SIGMORPHON 2022 Shared Task on Grapheme-to-Phoneme Conversion Submission Description: Sequence Labelling for G2P

Leander Girrbach
University of Tübingen, Germany
leander.girrbach@student.uni-tuebingen.de

Abstract

This paper describes our participation in the Third SIGMORPHON Shared Task on Grapheme-to-Phoneme Conversion (Low-Resource and Cross-Lingual) (McCarthy et al., 2022). Our models rely on different sequence labelling methods. The main model predicts multiple phonemes from each grapheme and is trained using CTC loss (Graves et al., 2006). We find that sequence labelling methods yield worse performance than the baseline when enough data is available, but can still be used when very little data is available. Furthermore, we demonstrate that alignments learned by the sequence labelling models can be easily inspected.

1 Introduction

This paper describes our participation in the Third SIGMORPHON Shared Task on Grapheme-to-Phoneme Conversion (Low-Resource and Cross-Lingual) (McCarthy et al., 2022). We evaluate 3 sequence labelling methods for grapheme-to-phoneme conversion (henceforth: g2p). We approach the challenge of different lengths of grapheme and phoneme sequences by allowing to predict multiple phonemes from each grapheme.

The shared task consists of 3 tracks and includes 10 languages. The 3 tracks are high resource, low resource, and transfer. For the high resource track, grapheme-phoneme pairs are given for 1000 words. For the low resource track, grapheme-phoneme pairs are given for 100 words. For the transfer track, grapheme-phoneme pairs are given for 100 words in the target language and additionally grapheme-phoneme pairs are given for 1000 words in a transfer language (that is related to the target language, e.g. Dutch → German). The test set is the same for each track and contains 100 words of the target language. Additionally, a development set is provided for each target language. The development set also is the same for each track. All of our models are applicable to all languages and tracks.

Sequence labelling approaches can claim several advantages over the main alternative, namely (neural) encoder-decoder approaches: Sequence labelling does not require beam search for inference, may allow for smaller models, and defines a direct alignment between the input and predictions. The latter property may make models more interpretable and help with error analysis. However, sequence labelling is less flexible than encoder-decoder approaches and requires special handling of cases where the input and target sequences are of different length.

2 Related Work

Common approaches to g2p are joint-ngram models (Galescu and Allen, 2001; Novak et al., 2016), encoder-decoder models (Wu et al., 2021; Makarov and Clematide, 2018a,b; Clematide and Makarov, 2021), and sequence labelling (Jiampojamarn et al., 2007; Rosca and Breuel, 2016; Schnober et al., 2016; Ribeiro et al., 2018). In previous iterations of this shared task on g2p, encoder-decoder models were dominant both in terms of performance and in terms of number of submissions (Gorman et al., 2020; Ashby et al., 2021).

While this shows that neural encoder-decoder models yield superior performance compared to joint-ngram models, little work has been done to evaluate the performance of neural sequence labelling models. Therefore, two of our three proposed methods (explained in Section 3) directly use or build on existing approaches, namely work by Jiampojamarn et al. (2007) and Liu et al. (2017). Our third method has so far, to our knowledge, not been proposed for string transduction. It is however close to the approaches by Rosca and Breuel (2016) and Ribeiro et al. (2018): Both propose to augment the grapheme sequence by extra symbols, so that phoneme sequences that are longer than the grapheme sequence can be predicted. We propose to turn their approach upside-down and allow each

Proceedings of the 20th SIGMORPHON workshop on Computational Research in Phonetics, Phonology, and Morphology, pages 239–244

grapheme to predict multiple phonemes, instead of optionally deleting unnecessarily added input symbols.

However, in our current implementation, we predict a constant number of phonemes (which includes blank symbols) from each grapheme which is less flexible than the method proposed by Ribeiro et al. (2018), but avoids error propagation due to incorrectly predicted number of insertions. Generally, no pure sequence labelling method can achieve the same flexibility as sequence-to-sequence models, but for some problems with strong local relationship between the input sequence and the target sequence, like g2p, sequence labelling may be sufficient.

