MULTILINGUAL GRAPHEME-TO-PHONEME CONVERSION WITH BYTE REPRESENTATION

Mingzhi Yu [1], *Hieu Duy Nguyen* [2], *Alex Sokolov* [2], *Jack Lepird* [2], *Kanthashree Mysore Sathyendra* [2], *Samridhi Choudhary* [2], *Athanasios Mouchtaris* [2], *and Siegfried Kunzmann* [2]
[1]University of Pittsburgh
[2]Amazon.com Inc., USA

ABSTRACT

Grapheme-to-phoneme (G2P) models convert a written word into its corresponding pronunciation and are essential components in automatic-speech-recognition and text-to-speech systems. Recently, the use of neural encoder-decoder architectures has substantially improved G2P accuracy for mono- and multi-lingual cases. However, most multilingual G2P studies focus on sets of languages that share similar graphemes, such as European languages. Multilingual G2P for languages from different writing systems, e.g. European and East Asian, remains an understudied area. In this work, we propose a multilingual G2P model with byte-level input representation to accommodate different grapheme systems, along with an attention-based Transformer architecture. We evaluate the performance of both character-level and byte-level G2P using data from multiple European and East Asian locales. Models using byte representation yield 16.2%– 50.2% relative word error rate improvement over character-based counterparts for mono- and multi-lingual use cases. In addition, byte-level models are 15.0%–20.1% smaller in size. Our results show that byte is an efficient representation for multilingual G2P with languages having large grapheme vocabularies.

Index Terms— Grapheme-to-phoneme (G2P), multilingual, end-to-end models, byte representation, pronunciation generation

1. INTRODUCTION

Grapheme-to-phoneme (G2P) is a process that converts a sequence of graphemes into a corresponding sequence of phonemes. Typically, a G2P model is responsible of generating the pronunciation for out-of-vocabulary (OOV) words given their written forms. An obvious application of G2P is traditional Automatic Speech Recognition (ASR) system, which uses decoding graph that might leverage on word lexicons. Another example is Text-to-Speech (TTS) for which an OOV word's pronunciation is required to generate responses.

Previous studies have proposed rule-based models that map each grapheme into its corresponding phoneme [1, 2]. Such approaches require handcrafted rules written by experts with knowledge of the target language. Alternatively, researchers have also considered statistical models based on conditional probability, such as joint n-gram methods [3, 4]. This approach takes into account the input history, i.e. previous graphemes, so that it is fully contextualized and data-driven. The main drawback is the requirement for explicit alignment between graphemes and phonemes.

Recently, architectures based on Connectionist Temporal Classification (CTC) and encoder-decoder have been introduced to allow sequence-level G2P conversion without alignment [5, 6]. In particular, considerable performance improvements were reported for neural networks using sequence-to-sequence architectures [7, 8]. The intuition is that G2P is similar to a translation system, in which the source and target languages are the graphemes and phonemes, respectively. Long Short Term Memory (LSTM), which takes into account the full history of the graphemes, has been applied for this task. In this work, we instead adopt the Transformer [9] as the neural architecture for our G2P model. The Transformer is an attention-based neural model that outperforms recurrent/LSTM counterparts for various machine learning applications, notably machine translation. In [10], Transformer-based G2P is also observed to achieve better performance than recurrent/LSTM encoder-decoder. Another advantage of the Transformer-based model is the ability for parallel training, leading to significantly less learning time.

Monolingual G2P, in which the model predicts the phonemes for graphemes from one language only, has been studied extensively in the literature [3, 11, 12]. Recent advances of deep learning, described above, have enabled more investigations on multilingual G2P, which incorporates multiple languages into one model [1, 2, 6]. The benefits of a multilingual model are two-fold. Firstly, it enables the implementation of multilingual end-to-end ASR and TTS, of which G2P is an integral part. Secondly, it is capable of leveraging the shared knowledge between languages under the same pronunciation system. For example, the phonemes of an unknown English word could be approximately estimated based on the word's pronunciation in a German dataset. This mainly benefits low-resource languages with limited/small data [6, 13].

