Neural Machine Translation for Multilingual Grapheme-to-Phoneme Conversion

Alex Sokolov, Tracy Rohlin, Ariya Rastrow

Amazon.com, Inc. USA

alexsoko,tracyroh,arastrow@amazon.com

Abstract

Grapheme-to-phoneme (G2P) models are a key component in Automatic Speech Recognition (ASR) systems, such as the ASR system in Alexa, as they are used to generate pronunciations for out-of-vocabulary words that do not exist in the pronunciation lexicons (mappings like "$e\ c\ h\ o$" → "$E\ k\ oU$").

Most G2P systems are monolingual and based on traditional joint-sequence based n-gram models [1, 2]. As an alternative, we present a single end-to-end trained neural G2P model that shares same encoder and decoder across multiple languages. This allows the model to utilize a combination of universal symbol inventories of Latin-like alphabets and cross-linguistically shared feature representations. Such model is especially useful in the scenarios of low resource languages and code switching/foreign words, where the pronunciations in one language need to be adapted to other locales or accents. We further experiment with word language distribution vector as an additional training target in order to improve system performance by helping the model decouple pronunciations across a variety of languages in the parameter space. We show 7.2% average improvement in phoneme error rate over low resource languages and no degradation over high resource ones compared to monolingual baselines.

Index Terms: grapheme-to-phoneme conversion, sequence-to-sequence model, multilingual machine translation, pronunciation generation.

1. Introduction

Alexa's ASR platform relies on a large hand-curated lexicon which is comprised of word-pronunciation pairs. This lexicon can never provide complete coverage over the vocabulary as it is often not worth the time or cost required to create these phonetic sequence mappings compared to how infrequently certain words occur during any given Alexa interaction.

Grapheme to Phoneme systems, on the other hand, can learn these mappings automatically with high accuracy and are responsible for transcribing any out-of-vocabulary (OOV) tokens into phonemic representations. These phonemic representations are an important component that lies between the language model and acoustic model of an ASR system. These OOV tokens often include rare and foreign words, the amount of which may vary depending on the language.

The challenge in designing the G2P system is to create a many-to-many mapping system that will learn not only the mapping between one grapheme and one phoneme, but also where one phoneme is represented by multiple graphemes (such as "$s\ h$" → "$ʃ$"). And for certain languages like English, these mappings can be inconsistent and ambiguous, especially in the case of names and foreign words.

Sequence to sequence (Seq2Seq) neural network models are one such way of learning the mappings between graphemes and phonemes where the input and output sequence can vary in length. Originally designed for machine translation, they have been applied on wide variety of problems, such as generative language models. More recent Seq2Seq models heavily incorporate attention mechanism [3] and residual learning, while other models use encoder-decoder architectures that are recurrent [4, 5, 6, 7, 8] or self-attention [9] based.

Recurrent seq2seq models pose a distinct advantage due to their ability to take in the input history when determining the output state, often outperforming n-gram models on classification tasks due to an n-grams heavy dependence on the previous n graphemes [10]. This means that Recurrent Neural Networks (RNN) are better suited for sequence problems where longer term context and soft input embeddings are important. Long-Short-Term Memory (LSTM) networks [11] are in turn better at handling longer sequences and can have more layer depth as they are less prone to diminishing and exploding gradients. Bi-directional LSTMs that consider both past and future contexts have become increasingly popular over RNNs or uni-directional LSTMs that only consider past contexts.

Seq2Seq models can be trained on multiple languages at once and used in multi-task and multi-modal learning scenarios. In context of G2P conversion, they allow joint learning of the alignment and translation of graphemes and phonemes in an end-to-end fashion. Therefore, they are a natural fit for our multilingual G2P task, especially since we can train it on large pronunciation lexicons with relatively short sequences lengths. Seq2Seq models have been found to perform better on these short sequences than compared to very long sequences [7]. However, until now neural G2P models have not shown superior results on their own [12], compared to traditional joint-sequence based n-gram models for G2P.

