# The SIGMORPHON 2020 Shared Task on Multilingual Grapheme-to-Phoneme Conversion

**Kyle Gorman**[*], **Lucas F.E. Ashby**[*], **Aaron Goyzueta**[*],
**Arya D. McCarthy**[†], **Shijie Wu**[†], **Daniel You**[‡]
[*]The Graduate Center, City University of New York
[†]Johns Hopkins University
[‡]Jericho High School

## Abstract

We describe the design and findings of the SIGMORPHON 2020 shared task on multilingual grapheme-to-phoneme conversion. Participants were asked to submit systems which consume a sequence of graphemes then emit output a sequence of phonemes representing the pronunciation of that grapheme sequence in one of fifteen languages. Nine teams submitted a total of 23 systems, at best achieving an 18% relative reduction in word error rate (macro-averaged over languages), versus strong neural sequence-to-sequence baselines. To facilitate error analysis, we publicly release the complete outputs for all systems—a first for the SIGMORPHON workshop.

## 1 Introduction

Speech technologies such as automatic speech recognition and text-to-speech synthesis require mappings between written words and their pronunciations. Even recent attempts to do away with explicit pronunciation models via “end-to-end” systems (e.g., Watts et al. 2013, Chan et al. 2016, Sotelo et al. 2017, Chiu et al. 2018, Pino et al. 2019, McCarthy et al. 2020) must induce an implicit mapping of this sort. For open-vocabulary applications, these mappings must generalize to unseen words, and so must be expressed as mappings between sequences of *graphemes*—i.e., glyphs—and *phonemes* or *phones*—i.e., sounds.[1]

For some languages, this mapping is sufficiently consistent that a literate, linguistically-sophisticated speaker can simply enumerate the necessary rules; this sequence of rules can then be compiled into a finite-state transducer (e.g., Sproat 1996, Black et al. 1998). However, rule-based systems require linguistic expertise to develop and maintain, and may be brittle or inaccurate. Therefore, modern speech engines usually treat grapheme-to-phoneme conversion as a machine learning problem, either using generative models expressed as weighted finite-state transducers (e.g., Taylor 2005, Bisani and Ney 2008, Wu et al. 2014, Novak et al. 2016) or discriminative models based on conditional random fields (Lehnen et al. 2013), recurrent neural networks (e.g., Rao et al. 2015, Yao and Zweig 2015, van Esch et al. 2016, Lee et al. 2020) or transformers (Yolchuyeva et al. 2019).

While the grapheme-to-phoneme conversion (or G2P) task is crucial to speech technology, the vast majority of published research focuses on English or a few other highly-resourced, globally hegemonic languages for which free pronunciation dictionaries are available. One exception, a recent study by van Esch et al. (2016), compares naïve rule-based systems and neural network-based sequence-to-sequence models for 20 languages; unfortunately, the data used in this study is proprietary. Like many other types of language resources, pronunciation dictionaries are expensive to create and maintain, and until recently, free high-quality dictionaries were only available for a small number of languages.

This limitation to a handful of languages is unfortunate because, as we discuss below, writing systems are almost as diverse as languages themselves. Therefore, we present a *multilingual grapheme-to-phoneme conversion task* with data sets, evaluation metrics, and strong baselines. In this we are aided by the recent release of WikiPron (Lee et al. 2020), a freely available collection of pronunciation dictionaries. The resulting task, the

[1] We note that the term *phoneme* is a well-defined object in linguistic theory, and that referring to the elements of transcriptions as *phonemes* makes strong ontological commitments which may not be appropriate for a given pronunciation dictionary (cf. Lee et al. 2020, fn. 4). Therefore, in what follows we use the term *phone*, in a pre-theoretical sense, to refer to transcriptions symbols.

*Proceedings of the Seventeenth SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 40–50
Online, July 10, 2020. ©2020 Association for Computational Linguistics
https://doi.org/10.18653/v1/P17

first of its kind, included data from fifteen languages and scripts, and received 23 submissions from nine teams.