3 Method

We propose and evaluate 3 different sequence labelling methods. To refer to the different methods, we term them by their main inspiration: "Supervised" (cf. Jiampojamarn et al. (2007); Novak et al. (2016)), "Gram-CTC" (cf. Liu et al. (2017)), and "Inverse-Scatter-CTC" (cf. Ribeiro et al. (2018); Rosca and Breuel (2016)). The main challenge when applying sequence labelling methods to sequence transduction problems is finding a way to handle different lengths of the grapheme sequence and the phoneme sequence. The solution common to all our proposed methods is allowing to predict phoneme ngrams from grapheme unigrams and allowing to delete graphemes. Since in g2p the length of phoneme sequences cannot differ arbitrarily from the respective grapheme sequences, predicting ngrams, which imposes a strict bound on the length of the predicted phoneme sequence, is still a realistic approach. In the following, we describe each method in more detail.

3.1 Supervised Sequence Labelling

The supervised method is a pipeline consisting of aligning grapheme ngrams to phoneme ngrams and then training a sequence labelling model to predict phoneme ngrams from a sequence of graphemes (cf. Jiampojamarn et al. (2007)). We make the following design choices:

Our aligner is a neuralisation of the EM many-to-many aligner proposed by Jiampojamarn et al. (2007). The aligner calculates grapheme (unigram) and phoneme (unigram) embeddings from 1d convolutions applied to the grapheme sequence and phoneme sequence. Alignment scores of grapheme unigrams and phoneme unigrams are the dot-product between their embeddings. The aligner is trained by normalising the resulting alignment score matrix and maximising the alignment probability of the grapheme sequence and phoneme sequence, which can be efficiently calculated by dynamic programming. Alignments are obtained by calculating the Viterbi path through the alignment matrix. Note that this approach generalises unigram alignment scores to ngram alignments and therefore does not support deletion of graphemes, insertion of phonemes, or alignments of types other than 1-to-many and many-to-1 (which includes 1-to-1). Furthermore, the length of aligned ngrams is learned automatically and does not have to be set as hyperparameter.

Having obtained such alignments, any sequence labelling model can be trained to predict phoneme ngrams from graphemes. However, different from Jiampojamarn et al. (2007), we want to avoid training a chunker to deal with the many-to-1 case. We convert the many-to-1 case to 1-to-1 cases in the following way: Assign the aligned phoneme as label to the first grapheme in the grapheme ngram and assign deletion as label to all following graphemes in the grapheme ngram.

3.2 Gram-CTC Sequence Labelling

Gram-CTC as proposed by Liu et al. (2017) works as follows: Given a whitelist of allowed ngrams, decompose the target sequence (here: the phonemes) into all possible decompositions only containing ngrams in the whitelist. Then, for each symbol in the input sequence (here: the graphemes), calculate prediction probabilities for all ngrams in the whitelist. Also, prediction of a special blank token is possible (cf. Graves et al. (2006)). Finally, the model is trained by maximising the prediction probability of the target sequence, which is the sum of prediction probabilities of all decompositions. This sum can be efficiently computed by dynamic programming.

As whitelist, we use all phoneme ngrams that appear in alignments calculated for the supervised method (see Section 3.1). Compared to the main modus operandi described by Liu et al. (2017), who propose to use all ngrams up to a certain length, restricting the whitelist in this way stabilises and speeds up training.

Compared to the supervised method described in Section 3.1, Gram-CTC is not directly dependent

on explicit grapheme-phoneme alignments, but learns such grapheme-phoneme alignments from scratch. Therefore, Gram-CTC can correct errors made by the aligner that would otherwise directly propagate to the sequence labelling model.

3.3 Inverse-Scatter-CTC Sequence Labelling

Inverse-Scatter-CTC works as follows: For each grapheme, predict τ phoneme unigrams. Thereby, the length of the predicted phoneme sequence is increased and, given a suitable τ, so that the number of predicted phonemes is always strictly greater than the number of target phonemes, we can use standard CTC (Graves et al., 2006) to train the model. We find $\tau \geq 3$ to work for all languages in the shared task except for Persian, where only $\tau \geq 4$ works. Therefore, we evaluate $\tau \in \{3, 4, 5\}$.

Compared to the supervised approach (see Section 3.1) and Gram-CTC (see Section 3.2), Inverse-Scatter-CTC has the advantage of only using phoneme unigrams as labels, thereby reducing the number of labels and allowing for more flexible alignments. Furthermore, Inverse-Scatter-CTC is not affected by an external aligner in any way.