A majority of research studies has been focused on mul-

This work was conducted during Mingzhi's internship at Amazon.com Inc., Pittsburgh, PA, USA

tilingual G2P for European languages, which have a small number of (mostly similar) graphemes. There are much fewer studies on multilingual model for *different* language systems, such as European and East Asian ones. One of the challenges for constructing such neural multilingual G2P is the model vocabulary size. Chinese, Japanese, and Korean (so-called CJK group) have tens of thousands of distinct graphemes that vary in writing and pronunciation [14]. Due to a large number of graphemes, the model faces label sparsity issue. More importantly, the sheer size of CJK vocabulary dominates that of European languages, reducing performance for both groups.

To address the aforementioned issues, we introduce a novel input representation for multilingual neural G2P which leverages on bytes. The byte-level representation has been explored in other multilingual tasks such as part-of-speech tagging, name-entity-recognition [15], and speech-to-speech conversion [16]. An attractive feature for byte representation is the small vocabulary size compared to that of conventional character representation. Using Unicode encoding [14], each character can be converted into a unique sequence of, either one or multiple, bytes. For example, most of English graphemes can be represented by one byte, whereas those of CJK often need three bytes. Nevertheless, the cardinality of the byte set is 256. A multilingual neural G2P using bytes thus has a vocabulary size constrained to 256, regardless of the incorporated languages.

In this paper, we propose a multilingual neural G2P model using bytes as the input representation, combined with a Transformer-based architecture. We consider three European languages, i.e. English, French, Spanish, and two East Asian ones, namely Chinese and Japanese, that have extensive numbers of graphemes. To the best of our knowledge, this study is one of the first to investigate different language systems for multilingual G2P using a neural network architecture. Comparing with character-level models, we observe a significant performance improvement for byte-level models under both mono- and multi-lingual G2P tasks. Furthermore, as described above, a model with byte representation has a smaller vocabulary size and model footprint.

2. PROPOSED APPROACH

2.1. Byte-Level Representation

The byte representation of a character is generated based on its UTF-8 encoding [14]. For English, French and Spanish, byte and character representations result in mostly similar input vocabularies, as one byte is able to represent one character in the lexicons. However, East Asian languages like Chinese and Japanese require multiple bytes per character. Please refer to Table 1 for examples. Using bytes as the input representation for G2P models is beneficial mainly from two points.

- **Smaller vocabulary size:** European languages are phonographic, in which each grapheme set has a small cardinality. For example, English and French have 26 and 46 distinct characters, respectively. However, some Asian languages, especially CJK, follow logographic writing systems. In this case, each character has a unique pronunciation and meaning. The number of characters in each of CJK languages is significantly larger than all European alphabets. For example, Chinese has over 50000 characters, and even the number of frequently occurring Chinese characters exceeds 3000 [14]. East Asian languages like CJK, therefore, result in sparser datasets and more training parameters for G2P. Under multilingual character-based G2P systems for European and CJK languages, the problem is especially severe as 95–99% of the model vocabulary will come from CJK, which furthermore have much less training samples than European languages. Using byte representation efficiently solves this problem, as each character is decomposed into a series of bytes. The byte-level vocabulary cardinality is constrained to be equal to or smaller than 256. Byte representation thus reduces vocabulary sparsity which improves the model accuracy.

Table 1: Byte representation for English and Chinese words

h, e, l, l, o	0x68, 0x65, 0x6c, 0x6c, 0x6f
你, 好	0xe4 0xbd 0xa0, 0xe5 0xa5 0xbd

- **Knowledge sharing:** The byte representation for each character is defined by groups of experts under the Unicode consortium. It is a rigorous process with the purpose of creating a concise and versatile character encoding standard [14]. Due to experts' assessment, the byte representation for a character contains certain intuitions on how the character is written or pronounced. For CJK, characters having similar written forms or pronunciations are indexed close to each others in Unicode, e.g. Japanese characters with the same vowel or Chinese characters with an identical radical. To illustrate this point, consider the Japanese Hiragana 3072 (ひ), 3073(び), 3074(ぴ) under UTF-8 [14]. They contain the same vowel "i" with consonants "h","b", and"p", respectively, and their byte representations are [0x-e3, 0x-81, 0x-b2/b3/b4]. Another example is the Chinese Han characters with indexes from 4F10 (伐) to 4F19(伙). They share the radical "人" and the first two bytes [0x-e4, 0x-bc, ...] in their encodings. This property of the Unicode might improve the knowledge sharing between characters, especially CJK ones, that helps the neural embedding to capture more information. In contrast, there is no such knowledge sharing for character representation.