In this paper, we examine whether Seq2Seq LSTM models perform better compared to traditional joint-sequence based n-gram models, given the latest advancements in Seq2Seq modeling for single language pairs. In addition, we investigate whether we can build a single multilingual G2P model which may outperform individual models trained on single language lexicons by utilizing transfer learning thereby improving performance on under-resourced languages and foreign words from different locales. The goal of such a system is to have a single multilingual model that matches or improves over the results of monolingual models without the degradation in accuracy introduced by using multilingual dataset. In particular, the model must be able to distinguish between languages where the same grapheme is paired to different phones, such that the model does not learn a single pairing and apply this pairing to all instances of the grapheme, regardless of input language. Thus, we wish to avoid situations where a larger lexicon overwhelms a smaller lexicon and erroneously labels a particular grapheme. By using Seq2Seq LSTM models we achieve better PER and WER than low-resource monolingual models, while reducing the potential

influence larger-resource lexicons might have in a multilingual model.

2. Related work

Traditionally G2P is done with joint-sequence n-gram models [1, 2], [13, 14]. While these are still considered to be state of the art models best designed for single lexicons, they require explicit alignment information to be provided during training.

The Seq2Seq approach to G2P was first proposed in [12], but the work only showed improvement over state of the art results when using the same alignments in RNN slot tagger fashion. The alignments were generated by a separate HMM many-to-many alignment procedure [15], a weakness inherited from joint sequence models. [16] used connectionist temporal classification with output delays instead of Seq2Seq for joint alignment and translation which resulted in a significant improvement when compared to the Phonetisaurus [1] baseline implemented as finite-state transducer (FST).

A Seq2Seq bi-directional LSTM were used in [17] to perform G2P tasks on Persian, a particularly challenging task as often times vowels are not indicated in the languages orthography and are often dependent on their positions within the sentence. [1] likewise uses an RNNLM to perform N-best re-scoring on the G2P alignments produced by their Weighted Finite State Machine, beating state-of-the-art models using CMUDict and NetTalk data.

Most related to this work, [18] trained a Seq2Seq G2P model with stacked bi-directional LSTMs using attention and residual connections on two languages at once. [19] used a similar model to train a G2P model on hundreds of low resource languages, some of them in zero-shot fashion as described in [5]. The focus of this paper, however, is on achieving state of the art performance on a limited set of languages, both high and low resource, with a single multilingual model.

3. Dataset

Amazon has an extensive English ASR lexicon containing word-pronunciation pairs for En-US and En-UK locales, in addition to a large German ASR lexicon. Other data sources we used included CMUDict and the Wiktionary dataset [20] which includes lower-resource languages such as Czech, Finnish, Hungarian and Polish. The data was transcribed using the standard X-Sampa phone set and stripped of any syllabic or stress markers. Tokens that had rare grapheme or phone (less than 25 and 5 times, respectively) were considered to be noise and filtered out. The combined dataset contains 4 million grapheme-phoneme pairs from 18 languages (21.02% of which comes from En-US data alone), 84 unique graphemes and 117 phones (including 9 phones found in en-UK dataset but not in en-US, comprised primarily of vowels).

The majority of graphemes (90.8%) are unique to one lexicon, but still have multiple phonemic representations within each lexicon, as only 22.79% of graphemes had a single phonemic representation. Block sampling was used to ensure that consecutive words sorted in alphabetical order were placed in the same train, dev or test partition. This would not allow the models to "cheat" by seeing almost identical words during train and test time (i.e. a word "$dogs$" pronounced as "$d\ O\ g\ z$" for en-US and as "$d\ O\ k\ s$" for de-DE).

