Data Augmentation for Transformer-based G2P

Zach Ryan and **Mans Hulden**
University of Colorado
{zachary.j.ryan,mans.hulden}@colorado.edu

Abstract

The Transformer model has been shown to outperform other neural seq2seq models in several character-level tasks. It is unclear, however, if the Transformer would benefit as much as other seq2seq models from data augmentation strategies in the low-resource setting. In this paper we explore methods for data augmentation in the g2p task together with the Transformer model. Our results show that a relatively simple alignment-based approach of identifying consistent input-output subsequences in grapheme-phoneme data combined with a subsequent splicing together of such pieces to generate hallucinated data works well in the low-resource setting, often delivering substantial performance improvement over a standard Transformer model.

1 Introduction

The Transformer model (Vaswani et al., 2017) has recently been shown to be robust for character-level translation tasks, outperforming other recurrent sequence-to-sequence (seq2seq) models in a wide range of tasks, including morphological inflection, grapheme-to-phoneme (g2p), and text normalization (Wu et al., 2020). A transformer-based model also served as the baseline system for both the SIGMORPHON 2020 shared tasks on grapheme-to-phoneme conversion (Gorman et al., 2020) and low-resource morphological inflection (Vylomova et al., 2020), delivering substantially better performance than other models.[1]

A common thread in research with character-level seq2seq has been that, for situations where few training examples are available, alternative strategies to produce more robust performance must be taken. For morphology tasks, this has included strategies such as instructing the model to copy input symbols to the output (Makarov et al., 2017; Makarov and Clematide, 2018; Anastasopoulos and Neubig, 2019), which may require alignment of the input and output in the training. Another strategy is data augmentation (Bergmanis et al., 2017; Silfverberg et al., 2017), whereby some mechanism is employed to generate additional training examples from the few available ones. Pointer-generator networks (Vinyals et al., 2015), which facilitate copying of the input, have also been employed (Sharma et al., 2018). Perhaps since no low-resource g2p task has previously been organized, the performance of standard models of seq2seq in settings with limited training data have not been explored as much as in morphological inflection, where data augmentation has proven to be a successful strategy.

The Transformer model, as opposed to other seq2seq models such as bidirectional LSTM encoder-decoders, however, seems to be more ro-

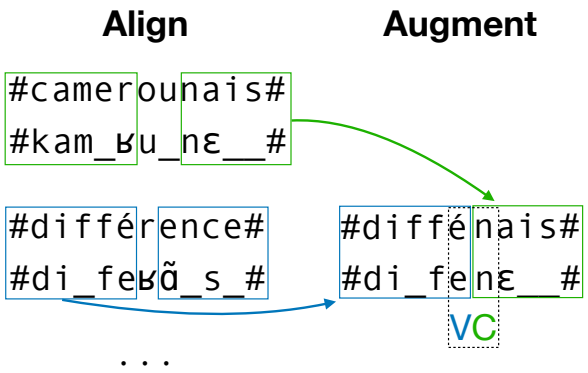

Figure 1: Augmentation strategy: after aligning all grapheme-phoneme pairs, we use input subsequences that are reliably mapped to the same output in the whole data set in creating an augmented data set from the pieces. We also enforce—using an unsupervised algorithm for detecting consonants and vowels—that only CV or VC is allowed at the boundary of the pieces spliced together.

[1]Our code is available at https://github.com/LonelyRider-cs/sig_shared_tasks.

bust by itself in the low-resource setting, at least for morphology tasks. The multiple Transformer-based baselines in the SIGMORPHON 2020 morphological task did not provide any consistent improvement by data augmentation. Also, the best-performing systems did not seem to use this strategy, even in low-resource cases. For the grapheme-to-phoneme task, it is therefore unclear if the Transformer would also benefit from one of these strategies in low-resource scenarios. The SIGMORPHON g2p task (task 1) featured uniform amounts of training data of 3600 g/p word pairs, and so can not be considered a low-resource task.

