paragraph (labeled 1, 2, 3, 4, 5 and 6 in the PTN notation) were considered baseline models. For three-model combinations, we proposed PTN(1+2+3) and PTN(4+5+6) for comparing the performance between the PTN-based model with only the conventional g-p sequences and the one with only reversed g-p sequences. PTN(1+...+6) was proposed both to evaluate the performance of the PTN-based model with all six baseline models and also to observe the effect and risk from combining accurate and inaccurate source models.

On the other hand, in the evaluation of our second PTN-based architecture (see Sect. 4), we implemented four baseline models (viz., *1_Slearp, 2_Slearp.GGR$_{0+2}$, 3_Slearp.reverse*, and *4_Slearp.GGR$_{0+2}$.reverse*), as seen in Fig. 2 and Table 5. The first and third models were trained using the training and development datasets from the original corpus, CMUdict_noisy, whereas the second and fourth models were trained using datasets from the newly generated corpus CMUdict_noisy_GGR$_{0+2}$. For each input word, two different representations of a grapheme sequence can be encoded using GGR$_0$ and GGR$_2$. Thus, two phoneme-sequence hypotheses must be generated from each of the models using GGRs (i.e., 2_Slearp.GGR$_{0+2}$ or 4_Slearp.GGR$_{0+2}$.reverse). In our evaluation, we considered

Table 5 Performance of the compact PTN-based G2P conversion using only the Slearp toolkit, GGRs, and reversed g-p sequences. The bold text and gray background in this table are used in the same manner as Table 4.

	Phoneme-sequence hyp.	CMUdict_Noisy	
Trained model	Model name for evaluation	PAcc	WAcc
1_Slearp	**A-Slearp**	93.83%	73.55%
2_Slearp.GGR_{0+2}	**B-Slearp.GGR_0**	93.93%	74.16%
	C-Slearp.GGR_2	93.96%	74.21%
3_Slearp.reverse	**D-Slearp.reverse**	93.84%	73.96%
4_Slearp.GGR_{0+2}.reverse	**E-Slearp.GGR_0.reverse**	94.08%	74.99%
	F-Slearp.GGR_2.reverse	*94.08%*	*75.08%*
	compactPTN(A+B+C)	93.97%	74.28%
	compactPTN(D+E+F)	**94.11%**	**75.09%**
	compactPTN(A+B+C+D+E+F)	**94.29%**	**75.56%**

these hypotheses as belonging to two separated models. The evaluation results from all the baseline models (A, B, C, D, E and F) in Table 5 were obtained using the same test data—i.e., the same input words—but with different graphemic representations. In order to compare the performance between G2P conversion based on a single model with G2P conversed based on a compact PTN, we proposed the same three PTN models with respect to the evaluation of our first architecture.

5.2.2 Experiment Configurations

According to the results of the preliminary experiments using the NETtalk corpus, the necessary parameters for the three selected methods for G2P conversion were tuned as follows:

- In the DIRECTL+ toolkit, the size of the n-gram context features and joint n-gram features was set to 7 and 3, respectively. Data alignment was based on the mpaligner software [19], and the association between graphemes and phonemes was set to 2-3.
- In the Phonetisaurus toolkit, the number of discounting (*bins*) and the maximum length of n-grams to count (*order*) were set to 3 and 8, respectively.
- In the Slearp toolkit, the size of the n-gram context and chain features was set to 5, while the joint n-gram feature size was set to 7. Pre-alignment was also based on the mpaligner software with m-m association.
- For both Slearp and DIRECTL+, the minimum number of iterations before ending the training process and the maximum number of iterations after a degradation in the performance of the development data were both set to 10. The best iteration was selected based on both phoneme and word accuracy, and this was measured with the development dataset.