4 Models

For sequence labelling, we always use a plain 1-layer BiLSTM model. Models are trained using the different approaches described in Section 3. For the transfer track, we do not entirely mix the target language and the transfer language, but we share the same embeddings and LSTM encoder for both languages and use separate classification layers, since we found this to yield better performance. Our intuition is that transfer data stabilises training and mitigates overfitting of embeddings and the LSTM encoder, but the target phonemes differ to a degree that makes separate decoding necessary.

For each language and track, we train 10 models and keep the best 5 performing models in terms of WER on the development set. We use these 5 models to compute word-level majority-voted ensemble predictions. We resolve ties by choosing the prediction from the model with lowest WER on the development set among all predictions with most votes.

The different approaches also require different hyperparameters. However, common to all setups are embedding size 64, no dropout or weight decay, vanilla SGD optimizer with one-cycle learning rate scheduler (Smith and Topin, 2019), and only keeping the best checkpoint from each training run based on WER on development set evaluated after every epoch. Hyperparameters that differ are training for 100 epochs with batch size 2, max. learning rate 0.01, clipping gradients with absolute value greater than 1 and the LSTM encoder having 128 hidden units for supervised and Gram-CTC, whereas Inverse-Scatter-CTC models are trained for 80 epochs with batch size 16, max. learning rate 0.1, no gradient clipping, and 256 hidden units for the LSTM encoder. All models and training routines are implemented in PyTorch (Paszke et al., 2019).

5 Results

In Table 1, we report test set word error rates (WER) of supervised and Gram-CTC models for all languages and tasks. These are the official results made available by the organisers. For the high and low resource scenarios, Gram-CTC improves upon supervised training for 6 out of 10 languages, and the macro-average WER is around 4 points lower. The most pronounced difference is for Thai in the high resource setting. This suggests that it is indeed helpful to allow learning to realign phoneme ngrams, as is done by Gram-CTC.

However, for the transfer task, we make different observations: While Gram-CTC performs worse in the transfer setting than in the low resource setting, supervised training is able to use the additional transfer data in order to improve upon its performance in the low resource setting. The improvement amounts to approximately 7 points in WER (macro-average). Therefore, transfer data can indeed be very helpful when used with the right kind of model.

This being said, Gram-CTC still outperforms supervised training in 6 out of 10 languages (transfer track). The fact that the macro-average WER of the supervised model is eventually lower is mainly due to the much better performance of supervised training on Tagalog (24 vs 50). If we ignore Tagalog when calculating macro-average scores, Gram-CTC (both low and transfer tracks) and supervised training (transfer track) perform almost equally well, with a slight advantage for Gram-CTC in transfer setting.

In Table 2, we report WER for Inverse-Scatter-CTC. In the high resource setting, we observe that greater τ (more outputs predicted from each grapheme) is beneficial. While there does not seem

	high		low		transfer	
	gram-ctc	supervised	gram-ctc	supervised	gram-ctc	supervised
ben	68.49	71.23	90.41	91.78	82.19	80.82
bur	37.00	51.00	90.00	95.00	92.00	94.00
ger	50.00	47.00	81.00	89.00	79.00	80.00
gle	33.00	39.00	78.00	86.00	78.00	82.00
ita	19.00	15.00	44.00	48.00	41.00	36.00
per	57.89	61.40	75.44	80.70	80.70	82.46
swe	54.00	51.00	84.00	86.00	82.00	74.00
tgl	15.00	14.00	40.00	42.00	50.00	24.00
tha	39.00	57.00	91.00	96.00	91.00	95.00
ukr	36.00	41.00	73.00	84.00	77.00	81.00
Avg.	40.94	44.76	74.68	79.85	75.29	72.93

Table 1: Word error rates (WER) for supervised and Gram-CTC models. Avg is macro-average over languages.

to be a visible trend for the low resource track, greater τ seems harmful in terms of macro-average WER for the transfer track.

Depending on τ, performance of Inverse-Scatter-CTC can be slightly better than performance of Gram-CTC, but we do not observe any decisive advantages. We demonstrate this in Figure 1: The numbers in the heatmap show for how many languages the model on the x-axis achieves strictly lower WER than the model on the y-axis. Despite having the lowest macro-average, supervised training actually is not superior to any model for more than 50% of the languages. Contrarily, Inverse-Scatter-CTC with τ=4 achieves better performance than most models for more than 50% of the languages, but has second-worst macro-averaged WER. Overall, Figure 1 shows that which model is best is language dependent and there is no clear winner among the models evaluated in this paper. Similar results can be found also for the high and the low resource tracks.