The different strategies to fortify seq2seq models in the low-resource setting in other character-level tasks are not all applicable to the g2p task, however. The array of mechanisms for learning to copy the input—special copy symbols, pointer-generator networks— favored by many low-resource morphology systems do not naturally transfer to the g2p task since the input and output pairs use different alphabets. Data augmentation, however, remains a potentially viable strategy.

In this paper we discuss experiments on the SIGMORPHON 2020 task 1 data sets where we explored data augmentation strategies for the g2p setting. Our actual submission (team **CU-Z**) to the task was a bidirectional LSTM encoder-decoder which later turned out to perform much worse than the Transformer model described in this paper. We did not finish training the Transformer models before the submission deadline, and only submitted the BiLSTM. In this paper we only discuss data augmentation and the Transformer model.

2 Data Augmentation

We experimented with two strategies of data augmentation: our first strategy was to identify in the training data grapheme sequences in the beginning of words and at the ends of words that (almost) always map to the same phoneme sequence, such as a word-initial **c** consistently mapping to **k**. Subsequently we generated new training data by swapping such sequences across words, generating new words. This initial strategy failed to provide improvements on the development set, and we moved to a more refined version of this idea, discussed in more detail below.

procurions	p ʁ ɔ k y ʁ j ɔ̃
reconnaituer	ʁ ə k ɔ n ɛ t ɥ e
brancétude	b ʁ ɑ̃ ʃ e t y d
davasonnage	d a v ɑ̃ s ɔ n a ʒ
magazoulevard	m a g a z u l v a ʁ
oucoutume	w ɛ k u t y m
socendredi	s ɔ s ɑ̃ d ʁ ə d i
thapu	t a p y
sedi	s ə d i
sagementsier	s a ʒ m ɑ̃ z j e

Table 1: Example augmented French data from the original **min** data set that contains 100 examples. In total, 50,000 examples such as the ones shown here are created from each data set.

2.1 Slice-and-shuffle

In our main strategy, we first perform a 1-to-1 alignment of the input-output data, yielding alignments such as are shown in Figure 1. For the alignment, we use an MCMC-algorithm originally developed by the second author for the SIGMORPHON 2016 shared task baseline for morphological inflection (Cotterell et al., 2016), largely similar to Expectation-Maximization based models (Ristad and Yianilos, 1998; Novak et al., 2012), but using an MCMC sampler instead. After the alignment, we investigate how consistently some part of the word-initial substring graphemes $\#i_1, \ldots, i_m$ maps to the same phonemes $\#o_1, \ldots, o_n$, and likewise for the word-final parts $i_1, \ldots, i_m\#$ and $o_1, \ldots, o_n\#$. We use # here as a symbol to denote either beginning-of-word or end-of-word. Whichever is intended should be clear from the context.

For example, in French, the initial grapheme sequence `#poin`, whenever found in the data, is always aligned with `#pwɛ̃`, and the final grapheme sequence `parer#` is consistently aligned with the phoneme sequence `paʁe#`. Such pieces can then be used to create new grapheme/phoneme pairs in an augmented training data set, such as `poinparer` → `pwɛ̃paʁe`. See Figure 1 for another example.

In particular, for an input subsequence i, we estimate its reliability as being associated with an output subsequence o as the conditional probability of the output, given the input in the usual way as:

$$p(o|i) = \frac{\text{count}(i:o) + \alpha}{\sum_{\text{ANY}} \text{count}(i:\text{ANY}) + \alpha|\text{ANY}|} \quad (1)$$

Here α is a smoothing parameter and ANY—through a slight notational abuse—represents all the witnessed different output alignments for a particular input subsequence i. For example, if we are calculating the conditional probability of some output sequence, conditioned on an initial `#pho`-sequence, and `#pho` has been aligned in the training data with `#p`, `#f`, and `#fo`, then ANY represents the set {`#p`, `#f`, `#fo`}.