4. Proposed model

The core of our model (Figure 1) is an RNN encoder-decoder model with attention mechanism [4]. The model consists of an encoder which compresses each source grapheme in the input sequence into a fixed-length vector, and a decoder which generates a phoneme sequence as output, conditioned on the attention over the encoder hidden states. They are trained jointly to minimize cross-entropy on the training data. For both the encoder and decoder, we use long short-term memory units (LSTM) [11], which can process sequences of arbitrary length and use long histories efficiently. We use trainable character and phoneme embeddings as encoder and decoder inputs correspondingly.

The encoder is a stacked bidirectional LSTM, which processes the input in both forward and backward directions, as to represent both past and future dependencies at every time step. Forward and backward cells are stacked vertically and their outputs are concatenated at each layer.

The decoder is a language model conditioned on the past phoneme sequence and global attention mechanism [3]. Attention vectors are added to the decoder inputs which helps to facilitate information flow between the source and the target sequences. Instead of forcing the encoder to compress information about the whole input sequence in its last hidden state, a weighted sum over all encoder hidden states forms the context vector. At each decoding step, the decoder generates a softmax distribution over phonemes, until it outputs end-of-sentence symbol. N-best decoding is used to keep N different paths, while searching through the search space, to find more optimal solutions.

To generate pronunciation for the desired target language in a multilingual G2P scenario, we need to explicitly condition our model on the corresponding input parameter. Similar to [6] we use a system ID (language token like <en-US>, <en-UK>, <de-DE>, <fr-FR>, etc) as input embedding, but instead of appending this token to the input sequence on the encoder side, we concatenate it with the attention context vector and feed into the decoder's fully-connected layer just before the softmax. The two approaches have similar performance, but our implementation is motivated by the need to have one other input parameter representing the language distribution (language ID) - a vector of length equal to the number of languages, which is represented by a set of tokens that are mutually exclusive to the tokens in the vocabulary. The language distribution vector represents the extent to which a particular word belongs to every language and is pre-computed using the following formula:

$$p(id \mid l) = \frac{C(w)}{log\,|N|}, \quad \sum_{l} p(id \mid l) = 1,$$

where N refers to the lexicon size and $C(w)$ refers to the count of word occurrences found within a given language lexicon. We have also tried to smooth it by taking logarithm of word count and multi-hot binary vector representation, but that didn't have any significant impact on the results. The intuition behind the language ID vector is that there may be a correlation between the word's origin language and its pronunciation for any given locale. Since it is hard for the model to learn language identification only having a single word as an input, the language ID might be a useful signal to help the model distinguish words having multiple pronunciations in different languages and model their phone distributions accordingly. Since a very large number of words have several pronunciation alternatives even for the same language, we are doing n-best decoding
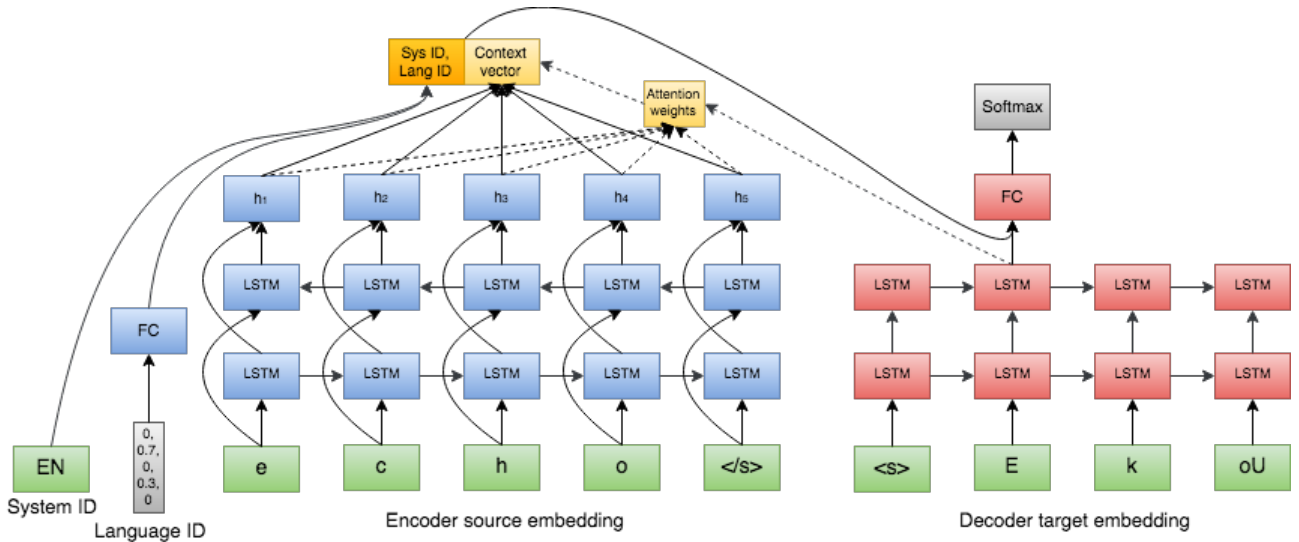
Figure 1: *Proposed encoder-decoder model architecture for a single decoding step. Language distribution (Language-ID) and language label (System-ID) are concatenated together with attention context vector and fed to the decoder fully connected layer just before softmax.*