## 2 Data

Fifteen language/script pairs were chosen to cover a wide variety of script types. Ten of the scripts are alphabetic systems known to descend from Phoenician (and ultimately from Egyptian hieroglyphs); of these, seven are variants of the Latin script. Two others, the Armenian *aybuben* and the Georgian *mkhedruli*, are alphabetic scripts of unknown origin, but may ultimately be modeled on Greek (Sanjian 1996). The *devanāgarī* script used to write Hindi, is an *alphasyllabary*, in which most glyphs—known traditionally as *akṣara*—denote consonant and consonant-vowel sequences. Vowels (or their absence) are primarily indicated with diacritics. It too is thought to ultimately descend from Phoenician. *Hiragana*, one of several scripts used to write Japanese, is a *syllabary*, in which most glyphs denote entire syllables The glyphs themselves are derived from Chinese characters. Like hiragana, the Korean *hangul* script is also a syllabary It may have been have been inspired by *'phags-pa*, a Tibetan alphabet which is itself a distant cousin of devanāgarī (Ledyard 1966).

It is important to note that languages—and the scripts used to write them—differ enormously in their affordances for grapheme-to-phoneme conversion. Writing systems are, at their core, linguistic analyses, albeit sometimes quite naïve, and (as argued in DeFrancis 1989) explicitly encode details of the phonological and phonetic structure of the language they are used to write. Still, the exact details of these mappings can vary greatly between even closely related languages and/or scripts. Whereas related languages may retain telltale grammatical features across millennia, dozens of languages have abruptly switched from one script to another in just the last century, usually in response to political—rather than linguistic—concerns. It is thus unsurprising that Bjerva and Augenstein (2018) find grapheme embeddings induced by training G2P systems are poorly correlated with gross phonological typology, and experiments with "polyglot" G2P models (e.g., Peters et al. 2017) have produced equivocal results.

While we did not pay particular attention to language families when selecting language family, we note that nine of the languages are Indo-European (though no two are closely related) and none of the remaining six—Adyghe, Georgian, Hungarian, Japanese, Korean, and Vietnamese—are known to be genetically related to each other.

## 3 Methods

The primary data for the shared task is derived from WikiPron (Lee et al. 2020), a massively multilingual resource of grapheme–phoneme pairs extracted from Wiktionary, an online multilingual dictionary. Depending on language and script, these pronunciations may be manually entered by human volunteers—usually working from language-specific pronunciation guidelines—or generated using server-side scripting routines; some languages (e.g., Bulgarian and French) use a mixture of the two approaches. WikiPron is configured to apply case-folding where appropriate. It removes stress and syllable boundary markers and segments pronunciation strings—encoded in the International Phonetic Alphabet—using the `segments` library (Moran and Cysouw 2018).

For this task, words with multiple pronunciations—both homographs and free pronunciation variants—were excluded, since pronunciations for such words are often selected by a rather different procedure: they are chosen from a small, predetermined set of possible pronunciations using classifiers conditioned on local context (e.g., Gorman et al. 2018).

Training and development data for ten languages—the "development" languages—was released at the start of the task; equivalent data for the five "surprise" languages was released one week before the start the evaluation phase. Table 1 provides sample training data pairs for the development and surprise languages.

As there is considerable variation in the number of available examples for any given language, each languages' data was downsampled to 4,500 examples. We regard as a "medium-resource" setting for this task; these data sets are, for instance, several orders of magnitude smaller than the proprietary G2P data used by van Esch et al. (2016). Following similar procedures in other shared tasks (e.g., Cotterell et al. 2017), words were sampled according to their frequency in the largest available Wortschatz (Goldhahn et al. 2012) corpus for that language. These frequencies were smoothed by adding a 0.3 pseudo-count to the frequency of all WikiPron entries. Wortschatz frequency data