To select which beginning and ending pieces can be reliably used for creation of augmented data, we declare a cutoff probability $c = 0.98$ and only use those pieces $i : o$ where $p(o|i) > c$. This yields a large number of usable pieces for each language, even in the lowest-resource setting of 100 training examples.[2] Note that the number of actual potential augmented input-output mappings corresponds to roughly the square of the number of discovered reliable beginning and ending pairings. We generate augmented words completely at random from all the pieces available to us, except we limit the output sequence length to 15 by excluding longer sequences, and put an additional restriction on the juncture where the splices come together regarding consonants and vowels, discussed below.

2.2 Consonants and Vowels

After estimating $p(o|i)$ for each seen subsequence in the training data the resulting "reliable" pieces can be spliced together to augment the data set, by combining word-initial and word-final pieces. Since phonological assimilations and coarticulations are very common in vowel-vowel and consonant-consonant sequences, and since we wish to avoid generating unnatural syllables, we do not splice together pieces where a slice ending in a phoneme-side consonant would be paired up with another one that begins with a vowel and vice versa. This is also shown in the example Figure 1. To determine which symbols on the phoneme side are consonants and vowels, we use the unsupervised algorithm in Hulden (2017) to divide the set of phonemes seen in the training data for a language into consonants and vowels. Table 1 shows a selection of French "words" generated by this complete process of aligning, determining useful pieces, and splicing them together while avoiding CC or VV sequences at the juncture of splicing.

For each language and each original-size data set (100, 500, 3600) we generate 50,000 additional training examples from the original training data. To create the low-resource data training sets from the shared task training sets, we randomly select 100 (**min**), or 500 (**med**) examples from the original training data consisting of 3,600 examples. To determine the cutoff where the data-augmentation strategy stops paying dividends, we also create an augmented data set of 50,000 examples from the original data (we call the original task data the **full**) data set.

2.3 Training details

Following Wu et al. (2020), we use a relatively small transformer model (the Fairseq implementation; Ott et al. (2019)) with 4 encoder-decoder layers, and 4 attention heads. The embedding size is 256 and hidden layer size 1024. We use dropout (0.3) during training and a batch size of 400, a learning rate of 0.001. We train the models until no improvement is seen on the dev-set for 5 epochs.

3 Results

The main results are shown in Table 2 and Figure 2. As can be seen, there is a consistent pattern of diminishing returns as more training data becomes available, with word error rates being significantly lower for almost all the augmented cases where 100 or 500 examples were used.

4 Related Work

Recurrent neural networks in a variety of models have been applied to the g2p problem, including LSTMs and bidirectional LSTMs (Rao et al., 2015), as well as convolutional networks (Yolchuyeva et al., 2019). The Transformer for g2p is investigated in Wu et al. (2020) and Yolchuyeva et al. (2020), showing improvements over previous models, at least in high-resource settings. Low-resource settings for g2p in general are examined in Jyothi and Hasegawa-Johnson (2017), and a number of papers have experimented with high-resource to low-resource transfer learning (Schlippe et al., 2014;

[2] `ady`: 2933 (100), 11358 (500); `arm`: 2789 (100), 11840 (100), `bul`: 2862 (100), 10657 (500), dut: 4422 (100), 19058 (500); `fre`: 3005 (100), 11996 (500); 3089 (100), 13698 (500); `gre`: 3667 (100), 16341 (500); `hin`: 2073 (100), 8141 (500); `hun`: 3282 (100), 13748 (500); 2438 (100), 9556 (500); `jpn`: 725 (100), 3252 (500); `kor`: 331 (100), 1280 (500); `lit`: 4328 (100), 15414 (500); `rum`: 3330 (100) 12360 (500); `vie`: 2567 (100), 12309 (500).