In order to improve the performance of the most accurate source models in the PTN-based G2P conversion, a set of confidence scores in Eq. (5) should be assigned with respect to the rank of model accuracies. If $\{a, b, c, d, e, f\}$ is a set of scores for our six baseline models sorted by its accuracy, then each phoneme of the sequence hypothesis generated from the model with highest accuracy was assigned the highest score a and the one generated from the model with lowest accuracy was assigned the lowest score f. Based on our experiments, for both PTN(1+...+6) and compactPTN(A+...+F), the best results were obtained when the values of a, b, c, d, e and f were assigned to 1.0, 0.7, 0.6, 0.5, 0.4 and 0.2, respectively; for the ROVER system, the value of α and the confidence score of NULL phoneme /@/ (noted as $Nconf$) in Eq. (4) and Eq. (5) should be equal to 0.7 and 0.8, respectively. On the other hand, we used only a set of three scores $\{a, b, c\}$ for PTN(1+2+3), PTN(4+5+6), compactPTN(A+B+C) and compactPTN(D+E+F); in this case, the best results were obtained when the values of a, b and c were assigned to 1.0, 0.7, and 0.6, respectively.

To conduct our experiments, we simultaneously executed multiple programs on a shared server (CentOS 6.6, Intel(R) 12-Core(TM) i7-4930K CPU 3.40 GHz, RAM 64 GB, HDD 630 GB) in our laboratory.

5.2.3 Performance Metrics

We evaluated the models' performance in terms of phoneme accuracy (PAcc) and word accuracy (WAcc), using the NIST SCLITE scoring toolkit.[†] In this paper, we report only the results concerning the OOV words in the testing dataset. We also measured the statistical significance (i.e., p-values) using McNemar's test.

5.3 Experimental Results

All of the evaluation results for the baseline models and the G2P conversion models based on our first (Fig. 1) and second (Fig. 2) PTN-based architectures are described hereafter.

According to Table 4, and with the exception of the NETtalk corpus, Slearp generally performed best among the three baselines (i.e., 1_Slearp, 2_Phon. and 3_DIRECTL+) in which the conventional g-p sequences were used. For instance, in terms of the WAcc, Slearp achieved 95.65%, 73.12%, and 73.55% for the Brulex, CMUdict and CMUdict_noisy corpora, respectively.

Surprisingly, when using reversed g-p sequences rather than conventional sequences, there was a slight improvement (0.4 ~ 1% for 4_Slearp.reverse and 0.2 ~ 0.5% for 5_Phon.reverse), with the exception of the DIRECTL+ models (i.e., 6_DIRECTL+.reverse) and the Brulex corpus.

When the three models based on the selected methods (viz., SSMCW-, WFST- and MIRA-based methods) were combined, the evaluation results in Table 4 further reveal that our first proposed PTN-based architecture can improve the performance of G2P conversion. PTN(4+5+6), the model with reversed g-p sequences, typically outperformed PTN(1+2+3), the same model but with conventional g-p sequences. Owing to the fact that reversed g-p sequences allow each single model to train an additional

[†]http://www.itl.nist.gov/iad/mig/tools/

Table 6 Percentage of input words where one model ($Model_A$) provides the correct phoneme-sequence hypotheses while another model ($Model_B$) provides an incorrect-sequence hypotheses. The results in this table are based on Fig. 1. When comparing the result of two models trained using the same method, the result in bold font indicates the model with higher percentage of correct phoneme-sequence hypotheses. For example, in the result for the NETtalk corpus, one cell [$Model_A$(Slearp.reverse), $Model_B$(Slearp)] has a higher percentage than its comparative cell [$Model_A$(Slearp), $Model_B$(Slearp.reverse)].