during inference. We then select up to 3 best scoring hypothesis with average token posterior values above a threshold we optimized on the dev set (25% for 2-best and 18% for 3-best). This way, we increase the coverage of our output hypothesis without adding too much noise into it.

5. Evaluation and results

We used Phoneme Error Rate (PER) and Word Error Rate (WER) metrics on the target phoneme sequence to evaluate the G2P models. PER is defined as $PER = \frac{LD(p,p')}{|p|}$, where $LD(p,p')$ represents the minimum (Levenshtein) distance between the predicted and ground truth phoneme sequence, length of which is represented by $|p|$. WER is defined as $WER = \frac{|E|}{|N|}$, where $|E|$ represents the number of predicted pronunciations with one or more errors.

A baseline multilingual Seq2Seq model (System 0) which did not include language distribution and language label information performed very poorly across all the languages as expected and achieved average PER and WER of 10.09% and 47.37% , respectively. What was surprising, however is that System 2 also hasn't outperformed System 1 on any language, although it still performed much better than the baseline multilingual model, with a multilingual PER around 5.7% and WER around 38.8% for System 2. This might be due to the fact that language IDs are a function of the completeness of human labelled lexicons, sparsity of which might be introducing too much noise into the labels, especially for the languages for which we don't have a lot of annotations.

System 1 performed significantly better than the baseline model, with a 5.3% PER and 36.4% WER (table 1). This must be due to the fact that by feeding the language label into the model, the model is receiving a hint as to which pronunciation is mapped to a particular grapheme. This helps to delineate between graphemes that might occur in multiple dialects but routinely have different phoneme (in particular with vowels) qualities. For example, the vowel "ɒ" ("Q" in X-Sampa notation) is much more likely to show up in British English than American

English, as there are absolutely no instances of "ɒ" in the en-US lexicon, but over 8,000 instances in the en-UK lexicon.

Table 1: *PER and WER results of baseline monolingual models and multilingual model + system ID (System 1) on large (>300K words) and low-resource (<15K words) lexicons. Best model (System 1) also didn't show any significant degradation compared to monolingual models for the high resource languages. Numbers in bold indicate languages for which the multilingual model improved in error rates, despite the class imbalance problem and opportunity for larger lexicons to influence the transcription.*