| Language | ISO 639-2 | Example training data pair | |
|---|---|---|---|
| Armenian | arm | մեծաքանակ | m ɛ t͡s ɑ kʰ ɑ n ɑ k |
| Bulgarian | bul | североизток | s ɛ v ɛ r o i s t o k |
| French | fre | hébergement | e b ɛ ʁ ʒ ə m ɑ̃ |
| Georgian | geo | ფორმიანი | pʰ ɔ r m ɪ ɑ n ɪ |
| Modern Greek | gre | καθισμένες | k a θ i z m e n e s |
| Hindi | hin | कैलकुलेटर | k ɛː l k ʊ l eː ʈ ə ɾ |
| Hungarian | hun | csendőrök | t͡ʃ ɛ n d øː r ø k |
| Icelandic | hin | þýskaland | θ i s k a l a n t |
| Korean | kor | 말레이시아 | m a̠ ɭ ɭ e̞ i ɕʰ i a̠ |
| Lithuanian | lit | galinčiais | ɡ aː lʲ ɪ nʲ tʲ ʃʲ ɛ j s |
| | | | |
| Adyghe | ady | бзыукъолэн | b z ə w qʷ a l a n |
| Dutch | dut | aanduiding | aː n d œ y̯ d ɪ ŋ |
| Japanese hiragana | jpn | どちらさま | d o̞ t͡ɕ i ɾ ȃ̠ s a̠ m a̠ |
| Romanian | rum | bineînțeles | b i n e ɨ n t s e l e s |
| Vietnamese | vie | duyên phận | z w i ə n ˧˧ f ə n ˧˨ ʔ |

Table 1: Languages, language codes, and example training data pairs for the shared task.

was not available for Adyghe, so uniform sampling was used for this language.

The downsampled data was then randomly split into training (80%; 3,600 examples), development (10%; 450 examples), and testing (10%; 450 examples) shards. For some languages, Wiktionary contains pronunciations for both lemmas (i.e., headwords, citations forms) and inflection variants; for others, pronunciations are only available for lemmas. We hypothesized that cases where one inflectional variant of a lemma is present in the training data and another in the test data—as might occur if the data was split totally at random—would make the overall task somewhat easier. To forestall this possibility, the splitting procedure was constrained so that all inflectional variants of any given lemma—according to the UniMorph 2 (Kirov et al. 2018) paradigm tables, also extracted from Wiktionary—are limited to a single shard. For example, since the French word *acteur* 'actor' occurs in the training shard, so must its plural form *acteurs*. This additional constraint was applied to all languages but Japanese and Vietnamese, for which no UniMorph data was available. We note that Wiktionary does not generally provide pronunciations for inflectional variants in Japanese, and that Vietnamese is a highly isolating language with no discernable system of inflection (Noyer 1998), so this is unlikely to have introduced bias.

## 4 Evaluation

The primary metric for this task was word error rate (WER); we also report phone error rate (PER).

**WER** This is the percentage of words for which the hypothesized transcription sequence is not identical to the gold reference transcription; lower WER indicates better performance. Following common practice in speech research, we multiply the WER by 100 and display it as a percentage. We choose this as the primary metric for the shared task because we hypothesize that *any* G2P error, no matter how small, will result in a substantial degradation in subjective quality for downstream speech applications.

**PER** This is a more forgiving measure measuring the normalized distance (i.e., in number of insertions, deletions, and substitutions) between the predicted and reference transcriptions. It is computed by summing the minimum edit distance—computed with the Wagner and Fischer (1974) algorithm—between prediction and reference transcriptions, and dividing by the sum of the reference transcription lengths. That is,

$$\text{PER} := 100 \times \frac{\sum_i^n \text{edits}(p, r)}{\sum_i^n |r|}$$

where $p$ is the predicted pronunciation sequence, $r$ is the reference sequence, and edits$(p, r)$ is the

Levenshtein distance between the two. Once again, we multiply it by 100, though strictly speaking it is not a true percentage because it can hypothetically exceed 100. As with WER, lower PER indicates better performance.

Participants were provided with two evaluation scripts: one which computes the two metrics for a single language, and another which macro-averages the metrics across all languages.