$Model_A$ = correct and $Model_B$ = incorrect		$Model_B$						
		1_Slearp	2_Phon.	3_DirecTL+	4_Slearp .reverse	5_Phon .reverse	6_DirecTL+ .reverse	
$Model_A$	1_Slearp	0	**9.03%**	6.75%	4.25%	**8.66%**	**6.99%**	NETtalk
	2_Phon.	6.55%	0	7.75%	6.37%	2.69%	8.00%	
	3_DirecTL+	**7.05%**	**10.52%**	0	6.65%	**10.11%**	**3.02%**	
	4_Slearp.reverse	**5.20%**	**9.79%**	**7.31%**	0	**9.46%**	**7.62%**	
	5_Phon.reverse	6.70%	**3.21%**	7.85%	6.55%	0	8.03%	
	6_DirecTL+.reverse	6.88%	**10.37%**	2.62%	6.57%	**9.89%**	0	
$Model_A$	1_Slearp	0	**2.68%**	**4.35%**	**0.73%**	**2.67%**	**4.41%**	Brulex
	2_Phon.	1.52%	0	**4.53%**	1.52%	**0.84%**	**14.54%**	
	3_DirecTL+	1.33%	2.68%	0	1.42%	2.71%	**0.56%**	
	4_Slearp.reverse	0.66%	**2.61%**	**4.36%**	0	**2.61%**	**4.42%**	
	5_Phon.reverse	1.45%	0.78%	**4.50%**	1.46%	0	**4.53%**	
	6_DirecTL+.reverse	1.35%	2.64%	0.51%	1.43%	2.69%	0	
$Model_A$	1_Slearp	0	**6.74%**	**8.84%**	4.35%	**6.60%**	**8.91%**	CMUdict
	2_Phon.	6.00%	0	**9.23%**	5.57%	2.09%	**9.30%**	
	3_DirecTL+	5.64%	6.77%	0	5.03%	6.64%	**2.38%**	
	4_Slearp.reverse	**5.14%**	**7.10%**	**9.03%**	0	**6.95%**	**9.12%**	
	5_Phon.reverse	6.01%	**2.23%**	**9.24%**	5.56%	0	**9.31%**	
	6_DirecTL+.reverse	5.66%	6.79%	2.33%	5.09%	6.66%	0	
$Model_A$	1_Slearp	0	**6.83%**	**9.89%**	4.32%	**6.54%**	**10.07%**	CMUdict_noisy
	2_Phon.	5.99%	0	**10.57%**	5.67%	2.53%	**10.54%**	
	3_DirecTL+	5.38%	6.88%	0	5.23%	6.76%	**2.26%**	
	4_Slearp.reverse	**4.73%**	**6.92%**	**10.17%**	0	**6.86%**	**10.34%**	
	5_Phon.reverse	6.10%	**2.93%**	**10.84%**	6.00%	0	**10.84%**	
	6_DirecTL+.reverse	5.44%	6.75%	2.15%	5.29%	6.66%	0	

and superior model, the number of models and phoneme-sequence hypotheses for PTN-based G2P conversion doubles. Thus, the entire model performance improves. For example, PTN(1+...+6) improved the WAcc of the best baseline models for NETtalk, Brulex, CMUdict, and CMUdict_noisy from 71.93% to 73.45%, 95.65% to 95.98%, 73.91% to 75.17% and 73.96% to 75.30%, respectively.

As explained in Sect. 4, the compact PTN-based architecture for G2P conversion has been proposed in order to minimize the risk from combining inaccurate and accurate methods. Because the size of the training data increases after using GGRs, and despite using the same representation of the grapheme sequence, the results from both (B-Slearp.GGR_0 versus A-Slearp) and (E-Slearp.GGR_0.reverse versus D-Slearp.reverse) in Table 5 demonstrate another method for increasing the performance of the baseline models other than the use of the reversed g-p sequence. By applying both techniques—GGRs and reversed g-p sequences—it is sufficient to use only the most accurate method (e.g., the SSMCW-based method in the Slearp toolkit) when implementing as many models as needed. After merging the hypotheses generated from all of those models with respect to the second proposed architecture (in Fig. 2), the results from the compactPTN(A+...+F), evaluated using the CMUdict_noisy corpus, show even more improvement in terms of the PAcc and WAcc.