2.2. Language Index

Note that European languages share most of the alphabets, as do Japanese Kanji and Chinese. The same grapheme, there-

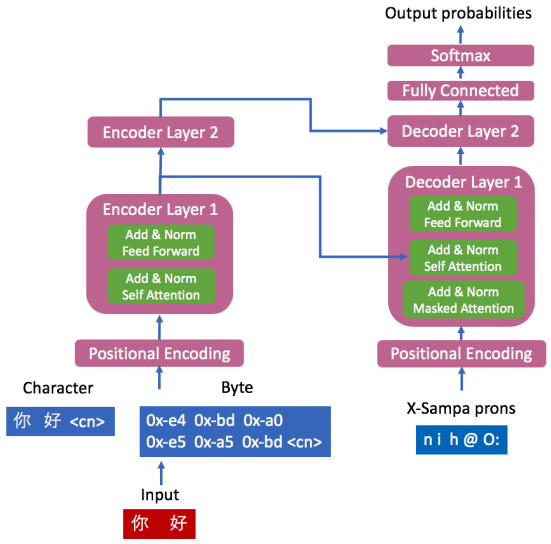

Fig. 1: Neural G2P model with Transformer architecture and character or byte input representation. Here, Encoder and Decoder Layer 2 have the same structures as those in Layer 1, respectively.

fore, could appear in various languages with different corresponding phonemes. For example, the grapheme "y" is pronounced as "i g R E k" and "w aI" in French and English, respectively. In order to address the variation of phonemes in different languages, we concatenate a language index to the end of each input grapheme sequence. Please refer to Fig. 1 for an illustration.

2.3. Model Architecture

We utilize a Transformer-based architecture with 2 layers for both encoder and decoder, following closely the original setup in [9]. Each of the self-attention mechanism contains 8 heads, and the last component inside each of the decoder layer has time step masked to prevent leaking future information to the decoder. Furthermore, the input positional embedding layer is trainable without tied weights [17] since the input and output tokens are very different. Each model is trained with a dropout of 0.15 and cross-entropy loss function. Fig. 1 shows our model architecture. Each input is a sequence of graphemes, represented by either characters or bytes, followed by a language index. The reference for a word is its pronunciation annotated under X-Sampa standard. In this work, we apply a train/dev/test data split of 70%/15%/15%.

2.4. Evaluation Metrics

To evaluate the model performance, we use Phoneme Error Rate (PER) and Word Error Rate (WER), defined in Eqs. (1) and (2), respectively. Here, ref and hyp are the reference and hypothesis predicted by the G2P model. $MED(ref, hyp)$ represents the minimum edit (Levenshtein) distance between

Table 2: Performance of monolingual models, in which there are 320 hidden units in each layer of the Transformer

	Character-Level			Byte-Level		
	PER(%)	WER(%)	# params	PER(%)	WER(%)	# params
English	5.25	19.89	3.4M	**5.25**	**19.89**	3.4M
French	6.24	23.67	3.4M	**5.77**	**22.05**	3.4M
Spanish	1.31	6.79	3.4M	**1.22**	**6.53**	3.4M
Japanese	5.60	12.46	4.0M	**5.52**	**11.87**	3.4M
Chinese	2.16	9.44	4.3M	**1.07**	**5.71**	3.4M

the reference and hypothesis. Finally, $|ref|$, $|E|$, and $|N|$ denote the length of reference, the number of wrongly predicted pronunciations, and the total number of reference tokens, in corresponding order. For each word, each G2P model only generates one best hypothesis. Since each word might have several pronunciations, the best hypothesis is compared with all possible correct references. The reference that has the shortest minimum edit distance to the hypothesis is selected as the correct ground-truth for calculating PER/WER.