High resource languages				
	Monolingual		Multilingual	
Language	PER	WER	PER	WER
En-US	0.127	0.525	0.129	0.528
En-UK	0.121	0.511	0.125	0.525
German	0.036	0.151	0.042	0.184
French	0.077	0.429	0.077	0.431
Dutch	0.023	0.121	0.028	0.135
Italian	0.003	0.027	0.005	0.046
Hindi	**0.131**	**0.584**	**0.114**	**0.545**
Low resource languages				# words
Czech	**0.081**	**0.489**	**0.071**	**0.439**	5238
Finnish	**0.038**	**0.322**	**0.034**	**0.288**	8500
Hungarian	0.014	0.143	0.015	0.155	10132
Polish	0.069	0.333	0.082	0.418	7249
Portuguese	**0.149**	**0.725**	**0.130**	**0.684**	6567
Spanish	**0.094**	**0.494**	**0.053**	**0.364**	4916

Certain languages also continue to perform better than others across all models. Italian, Dutch, and Hindi consistently have lower PER and WER in these models compared to the rest of the data set. It is important to note, however, that the Hindi dataset has graphemes transcribed into Latin-based characters and as such was likely phonetically transcribed. Still, the low PER and WER in these languages are likely due to a high one-

to-one correspondence between graphemes and phonemes. This is in contrast to languages like English that require the model to learn many-to-many mappings (such as "th" to "θ"). The many-to-many mapping is likely a factor in the high PER and WER for both dialects of English. In addition, certain languages in the dataset are more likely to use one particular grapheme for several different phonemes, leading to high PER and WER. Portuguese, for example, has several instances of one grapheme mapping to several phonemes, which is likely why this language consistently performs the worst out of any other language in the dataset.

In addition to these issues, certain datasets such as Portuguese and Polish likely suffered due to the mappings in these smaller datasets being out-weighed by larger lexicons like English and German. Additionally, the smaller datasets were often only comprised of Wiktionary phonetic transcriptions. [20] discusses how the Wiktionary lexicons are consistently more likely to perform worse on G2P tasks compared to the researchers own GlobalPhone dataset, except for German. Since the Spanish, Portuguese, Polish, Finnish, Esperanto and Czech lexicons are comprised entirely of Wiktionary data, it is likely the quality of the original human-annotated transcriptions posted on Wiktionary that are causing high error rates among these lexicons.
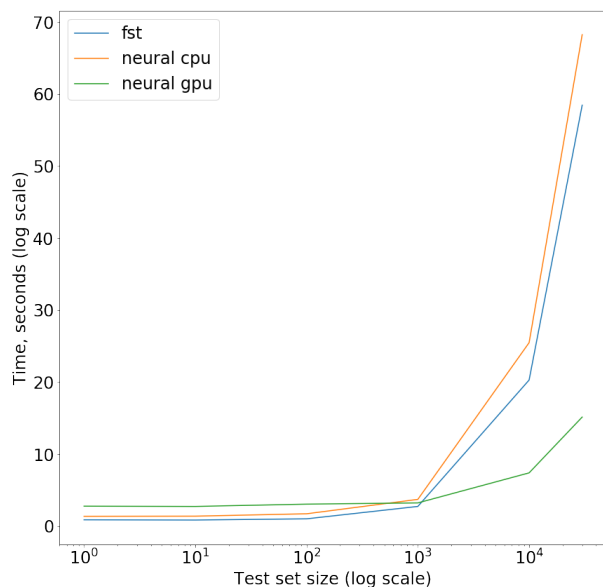

Figure 2: *Inference latency of FST and neural G2P models for varying amounts of input data.*

Figure 2 shows the inference latency of FST (Phonetisaurus on CPU) and proposed model (on CPU and GPU) in seconds. While fst is faster on CPU and for small amounts of input data, highly parallelized GPU based inference with sufficiently large batch size is much faster (7 times for batch size of 1024 on AWS Tesla V100 instance p3.2xlarge).

Model training time is very small it takes about 0.25 hours to train a monolingual model on CMUDict dataset in Sockeye [21] when using AWS p3.16xlarge, compared to 6 to 44 hours for previous systems reported in [18]. This greatly reduces the turn around time for hyper-parameter tuning when onboarding the model to new languages Alexa expands in.