6. Discussion

The results in Tables 4 and 5 demonstrate that there are two ways to improve the performance of each separated model, namely GGRs and reversed g-p sequences.

The previous evaluation results in Table 4 show that models using reversed g-p sequences generally outperformed those using conventional g-p sequences. After analyzing the data, we believe that some conventional sequences and their corresponding reversed g-p sequences were aligned differently owing to differing representations. Hence, we can assume that using the reversed g-p sequences provides better-aligned data for G2P conversion.

In order to appreciate the quality and helpfulness of the phoneme-sequence hypotheses involved in generating the PTN, we conducted an analysis of the sequences, inspired by McNemar's test theory. By calculating the percentage of words for which their corresponding phoneme sequences could be correctly established by one model (noted as $Model_A$) but not another (noted as $Model_B$), we can observe that the comparing results between any two different models in Table 6 are bigger than zero percentage for all the corpora. This means that when one model generates an incorrect phoneme sequence, other models can generate the correct sequence. In addition, by comparing two

Table 7 Percentage of words measured from the OOV dataset for different corpora. This measurement is needed to analyze the correctness and incorrectness between the input sequence hypotheses and the output sequence of the PTN. Here, the results belong to PTN(1+...+6) and compactPTN(A+...+F). *The second-row results in bold font are misjudged words. "Could be correct" refers to the result obtained on condition that the voting method could perfectly select the best phoneme-sequence from the generated PTN.*

A set of conditions			Percentage of words measured from the OOV dataset (%)				
Phoneme-sequence hypotheses (1, 2, ..., 6) or (A, B, ..., F) as inputs		Output sequence of the PTN	PTN(1+...+6)				compactPTN(A+...+F)
Status	Number of sequences	Status	NETtalk	Brulex	CMUdict	CMUdict_noisy	CMUdict_noisy
Correct / Incorrect	Some	Correct	19.15%	7.84%	17.83%	18.85%	9.70%
Correct / Incorrect	**Some**	**Incorrect**	**10.76%**	**1.99%**	**8.94%**	**9.42%**	**6.44%**
Correct	All	Correct	53.01%	87.33%	57.32%	56.43%	65.86%
Correct	All	Incorrect	0%	0%	0%	0%	0%
Incorrect	All	Correct	0.01%	0%	0.03%	0.06%	0%
Incorrect	All	Incorrect	17.08%	2.84%	15.88%	15.25%	18.01%
Correct/Incorrect	Some	*Could be correct*	29.91%	9.83%	26.77%	28.27%	16.14%
Incorrect	All	*Could be correct*	1.14%	0.08%	1.23%	1.00%	0.88%

models, especially models using the same method but with a different representation of the grapheme sequence (i.e., the conventional and reversed g-p sequences), we can assume that one model (or an accurate model) will not provide all of the correct results that were provided by another model (or an inaccurate model). This is because it is still likely that one model will generate the correct phoneme-sequence hypothesis, even when another cannot. For instance, a comparison between the Slearp.reverse and Slearp models using the NETtalk dataset shows that 5.20% of the words correctly phoneticized with the Slearp.reverse model were incorrectly phoneticized by Slearp, but only 4.25% the other way around (i.e., correctly phoneticized by Slearp, but not by Slearp.reverse). This evidence strongly reinforce the point that combining multiple models for G2P conversion is more effective than using any single model.