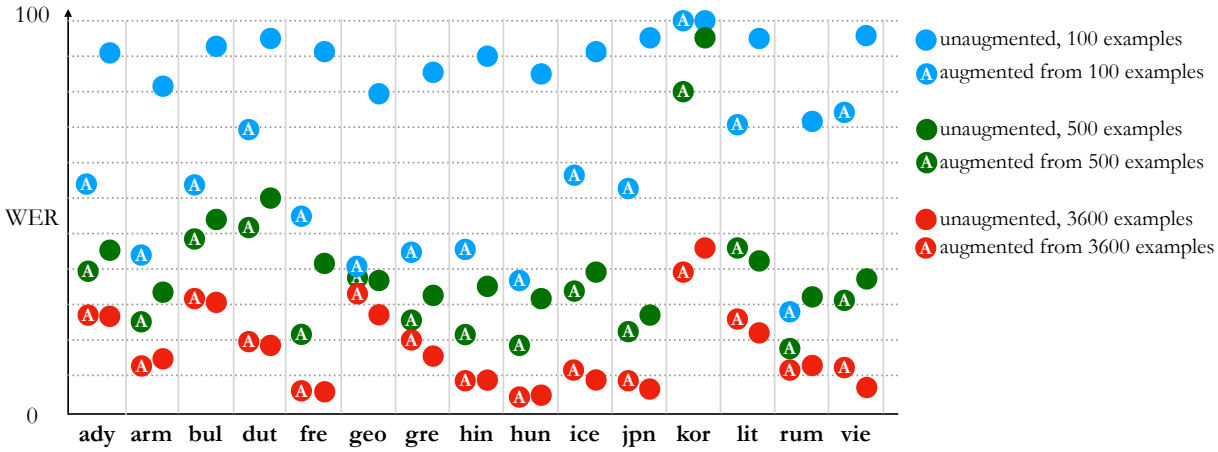

Figure 2: Main WER results on the SIGMORPHON test sets with augmented and unaugmented data.

Lang	100	100^{aug}	500	500^{aug}	full	full^{aug}
ady	90.22	**64.67**	45.33	**39.78**	**27.33**	27.78
arm	82.89	**45.33**	33.11	**24.89**	14.89	**13.33**
bul	93.56	**64.89**	53.78	**48.44**	**30.22**	32.22
dut	95.33	**69.11**	50.67	**42.00**	**18.22**	19.11
fre	91.56	**56.22**	41.78	**22.00**	**6.00**	6.22
geo	79.78	**40.89**	**37.33**	38.89	**27.78**	33.33
gre	86.00	**44.89**	32.00	**26.67**	**16.67**	20.67
hin	90.44	**46.22**	34.44	**21.33**	9.56	**9.11**
hun	84.89	**37.33**	31.78	**17.11**	4.67	**4.44**
ice	91.11	**66.89**	39.33	**33.78**	**9.56**	10.67
jpn	95.56	**62.22**	28.22	**22.00**	**6.67**	8.67
kor	**100.0**	**100.0**	95.78	**79.78**	46.22	**39.78**
lit	94.89	**70.89**	**42.89**	46.44	**21.78**	26.22
rum	70.67	**28.67**	31.56	**17.11**	12.22	**11.33**
vie	96.44	**74.89**	37.33	**30.89**	**7.11**	11.78

Table 2: Word error rate (WER) results on the test set when trained with 100 examples, 500 examples, and the full data set, compared to augmentation (^{aug}) for (100,500,3600) → 50,000 synthetic examples.

Deri and Knight, 2016), an avenue we did not explore in this work.

5 Conclusion

We have developed a method for data augmentation for the g2p task based on a 1-to-1 alignment of input/output strings together with a confidence calculation of what parts of the aligned strings can be used to splice together an augmented dataset. Used together with the popular Transformer seq2seq model, we see significant and consistent improvements on very small datasets of 100 examples, moderate improvements on medium-size datasets (500 examples), with the advantage tapering off and mostly disappearing completely with the shared tasks' datasets of 3,600 examples.

References

Antonios Anastasopoulos and Graham Neubig. 2019. Pushing the limits of low-resource morphological inflection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 984–996, Hong Kong, China. Association for Computational Linguistics.