## 5 Baselines

Three baselines were made available at the start of the task. To aid reproducibility, participants were also provided with a Conda "environment", a schematic that allows users to reconstruct the exact software environment used to train and evaluate the baselines. Several submissions made use of the baselines for data augmentation or ensemble construction. We make these baseline implementations available under the `task1/baselines` subdirectory of the shared task repository.[2]

**Pair n-gram model** The first baseline consists of a pair n-gram model, which be can thought of as a finite-state approximation of a hidden Markov model with states representing graphemes and emissions representing output phones. The model is quite similar to the Phonetisaurus toolkit (Novak et al. 2016), but here is implemented using the OpenGrm toolkit (Roark et al. 2012, Gorman 2016); see Lee et al. 2020 for a full description. The sole hyperparameter for this model, Markov model order, is tuned separately for each language using the development set.

**Encoder-decoder LSTM** The second baseline is a neural network sequence-to-sequence model consisting of a single-layer bidirectional LSTM encoder and a single-layer unidirectional LSTM decoder connected using an attention mechanism (Luong et al. 2015). It is implemented using the `fairseq` library (Ott et al. 2019). LSTM-based encoder-decoder models have been claimed to outperform pair n-gram G2P models, both in monolingual (e.g., Rao et al. 2015, Yao and Zweig 2015) and multilingual (e.g., van Esch et al. 2016, Lee et al. 2020) evaluations, though these prior studies use substantially more training data than is available in this task. During training, we perform 4,000 updates to minimize label-smoothed cross-entropy (Szegedy et al. 2016) with a smoothing rate of .1. We use the Adam optimizer (Kingma and Ba 2015) with a learning rate of $\alpha = .001$ and weight decay coefficients of $\beta = (.9, .98)$, and clip norms exceeding 1.0. We use the development set to tune—for each language—batch size (256, 512, 1024), dropout (.1, .2, .3), and the size of the encoder and decoder modules. A module is said to be "small" when it has a 128-dimension embedding layer and a 512-unit hidden layer, and "large" when it has a 256-dimension embedding layer and a 1024-unit hidden layer. In both cases, the decoder shares a single embedding layer for both inputs and outputs. Altogether, this defines a 36-element hyperparameter grid. During tuning, we employ a form of *early stopping*; we save a checkpoint every 5 epochs, and then use the checkpoint that achieves the lowest WER on the development set. We use a beam of size 5 for decoding.

**Encoder-decoder transformer** The third baseline is a transformer, a neural sequence-to-sequence models that replaces hidden layer recurrence with layers of multi-head self-attention (Vaswani et al. 2017). Once again, it is implemented using `fairseq`. Here the model consists of four encoder layers and four decoder layers, both with pre-layer normalization, tuned for character-level tasks (Wu et al. 2020). The hyperparameter grid, tuning procedures, and beam size are the same as for the LSTM model above, except that learning rate is decayed on an inverse square-root schedule after a 1,000-update linear warm-up period. While most participants chose to compare their results to the transformer and not the LSTM in system description papers, the transformer was outperformed by the LSTM baseline in most setting with the hyperparameter exploration budget.

## 6 System descriptions

Below we provide brief descriptions of submissions to the shared task.

**CLUZH** The Institute of Computational Linguistics at the University of Zurich submitted a single system (Makarov and Clematide 2020) extending earlier work (Makarov and Clematide 2018) on imitation learning-based transducers that output a sequence of edit actions rather than a target string itself. To adapt to the G2P task, where input (grapheme) and output (phone) vocabularies are largely disjoint, they add a substitution action. The costs of each edit action are drawn from a weighted

[2] `https://github.com/sigmorphon/2020`

finite state transducer (WFST). The authors suggest that external lexical information such as part of speech, etymology (borrowing particularly) and morphological segmentation would improve systems. During preprocessing, they decompose Korean hangul characters into their constituent *jamo*, each corresponding roughly to a single phoneme.

**CU** One team from the University of Colorado Boulder (Prabhu and Kann 2020) ensembled several transformer models created with different random seeds using majority voting. They also experiment with a form of multi-task learning: they train a "bidirectional" model to do both grapheme-to-phoneme and phoneme-to-grapheme prediction.