On the other hand, we used the eight conditions in Table 7 to analyze the relations in terms of correctness and incorrectness between the phoneme-sequence hypotheses (hyp.) generated from various source models and the output of the PTN-based model. These eight conditions are listed as follows:

- Some hyp. are correct → Output of PTN is correct
- Some hyp. are correct → Output of PTN is incorrect
- All hyp. are correct → Output of PTN is correct
- All hyp. are correct → Output of PTN is incorrect
- All hyp. are incorrect → Output of PTN is correct
- All hyp. are incorrect → Output of PTN is incorrect
- Some hyp. are correct → Output of PTN is "Could be correct"
- All hyp. are incorrect → Output of PTN is "Could be correct"

Results based on the second condition (i.e. the second row in Table 7) indicate that 10.76%, 1.99%, 8.94%, and 9.42% of the OOV words in NETtalk, Brulex, CMUdict, and CMUdict_noisy, respectively, were misjudged when using the first proposed PTN-based architecture. Moreover, the misjudged results from CMUdict_noisy were reduced to 6.44% when using the second proposed architecture. This shows that the proposed architectures can nevertheless improve the model performance when selecting a better technique for determining the best phoneme sequence from the PTN sequence.

Table 8 Example showing how a PTN-based G2P conversion can establish a correct output phoneme sequence even when all of the sequence hypotheses are incorrect.

	"BERENDS"							
Reference:	→ /B	EH	R	EH	N	D	Z/	
1_Slearp:	B	EH	R	AH	N	D	Z	Aligned sequence hypotheses
2_Phon.:	B	EH	R	EH	N		Z	
3_DirecTL+:	B		ER	EH	N	D	Z	
4_Slearp.reverse:	B	EH	R	AH	N	D	Z	
5_Phon.reverse:	B	EH	R	EH	N		Z	
6_DirecTL+.reverse:	B	EH	R	EH	N		Z	
PTN sequence:	B	{EH, @}	{R, ER}	{EH, AH}	N	{D, @}	Z	
	↓	↓	↓	↓	↓	↓	↓	
Voting(Output):	B	EH	R	EH	N	D	Z	

Even when all of the phoneme sequence hypotheses are incorrect, the PTN-based G2P conversion is still able to select the best phoneme candidate from each sequence (e.g., 0.01% for NETtalk, 0% for Brulex, 0.03% for CMUdict, and 0.06% for CMUdict_noisy). The example in Table 8 demonstrates that the PTN-based model can produce a correct output phoneme sequence for the word "BERENDS" even when all of the generated sequence hypotheses are incorrect. By supposing that the voting method could perfectly select the best output phoneme sequence from the generated PTN, the last row of Table 7 shows that the previous results could be improved to 1.14%, 0.08%, 1.23%, and 1.00% for NETtalk, Brulex, CMUdict CMUdict_noisy, respectively; in addition, if we also counted the cases that at least one correct phoneme-sequence hypothesis is used in the PTN generation, then both Tables 4 and 7 show that the performance of the PTN-based G2P conversion would be highly improved from 73.45% to 84.06% (1.14% + 29.91% + 53.01%) for NETtalk, from 95.98% to 97.24% (0.08% + 9.83% + 87.33%) for Brulex, from 75.17% to 85.32% (1.23% + 26.77% + 57.32%) for CMUdict, and from 75.30% to 85.70% (1% + 28.27% + 56.43%)

for CMUdict_noisy. These large improvements give us hope for the future challenge, which means that the voting method in our proposed PTN-based architectures for G2P conversion need to be improved.

The evaluation results for the compact version of the proposed PTN-based G2P conversion in Table 5 demonstrate that the novel use of reversed g-p sequences and GGRs improves PTN-based G2P conversions, even when only a single method is used. By comparing the evaluation results provided by the PTN-based architecture and its compact version, the results using the CMUdict_noisy corpus in Table 7 show that 18.85% of the correct words while using the first architecture, but only 9.70% of correct words while using the second architecture, has to take risk in the voting process. Thus, our compact PTN-based G2P conversion effectively minimizes risk in the voting process from combining inaccurate models with accurate ones. Furthermore, many different PTN-based architectures will be proposed to address challenges to G2P conversion in the future.