$$PER = \frac{MED(ref, hyp)}{|ref|} \tag{1}$$

$$WER = \frac{|E|}{|N|} \tag{2}$$

3. DATASET

Here we utilize a human-transcribed subset from Amazon that consists of 5 languages: English, French, Spanish, Japanese, and Chinese. Lexicons are collected across different locales for the same language. For example, en-US represents English spoken in the United States. Each token is transcribed into its pronunciation using X-Sampa phone set. Tokens with rare or very long graphemes/phonemes are considered as noise, and are removed according to predefined thresholds. In particular, a grapheme or phoneme will be filtered if it appears less than 50 times for European languages and 10 times for East Asian counterparts, respectively. Furthermore, a token contains more than 30 graphemes or phonemes will be removed. After filtering, there are approximately 10 million graphemes-phonemes pairs, of which languages with highest (English) and lowest (Japanese) resource contribute 30% and 8% to the whole set, respectively. Note that each word may have multiple pronunciations even in the same language. For example, the word "dog" may be pronounced as "d A g" and "d @U g" in English. We include all variations and use all of them as the references in the evaluation (see Section 2.4). Similar to [6], we leverage on block sampling for all languages so that consecutive words under alphabetical sorting are allocated into the same training, validation, or testing partition.

Table 3: Comparisons of multilingual models with character and byte input representation. Here Model A and B are character-based G2P with 4.7M and 3.5M parameters, respectively, whereas Model C is a 3.5M-parameter, byte-level counterpart. In each layer of the encoder and decoder, the numbers of hidden units in Model A, B, and C are 320, 288, and 320, correspondingly.

	Character – 4.7M – Model (A)		Character – 3.5M – Model (B)		Byte – 3.5M – Model (C)	
	PER(%)	WER(%)	PER(%)	WER(%)	PER(%)	WER(%)
English	5.99	23.45	6.23	23.66	**5.89**	**22.46**
French	6.15	**22.68**	6.62	23.35	**5.97**	22.83
Spanish	1.57	6.75	1.90	6.95	**1.18**	**6.38**
Japanese	5.82	15.63	6.19	16.34	**5.58**	**12.98**
Chinese	7.14	29.13	7.17	29.41	**1.34**	**6.04**
All	4.94	19.28	5.23	19.64	**4.03**	**16.14**

4. RESULTS

4.1. Monolingual Models

We first train and evaluate monolingual models for each language, which consists of multiple locales. Table 2 compares the performance of character-level and byte-level models in terms of PER and WER. Since each character in English is designated by exactly one byte, the byte-level representation is identical to the character counterpart, leading to similar error rates. Interestingly, French and Spanish G2P show some improvement when moving from character to byte-level input. This might be because their alphabets contain multi-byte characters, such as "ñ" and "à". The most noticeable improvements are observed for Chinese G2P with 50.2% WER and 39.4% PER relative reductions, respectively.

It is worth to note that byte representation also results into smaller model footprint. Specifically, Japanese and Chinese byte-level models have 15.0% and 20.1% less number of parameters than those of character-level. The primary reason, as discussed in Section 2.1, is the compact vocabulary space with a maximum cardinality of 256 symbols due to byte-level representation, given the UTF-8 encoding [14]. There is no significant difference in training time for character- and byte-level G2P models in Sockeye [18] when using Amazon Web Services (AWS) p3.16xlarge Graphics Processing Unit (GPU) instance. To summarize, the byte-level model already shows advantages under monolingual regime. The benefits of byte representation will be more transparent under multilingual G2P, described in the next section.

4.2. Multilingual Models

We trained multilingual G2P models using data from European and East Asian languages. Table 3 presents the results under different setups. Given the same 320 hidden units in each Transformer layer, byte-level Model C has a better performance compared to character-level Model A across all languages. Overall PER and WER decrease by 18.24% and 16.27% correspondingly. In particular, East Asian languages show more improvement than European counterparts, with Chinese and Japanese achieving a relative WER reduction of 79.2% and 16.9%, respectively. The difference in relative performance might be attributed to two reasons. Firstly, European languages already achieve very good performance via character-level G2Ps, making it a hard baseline to beat. Secondly, as described in Section 2.1, Chinese and Japanese languages have large vocabularies and thus benefit more from byte representation due to the reduced vocabulary size, as well as the knowledge sharing between characters under byte form. Note that Model A has more parameters than Model C. With the same number of hidden units, using bytes as the input representation helps reducing the model size by 25.53% relatively. For a fair evaluation, we also study another character-level G2P model (Model B) which has approximately equal number of parameters to Model C. We achieve that by decreasing the number of hidden units from 320 for Model A to 288 for Model B. As expected, the performance of Model B is worse than that of Model A and Model C, thus further justifying our approach.