Compared to the baseline[1], our models generally perform worse in the high-resource tack, but better in the low resource and transfer tracks. This suggests that sequence-to-sequence models may be superior to sequence labelling models when enough data is available, while it is still possible to train neural (sequence labelling) models in the ultra-low resource settings.

[1]Results taken from https://github.com/sigmorphon/2022G2PST#baseline

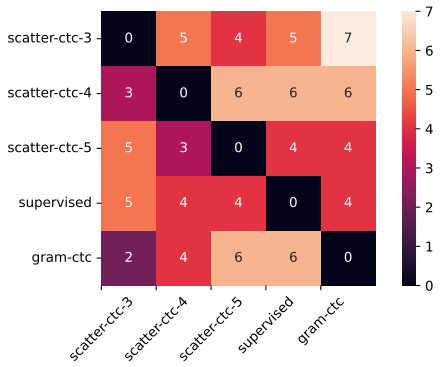

Figure 1: Heatmap showing for how many languages (out of 10) the model on the y-axis achieves strictly lower test set WER than the model on the x-axis. Results are shown for the transfer track.

6 Model Inspection

In the introduction, we claim that one advantage of sequence labelling methods is that they define a direct alignment between graphemes and phonemes, which can be inspected. Therefore, in Table 3, we give the 10 most frequent graphemes appearing in the German development set and their respective phonemes as predicted by the best trained German Inverse-Scatter-CTC model (τ=5). From Table 3, we can see that most alignments are reasonable (of course one would also have to look at the context). This means that the Inverse-Scatter-CTC indeed learns useful alignments of graphemes to phonemes. However, there are also some problems: While the German model seems to handle deletions (e.g. "sch" → "ʃ") rather well, it struggles with predicting multiple phonemes from one grapheme.

	high			low			transfer		
τ	3	4	5	3	4	5	3	4	5
ben	72.6	69.86	68.49	83.56	83.56	86.30	89.04	89.04	83.56
bur	31.0	37.00	35.00	87.00	86.00	87.00	90.00	93.00	86.00
ger	50.0	45.00	46.00	83.00	84.00	82.00	74.00	74.00	74.00
gle	35.0	37.00	36.00	76.00	76.00	79.00	74.00	80.00	81.00
ita	18.0	18.00	19.00	49.00	51.00	45.00	41.00	38.00	40.00
per	100.0	57.89	56.14	80.70	85.96	82.46	100.00	78.95	82.46
swe	53.0	51.00	52.00	81.00	81.00	81.00	77.00	80.00	74.00
tgl	16.0	18.00	15.00	35.00	37.00	32.00	40.00	68.00	92.00
tha	38.0	36.00	35.00	84.00	83.00	86.00	83.00	81.00	94.00
ukr	41.0	39.00	44.00	79.00	80.00	77.00	74.00	76.00	92.00
Avg.	45.46	40.88	40.66	73.83	74.75	73.78	74.20	75.80	79.90

Table 2: Word error rates (WER) for Inverse-Scatter-CTC models. τ is the number of outputs predicted from each grapheme. Avg is macro-average over languages.

Grapheme	Predicted Phonemes
e	ə, ␣, a, ɛ, eː, ʔ a, ɔ, ʔ ɛ, n̩
n	n, ␣, ŋ
r	ʁ, ɐ, ɐ̯, r
t	t, ␣
a	a, aː
i	ɪ, ɪ̯, iː, i, ʔ iː
s	ʃ, s, z, ␣, t͡s
h	␣, h
l	l, ␣
u	ʊ, u, ʊ̯, uː, ʊ̯

Table 3: Phoneme predictions of the 10 most frequent graphemes in the development set. Both graphemes and phonemes are sorted by frequency in descending order. "␣" denotes deletion.

In German, this is a rather rare phenomenon, occurring only for the grapheme "x", which is pronounced as "k s", and also for the glottal stop ʔ when words start with a vowel. For example, the pronunciation of German "axt" (English: "axe, ax") is predicted as "a k t", while "a k s t" is correct.

One possibility of how to make use of these direct alignments is using predefined mappings of graphemes to phonemes to restrict which phonemes may be predicted. Another advantage, of course, is error analysis. For the German model, for example, we would recommend adding more examples containing "x" to the training set. In fact, there is only one such example in the high resource training data, namely "existieren" (English: "to exist").