**CUZ** A second team from the University of Colorado Boulder (Ryan and Hulden 2020) uses a "slice-and-shuffle" data augmentation strategy. First, they perform character-level one-to-one alignment between graphemes and phonemes. Then they concatenate frequent subsequence pairs to each other to create nonce training examples. Their submission is an LSTM model with a bidirectional encoder trained on this augmented data. While they also developed transformer models, these did not finish training in time for submission. Results for their transformer system, not reported here, are included in their system description.

**DeepSPIN** Researchers at the Instituto Superior Técnico and Unbabel produced four submissions (Peters and Martins 2020) based on sparse attention models. Each submission consists of a single multilingual neural model in which separate learned "language embeddings" are concatenated to all encoder and decoder states, rather than prepending a language-identification token to the input sequence. Their submissions either use LSTM- or transformer-based encoder-decoder sequence-to-sequence models with different values of a hyperparameter enforcing sparsity in the final layer (Peters et al. 2019). Like CLUZH, they preprocess Korean hangul characters, decomposing them into constituent *jamo*, each corresponding roughly to a single phoneme.

**IMS** A single submission from the Institut für Maschinelle Sprachverarbeitung at the University of Stuttgart (Yu et al. 2020) uses *self-training* (Yarowsky 1995) and ensembles of the baseline models. The components of the ensemble are selected using a genetic algorithm. They report that their data augmentation does not affect performance substantially, except in a simulated low-resource setting with 200 training examples. They romanize Japanese and Korean texts as a preprocessing step, and they use external word frequency lists.

**NSU** The Novosibirsk State University team did not provide a system description.

**UA** The submissions from the University of Alberta (Hauer et al. 2020) either use a non-neural discriminative string transduction model (DTLM; Nicolai et al. 2018), or tranformers. They leverage both grapheme-to-phoneme and phoneme-to-grapheme models to filter candidates for data augmentation, enforcing a cyclic consistency constraint. They further show strong performance in a simulated low-resource scenario with 100 training examples. They note that the DTLM system is much faster to train than transformer models. Their six submissions vary the amount of training data and use either DTLM, a transformer, or a transformer with data augmentation.

**UBCNLP** The University of British Columbia submitted two systems (Vesik et al. 2020). One is a multilingual model akin to Peters et al. (2017), in which a language-identification token is prepended to the input sequence. They also ensemble multiple checkpoints. Their second submission adds self-training on Wikipedia text; they report that this data augmentation strategy does not improve scores.

**UZH** For all three of their submissions, the team from the Department of Informatics at the University of Zurich (ElSaadany and Suter 2020) used a single set of encoder-decoder parameters shared across all languages. UZH-1 is a large transformer model with large embedding, hidden layers, and batches, with a high dropout probability. UZH-2 augments this model with WikiPron data for six other languages. UZH-3 is an ensemble of the previous two models which selects from the predictions of the two component models using whichever model's prediction has a higher posterior probability. The ensemble outperformed the component models for most languages. During preprocessing they also decompose Korean hangul characters into their constituent jamo; they report this results in a 46% relative word error reduction.

# 7 Results

We now review baseline and submission results.

## 7.1 Baseline results

Baseline results are shown in Table 2. The encoder-decoder LSTM (Lee et al. 2020) performed best for nine out of fifteen languages; the transformer was the strongest for four languages, and for the remaining two—Modern Greek and Hungarian—there was a virtual tie between the two neural network baselines. The pair n-gram model was outperformed by the neural baselines on all languages, and by 10 or more points WER in Bulgarian, Georgian, and Korean. This suggest that this model is no longer competitive with powerful discriminative neural methods, at least in this medium-resource G2P task.

While this task was not designed explicitly to compare LSTM and transformer sequence-to-sequence models, it does suggest an advantage for LSTM models. However, we speculate that additional training data, or a more generous hyperparameter tuning budget, might favor transformer models. Indeed, anticipating the results below, the one team that directly compared transformer and LSTM systems, DeepSPIN, achieved the third best submission overall using a transformer.