5. CONCLUSION AND FUTURE WORKS

We propose a neural G2P model with byte-level input representation and Transformer-based architecture. Byte-level input representation restricts the model vocabulary, leading to a smaller footprint. It further enables the incorporation of multiple languages without increasing the complexity of the model. Using data from English, French, Spanish, Chinese, and Japanese lexicons, we construct mono- and multi-lingual neural G2P models at character- and byte-level. Our results show that byte-level models outperform character-level counterparts in terms of PER and WER under both mono- and multi-lingual cases. In the future, we intend to investigate how bytes-level representation reinforces knowledge sharing and improves model accuracy, and to evaluate byte-level G2P on public datasets when they are made available.

6. REFERENCES

[1] D. R. Mortensen, S. Dalmia, and P. Littell, "Epitran: Precision g2p for many languages," *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC)*, pp. 2710–2714, 2018.

[2] A. Deri and K. Knight, "Grapheme-to-phoneme models for (almost) any language," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, vol. 1: long papers, pp. 399–408, 2016.

[3] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion," *Speech Communication*, vol. 50, no. 5, pp. 434–451, 2008.

[4] J. R. Novak, N. Minematsu, and K. Hirose, "Wfst-based grapheme-to-phoneme conversion: Open source tools for alignment, model-building and decoding," *Proceedings of the 10th International Workshop on Finite State Methods and Natural Language Processing (FSMNLP)*, pp. 45–49, 2012.

[5] B. Milde, C. Schmidt, and J. Köhler, "Multi-task sequence-to-sequence models for grapheme-to-phoneme conversion.," *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 2536–2540, 2017.

[6] A. Sokolov, T. Rohlin, and A. Rastrow, "Neural machine translation for multilingual grapheme-to-phoneme conversion," *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2019.

[7] K. Rao, F. Peng, H. Sak, and F. Beaufays, "Grapheme-to-phoneme conversion using long short-term memory recurrent neural networks," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4225–4229, 2015.

[8] K. Yao and G. Zweig, "Sequence-to-sequence neural net models for grapheme-to-phoneme conversion," *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 3330–3334, 2015.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 5998–6008, 2017.

[10] H. Sun, X. Tan, J.-W. Gan, H. Liu, S. Zhao, T. Qin, and T.-Y. Liu, "Token-level ensemble distillation for grapheme-to-phoneme conversion," *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, 2019.

[11] P. Taylor, "Hidden markov models for grapheme to phoneme conversion," *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 1973–1976, 2005.

[12] R. S. Berndt, J. A. Reggia, and C. C. Mitchum, "Empirically derived probabilities for grapheme-to-phoneme correspondences in english," *Behavior Research Methods, Instruments, & Computers*, vol. 19, no. 1, pp. 1–9, 1987.

[13] B. Peters, J. Dehdari, and J. van Genabith, "Massively multilingual neural grapheme-to-phoneme conversion," *Proceedings of the 1st Workshop on Building Linguistically Generalizable NLP Systems*, pp. 19–26, 2017.

[14] The Unicode Consortium, "The unicode standard, version 12.0.0," `http://www.unicode.org/versions/Unicode12.0.0/`, 2019.

[15] D. Gillick, C. Brunk, O. Vinyals, and A. Subramanya, "Multilingual language processing from bytes," *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 1296–1306, 2016.

[16] B. Li, Y. Zhang, T. Sainath, Y. Wu, and W. Chan, "Bytes are all you need: End-to-end multilingual speech recognition and synthesis with bytes," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5621–5625, 2019.

[17] O. Press and L. Wolf, "Using the output embedding to improve language models," *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, vol. 2: short papers, pp. 157–163, 2017.

[18] F. Hieber, T. Domhan, M. Denkowski, D. Vilar, A. Sokolov, A. Clifton, and M. Post, "Sockeye: A toolkit for neural machine translation," *arXiv preprint arXiv:1712.05690*, 2017.