7 Conclusion

We presented and evaluated 3 sequence labelling methods for g2p: Supervised, Gram-CTC, and Inverse-Scatter-CTC. We show that all 3 methods can be applied to all 3 tracks, but no method seems clearly superior to the other methods. In the high resource setting, the baseline sequence-to-sequence model seems to yield better performance than sequence labelling methods. However, sequence labelling methods seem to perform better in the very low resource settings. Finally, we show that Inverse-Scatter-CTC models learn reasonable alignments of graphemes and phonemes, thereby validating the claim that sequence labelling models allow for comparatively easy model inspection.

Acknowledgements

We thank Çağrı Çöltekin for helpful discussions and providing access to computation resources. We thank the organisers for organising this shared task.

References

Lucas F.E. Ashby, Travis M. Bartley, Simon Clematide, Luca Del Signore, Cameron Gibson, Kyle Gorman, Yeonju Lee-Sikka, Peter Makarov, Aidan Malanoski, Sean Miller, Omar Ortiz, Reuben Raff, Arundhati Sengupta, Bora Seo, Yulia Spektor, and Winnie Yan. 2021. Results of the second SIGMORPHON shared task on multilingual grapheme-to-phoneme conversion. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 115–125, Online. Association for Computational Linguistics.

Simon Clematide and Peter Makarov. 2021. CLUZH at SIGMORPHON 2021 shared task on multilingual grapheme-to-phoneme conversion: Variations on a baseline. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 148–153, Online. Association for Computational Linguistics.

Lucian Galescu and James F. Allen. 2001. Bi-directional conversion between graphemes and phonemes using a joint n-gram model. In *4th ITRW on Speech Synthesis, Perthshire, Scotland, UK, August 29 - September 1, 2001*, page 131. ISCA.

Kyle Gorman, Lucas F.E. Ashby, Aaron Goyzueta, Arya 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*, pages 40–50, Online. Association for Computational Linguistics.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, volume 148 of *ACM International Conference Proceeding Series*, pages 369–376. ACM.

Sittichai Jiampojamarn, Grzegorz Kondrak, and Tarek Sherif. 2007. Applying many-to-many alignments and hidden Markov models to letter-to-phoneme conversion. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 372–379, Rochester, New York. Association for Computational Linguistics.

Hairong Liu, Zhenyao Zhu, Xiangang Li, and Sanjeev Satheesh. 2017. Gram-CTC: Automatic unit selection and target decomposition for sequence labelling. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 2188–2197. PMLR.

Peter Makarov and Simon Clematide. 2018a. Imitation learning for neural morphological string transduction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2877–2882, Brussels, Belgium. Association for Computational Linguistics.

Peter Makarov and Simon Clematide. 2018b. 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.

Arya D. McCarthy, Jackson L. Lee, Alexandra DeLucia, Winston Wu, Travis M. Bartley, Milind Agarwal, Lucas F.E. Ashby, Luca Del Signore, and Cameron Gibson. 2022. Results of the third SIGMORPHON shared task on cross-lingual and low-resource grapheme-to-phoneme conversion. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Seattle, USA. Association for Computational Linguistics.

Josef R. Novak, Nobuaki Minematsu, and Keikichi Hirose. 2016. Phonetisaurus: Exploring grapheme-to-phoneme conversion with joint n-gram models in the WFST framework. *Nat. Lang. Eng.*, 22(6):907–938.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Joana Ribeiro, Shashi Narayan, Shay B. Cohen, and Xavier Carreras. 2018. Local string transduction as sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1360–1371, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Mihaela Rosca and Thomas M. Breuel. 2016. Sequence-to-sequence neural network models for transliteration. *CoRR*, abs/1610.09565.

Carsten Schnober, Steffen Eger, Erik-Lân Do Dinh, and Iryna Gurevych. 2016. Still not there? comparing traditional sequence-to-sequence models to encoder-decoder neural networks on monotone string translation tasks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1703–1714, Osaka, Japan. The COLING 2016 Organizing Committee.

Leslie N. Smith and Nicholay Topin. 2019. Super-convergence: very fast training of neural networks using large learning rates. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, volume 11006 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, page 1100612.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2021. Applying the transformer to character-level transduction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1901–1907, Online. Association for Computational Linguistics.