Toms Bergmanis, Katharina Kann, Hinrich Schütze, and Sharon Goldwater. 2017. Training data augmentation for low-resource morphological inflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 31–39, Vancouver. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. The SIGMORPHON 2016 shared Task—Morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22, Berlin, Germany. Association for Computational Linguistics.

Aliya Deri and Kevin Knight. 2016. Grapheme-to-phoneme models for (almost) any language. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 399–408, Berlin, Germany. Association for Computational Linguistics.

Kyle Gorman, Lucas F. E. Ashby, Aaron Goyzueta, Arya D. McCarthy, Shijie Wu, and Daniel You. 2020. The SIGMORPHON 2020 shared task on multilingual grapheme-to-phoneme conversion. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*.

Mans Hulden. 2017. A phoneme clustering algorithm based on the obligatory contour principle. In *Proceedings of the 21st Conference on Computational*

Natural Language Learning (CoNLL 2017), pages 290–300, Vancouver, Canada. Association for Computational Linguistics.

Preethi Jyothi and Mark Hasegawa-Johnson. 2017. Low-resource grapheme-to-phoneme conversion using recurrent neural networks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5030–5034. IEEE.

Peter Makarov and Simon Clematide. 2018. Neural transition-based string transduction for limited-resource setting in morphology. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 83–93, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Peter Makarov, Tatiana Ruzsics, and Simon Clematide. 2017. Align and copy: UZH at SIGMORPHON 2017 shared task for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 49–57, Vancouver. Association for Computational Linguistics.

Josef R. Novak, Paul R. Dixon, Nobuaki Minematsu, Keikichi Hirose, Chiori Hori, and Hideki Kashioka. 2012. Improving WFST-based G2P conversion with alignment constraints and RNNLM N-best rescoring. In *Thirteenth Annual Conference of the International Speech Communication Association*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Kanishka Rao, Fuchun Peng, Haşim Sak, and Françoise Beaufays. 2015. Grapheme-to-phoneme conversion using long short-term memory recurrent neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4225–4229. IEEE.

Eric Sven Ristad and Peter N. Yianilos. 1998. Learning string-edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):522–532.

Tim Schlippe, Wolf Quaschningk, and Tanja Schultz. 2014. Combining grapheme-to-phoneme converter outputs for enhanced pronunciation generation in low-resource scenarios. In *Spoken Language Technologies for Under-Resourced Languages*.

Abhishek Sharma, Ganesh Katrapati, and Dipti Misra Sharma. 2018. IIT(BHU)–IIITH at CoNLL–SIGMORPHON 2018 shared task on universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 105–111, Brussels. Association for Computational Linguistics.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99, Vancouver. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in neural information processing systems*, pages 2692–2700.

Ekaterina Vylomova, Jennifer White, Elizabeth Salesky, Sabrina J. Mielke, Shijie Wu, Edoardo Ponti, Rowan Hall Maudslay, Ran Zmigrod, Joseph Valvoda, Svetlana Toldova, Francis Tyers, Elena Klyachko, Ilya Yegorov, Natalia Krizhanovsky, Paula Czarnowska, Irene Nikkarinen, Andrej Krizhanovsky, Tiago Pimentel, Lucas Torroba Hennigen, Christo Kirov, Garrett Nicolai, Adina Williams, Antonios Anastasopoulos, Hilaria Cruz, Eleanor Chodroff, Ryan Cotterell, Miikka Silfverberg, and Mans Hulden. 2020. The SIGMORPHON 2020 Shared Task 0: Typologically diverse morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2020. Applying the transformer to character-level transduction. *arXiv:2005.10213 [cs.CL]*.

Sevinj Yolchuyeva, Géza Németh, and Bálint Gyires-Tóth. 2019. Grapheme-to-phoneme conversion with convolutional neural networks. *Applied Sciences*, 9(6):1143.

Sevinj Yolchuyeva, Géza Németh, and Bálint Gyires-Tóth. 2020. Transformer based grapheme-to-phoneme conversion. *arXiv preprint arXiv:2004.06338*.