We also note that for four languages, the baseline system that achieves the best WER does not achieve the best PER, though the two metrics produce the same one-best ranking for the remaining eleven languages.

## 7.2 Submission results

Table 3 shows, for each language, the system or systems that achieved the best WER, as well as the best baseline WER. For all fifteen languages, at least one team outperformed the baselines, sometimes quite substantially. Six of the nine teams achieved the best WER on at least one language. More detailed per-language, per-submission results are available online.[3]

Table 4 gives the macro-averaged WER and PER for the three baselines, and for the best overall submission from each team. As expected, the strongest baseline is the LSTM model. Across all submissions, the IMS team achieves both the lowest average WER, a 3% absolute (18% relative) word error reduction over the LSTM baseline, and the lowest overall PER, a 1% absolute (31% relative) phone error reduction over the LSTM baseline. The CLUZH and DeepSPIN-3 submissions achieve second and third place, respectively; the CU, UCBNLP, and UZH teams also submitted systems that outperform the LSTM baseline's WER.

# 8 Discussion

When this task was initially proposed, there was some concern that the submissions—if not the baselines themselves—would easily achieve perfect or near-perfect performance on some languages. This was not the case. Even on the "easiest" language, the best submission has .89% WER, and for three languages, no submission achieves an error rate below 20%.

At the same time, we observe a large range of error rates across languages. It is tempting to speculate that word and/or phone error rates actually represent differences in difficulty. Insofar as this is correct, we can begin to ask what makes a language "hard to pronounce", much like how Mielke et al. (2019) ask what makes a language "hard to language-model".

One thing that may make a language hard to pronounce is data sparsity. Consider the case of Korean, which has by far the highest baseline error rate of all fifteen languages. Three features of Korean and of hangul conspire to make this task particularly challenging. First, hangul is a syllabary, and therefore necessarily has a much larger graphemic inventory than an alphabet or alphasyllabary. A whopping 889 unique hangul characters appear across the 4,500 words used for this task.[4] Secondly, hangul is a relatively *deep* or *abstract* orthography (in the sense of Rogers 2005); it operates at a roughly-morphophonemic level whereas Lithuanian and Hungarian, for example, are is roughly phonemic. Third, Korean has many phonological processes that operate across syllable boundaries. Since the effect of these processes is not indicated by the highly abstract, morphophonemic orthography, they can only be learned by observing the targeted syllable bigrams during training. Lee et al. (2020) perform a manual error analysis of a Korean G2P system similar

[3] https://docs.google.com/spreadsheets/d/1g0HyGeVzFrNt2pvNuu8L1voFFQY-0CwjTxGA3VXXNGI/edit?usp=sharing

[4] Few syllabaries are so large. For instance, there are only 79 unique hiragana symbols in the Japanese data, but this relative size difference is not surprising given that Korean has a more permissive syllable structure than Japanese.

| | Pair n-gram | | LSTM | | Transformer | |
|---|---|---|---|---|---|---|
| | WER | PER | WER | PER | WER | PER |
| arm | 18.00 | 3.90 | 14.67 | 3.49 | **14.22** | **3.29** |
| bul | 41.33 | 9.05 | **31.11** | **5.94** | 34.00 | 7.89 |
| fre | 13.56 | 3.12 | **6.22** | **1.32** | 6.89 | 1.72 |
| geo | 37.78 | 6.48 | **26.44** | **5.14** | 28.00 | 5.43 |
| gre | 21.78 | 4.05 | **18.89** | 3.30 | **18.89** | **3.06** |
| hin | 12.67 | 2.82 | **6.67** | **1.47** | 9.56 | 2.40 |
| hun | 6.67 | 1.51 | **5.33** | **1.18** | **5.33** | 1.28 |
| ice | 17.56 | 3.62 | **10.00** | 2.36 | 10.22 | **2.21** |
| kor | 52.22 | 15.88 | 46.89 | **16.78** | **43.78** | 17.50 |
| lit | 23.11 | 4.43 | **19.11** | **3.55** | 20.67 | 3.65 |
| ady | 32.00 | 7.56 | **28.00** | 6.53 | 28.44 | **6.49** |
| dut | 23.78 | 3.97 | 16.44 | 2.94 | **15.78** | **2.89** |
| jap | 9.56 | 2.07 | 7.56 | **1.79** | **7.33** | 1.86 |
| rum | 11.56 | 3.55 | **10.67** | **2.53** | 12.00 | 2.62 |
| vie | 8.44 | 1.79 | **4.67** | **1.52** | 7.56 | 2.27 |