7. Conclusion and Future Work

In this paper, we showed that the proposed PTN-based G2P conversion is a novel and effective method for improving the quality of phoneme prediction for OOV words. The proposal combines different approaches to phoneme prediction in order to address the various problems encountered by G2P conversion. It also provides significant and consistently improved results compared to models based on a single approach. The novel use of reversed g-p sequences and GGRs in this paper can make complementary models that allow to generate new hypotheses so that ensemble of them has considerable gain for the PTN-based G2P conversion model, and it also can minimize the risk associated with combining accurate and inaccurate models. Moreover, we demonstrated that the representation of both graphemic and phonemic information plays an important role in improving model performance.

In future work, we plan to create new and effective GGRs to further improve our proposed approach, enabling a trained model to generate more accurately output phoneme-sequence hypotheses, such that only two models (using conventional and reversed g-p sequences) will be sufficient for our PTN-based G2P conversion. Moreover, the hamming distance, calculated from the articulatory features of phonemes [20], shall be used for the DP alignment process in the ROVER system. Inspired by the Long Short-Term Memory Recurrent Neural Network-based G2P conversion [21], we shall attempt to challenge our approach at the voting level with the use of a finite state transducer and a joint n-gram model, rather than relying on the simplistic voting method available in the ROVER system.

Acknowledgments

This work is supported by a Grant-in-Aid for Scientific Research (B) 25280128 2013 from MEXT, Japan.

References

[1] G. Miller, Language and Speech, p.49, W.H. Freeman and Company, San Francisco, 1981.
[2] K.U. Ogbureke, C. Peter, and B.C. Julie, "Hidden Markov Models with Context-Sensitive Observations for Grapheme-to-Phoneme Conversion," in Proc. of Interspeech, pp.1105–1108, Japan, 2010.
[3] H. Che, J. Tao, and S. Pan, "Letter-To-Sound Conversion using Coupled Hidden Markov Models for Lexicon Compression," in Oriental COCOSDA, Macau, pp.141–144, 2012.
[4] S. Kheang, K. Katsurada, Y. Iribe, and T. Nitta, "Solving the Phoneme Conflict in Grapheme-to-Phoneme Conversion Using a Two-Stage Neural Network-based Approach," IEICE Transactions on Information and Systems, vol.E97-D, no.4, pp.901–910, 2014.
[5] M. Bisani and H. Ney, "Joint-Sequence Models for Grapheme-to-Phoneme Conversion," Speech Communication, vol.50, no.5, pp.434–451, 2008.
[6] S. Jiampojamarn, A. Bhargava, Q. Dou, K. Dwyer, and G. Kondrak, "DirecTL: a Language Independent Approach to Transliteration," in Proc. of ACL-IJCNLP Nammed Entities Workshop, pp.28–31, 2009.
[7] S. Jiampojamarn, K. Dwyer, S. Bergsma, A. Bhargava, Q. Dou, M.Y. Kim, and G. Kondrak, "Transliteration generation and mining with limited training resources," in Proc. of the Named Entities Workshop (NEWS), pp.39–47, Sweden, 2010.
[8] J.R. Novak, N. Minematsu, and K. Hirose, "Failure transitions for Joint n-gram Models and G2P Conversion," in Interspeech, pp.1821–1825, 2013.
[9] K. Kubo, S. Sakti, G. Neubig, T. Toda, and S. Nakamura, "Grapheme-to-Phoneme Conversion based on Adaptive Regularization of Weight Vectors," in Proc. of Interspeech, pp.1946–1950, 2013.
[10] K. Kubo, S. Sakti, G. Neubig, T. Toda, and S. Nakamura, "Narrow Adaptive Regularization of Weights for Grapheme-to-Phoneme Conversion," in Proc. of ICASSP, pp.2589–2593, 2014.
[11] K. Kubo, S. Sakti, G. Neubig, T. Toda, and S. Nakamura, "Structured Soft Margin Confidence Weighted Learning for Grapheme-to-Phoneme Conversion," in Proc. of Interspeech, Singapore, pp.1263–1267, Sept. 2014.
[12] S. Kheang, K. Katsurada, Y. Iribe, and T. Nitta, "New Grapheme Generation Rules for Two-Stage Model-based Grapheme-to-Phoneme Conversion," Journal of ICT Research and Applications, vol.8, no.2, pp.157–174, 2014.
[13] J.G. Fiscus, "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, 1997.
[14] T. Schlippe, W. Quaschningk, and T. Schultz, "Combining grapheme-to-phoneme converter outputs for enhanced pronunciation generation in low-resource scenarios," in Proc. of SLTU, pp.139–145, St. Petersburg, Russia, May 2014.
[15] S. Kheang, K. Katsurada, Y. Iribe, and T. Nitta, "Model Prioritization Voting Schemes for Phoneme Transition Network-based Grapheme-to-Phoneme Conversion," in Proc. of CIST'15, pp.100-1–100-7, Canada, 2015.
[16] I. Sutskever, O. Vinyals, and Q.V. Lee, "Sequence to Sequence Learning with Neural Networks," in Proc. of Neural Information Processing Systems (NIPS), pp.3104–3112, Canada, July 2010.
[17] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," in Proc. of ICML, pp.369–376, 2006.
[18] S. Jiampojamarn, G. Kondrak, and T. Sherif, "Applying Many-to-Many Alignments and Hidden Markov Models to Letter-to-Phoneme Conversion," in Proc. of NAACL HLT, pp.372–379, New York, 2007.