6. Conclusion and future work

In this work, we propose a novel approach to G2P that allows us to exploit large amounts of multilingual data to enhance prediction accuracy compared to monolingual models, especially for low resource languages. Single multilingual G2P model is much more flexible and easier to maintain in production environment, compared to monolingual n-gram based models - it's easier to fine-tune it with the latest training data updates or adapt to a new language.

We are also experimenting with several general techniques, such as model ensembling, self-training, selecting n-best hypothesis based on the score of separately trained confidence model, optimizing on sequence level metric (phone error rate/phonemic distance) and using Transformers [9] as the core architecture. Each of these is bringing good accuracy improvements to the G2P model in general, although they are not directly related to the multilingual and low resource aspects we explored in this paper.

7. References

[1] J. R. Novak, N. Minematsu, and K. Hirose, "Wfst-based grapheme-to-phoneme conversion: Open source tools for alignment, model-building and decoding," in *FSMNLP*, 2012.

[2] M. Bisani and H. Ney, "Joint-sequence models for grapheme-to-phoneme conversion." *Speech Communication*, vol. 50, no. 5, pp. 434–451, 2008. [Online]. Available: http://dblp.uni-trier.de/db/journals/speech/speech50.html#BisaniN08

[3] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[4] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[5] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[6] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. B. Viégas, M. Wattenberg, G. Corrado, M. Hughes, and J. Dean, "Google's multilingual neural machine translation system: Enabling zero-shot translation," *CoRR*, vol. abs/1611.04558, 2016. [Online]. Available: http://arxiv.org/abs/1611.04558

[7] J. Pouget-Abadie, D. Bahdanau, B. van Merrienboer, K. Cho, and Y. Bengio, "Overcoming the curse of sentence length for neural machine translation using automatic segmentation." *CoRR*, vol. abs/1409.1257, 2014. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1409.html#Pouget-AbadieBMCB14

[8] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," *CoRR*, vol. abs/1705.03122, 2017. [Online]. Available: http://arxiv.org/abs/1705.03122

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[10] T. Mikolov, S. Kombrink, A. Deoras, and L. Burget, "Rnnlm-recurrent neural network language modeling toolkit," in *IEEE Automatic Speech Recognition and Understanding Workshop*, 2011.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.

[12] K. Yao and G. Zweig, "Sequence-to-sequence neural net models for grapheme-to-phoneme conversion," in *INTERSPEECH*, 2015.

[13] J. R. Novak, N. Minematsu, and K. Hirose, "Failure transitions for joint n-gram models and g2p conversion," in *INTERSPEECH*, 2013.

[14] ——, "Phonetisaurus: Exploring grapheme-to-phoneme conversion with joint n-gram models in the wfst framework," *Natural Language Engineering*, vol. 22, pp. 907–938, 2016.

[15] S. Jiampojamarn, G. Kondrak, and T. Sherif, "Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion," in *HLT-NAACL*, 2007.

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

[17] Y. M. Behbahani, B. BabaAli, and M. Turdalyuly, "Persian sentences to phoneme sequencesconversion based on recurrent neural networks," *Open Computer Science*, vol. 6, 2016.

[18] B. Milde, C. Schmidt, and J. Köhler, "Multitask sequence-to-sequence models for grapheme-to-phoneme conversion," in *INTERSPEECH*, 2017.

[19] B. Peters, J. Dehdari, and J. van Genabith, "Massively multilingual neural grapheme-to-phoneme conversion," *CoRR*, vol. abs/1708.01464, 2017.

[20] T. Schlippe, S. Ochs, and T. Schultz, "Wiktionary as a source for automatic pronunciation extraction," in *INTERSPEECH*, 2010.

[21] F. Hieber, T. Domhan, M. Denkowski, D. Vilar, A. Sokolov, A. Clifton, and M. Post, "Sockeye: A toolkit for neural machine translation," *CoRR*, vol. abs/1712.05690, 2017.