[19] K. Kubo, H. Kawanami, H. Saruwatari, and K. Shikano, "Unconstrained Many-to-Many Alignment for Automatic Pronunciation Annotation," in Proc. of APSIPA ASC, pp.1–4, 2011.
[20] Y. Iribe, T. Mori, K. Katsurada, and T. Nitta, "Pronunciation Instruction using CG Animation based on Articulatory Feature," in Proc. of ICCE2010, pp.501–508, 2010.
[21] K. Rao, F. Peng, H. Sak, and F. Beaufays, "Grapheme-to-Phoneme Conversion Using Long Short-Term Memory Recurrent Neural Networks," in Proc. of ICCASP, pp.4225–4229, 2015.

Seng Kheang received a B. Eng. degree in Computer Science from the Institute of Technology of Cambodia, Cambodia. In 2011, he obtained an M. Eng. degree from the Toyohashi University of Technology (TUT), Japan. Since October 2012, he has been a doctoral student at the Department of Computer Science and Engineering, Toyohashi University of Technology. His research interests include text-to-phoneme conversion for speech synthesis, natural language processing, and spoken term detection.

Kouichi Katsurada received a Ph. D. degree from Osaka University in 2000. He has been with the Toyohashi University of Technology since 2000. He is currently an associate professor at the Center for International Relations and the Department of Computer Science and Engineering at Toyohashi University of Technology. His research interests include multimodal interaction, facial image processing, and spoken term detection. He is a member of the IEEE, ISCA, IPSJ, JSAI, and ASJ.

Yurie Iribe received M. S. and Ph. D. degrees from the Graduate School of Human Informatics of Nagoya University, Japan. Until 2013, she worked as an Associate Professor in the Information and Media Center at Toyohashi University of Technology. She is currently an Associate Professor in the Information Science Department at Aichi Prefectural University, Japan. Her recent research interests include education support and speech recognition. She is a member of the ISCA, IPSJ, JSAI, and ASJ.

Tsuneo Nitta received a Ph. D. degree from Tohoku University, Japan. He worked at the R&D Center and Multimedia Eng. Lab. of Toshiba Corp. from 1970 to 1998. He subsequently joined the Toyohashi University of Technology as a Professor in the Graduate School of Engineering. In 2012, he became a TUT Professor Emeritus and a visiting Professor, both at TUT and Waseda University. His current research interests include speech recognition, speech synthesis, and multimodal interaction. He is an IPSJ Fellow and a member of the IEEE, JSAI, and ASJ.