Table 2: Results for the three baseline systems.

to the LSTM baseline and observe errors caused by underapplication of these coda-onset cluster rules. It is unsurprising then that several submissions achieved substantial gains by either romanizing hangul or decomposing it into its constituent jamo during preprocessing, since both techniques reduce the size of the input vocabulary.

The results suggest that G2P technologies are not yet language-agnostic (in the sense of Bender 2009). However, some caution is in order here: inter-language differences in word error rate may also reflect inconsistencies in the WikiPron data itself. During the task, participants reported apparent transcription inconsistencies in the Bulgarian, Georgian, and Lithuanian Wiktionary data. If these inconsistencies are due to overly-narrow allophonic transcriptions, one might suspect that they can be learned by sufficiently sophisticated sequence-to-sequence models. However, if they represent free variation, inconsistent application of the transcription guidelines, or even typographical errors, they inflate error rates and increase the risk of overfitting. In response to this, we have begun development of quality assurance software for WikiPron, including a phone-based whitelisting approach. We anticipate that manual error analysis will reveal errors in the Wiktionary data, similar to the large number of test data errors identified by Gorman et al. (2019) for the 2017 CoNLL–SIGMORPHON morphological inflection task. To encourage this sort of error analysis, for the first time in the history of the SIGMORPHON workshop, we publicly release the predictions made by all 23 submissions.[5] Finally, we plan to apply large-scale consistency-enforcing edits upstream, i.e., to Wiktionary itself.

While the baselines are somewhat naïve and lack the sophisticated data augmentation and ensembling techniques used by the top submissions, we were pleasantly surprised by the substantial reductions in error achieved by the participating teams. As mentioned above, the best submissions handily outperforms the baselines for all languages. Interestingly, this is true for the most challenging languages—like Korean, where the best submission achieves a 45% relative word error reduction over the baseline—but also for Vietnamese, the language with the lowest baseline WER; there, the best submission achieves an impressive 81% relative word error reduction.

As mentioned above, top submissions make use of techniques such as preprocessing, data augmentation, ensembling, multi-task learning (e.g., phoneme-to-grapheme conversion), and self-

[5] https://drive.google.com/drive/folders/1kdawyeI17iGC0j1Y_2dZQpK75hpShY_H?usp=sharing

|  | Best baseline |  | Best submission |  |
|---|---|---|---|---|
| arm | 14.22 | transformer | **12.22** | CLUZH |
| bul | 31.11 | LSTM | **22.22** | IMS |
| fre | 6.22 | LSTM | **5.11** | DeepSPIN-3 |
| geo | 26.44 | LSTM | **24.89** | IMS |
| gre | 18.89 | LSTM, transformer | **14.44** | CU-2, CUZ |
| hin | 6.67 | LSTM | **5.11** | CLUZH, IMS |
| hun | 5.33 | LSTM, transformer | **4.00** | CLUZH |
| ice | 10.00 | LSTM | **9.11** | CLUZH, UBCNLP-2 |
| kor | 43.78 | transformer | **24.00** | DeepSPIN-1, DeepSPIN-2 |
| lit | 19.11 | LSTM | **18.67** | CLUZH |
| ady | 28.00 | LSTM | **24.67** | DeepSPIN-4 |
| dut | 16.44 | transformer | **13.56** | IMS |
| jap | 7.33 | transformer | **4.89** | DeepSPIN-4 |
| rum | 10.67 | LSTM | **9.78** | DeepSPIN-3 |
| vie | 4.67 | LSTM | **0.89** | DeepSPIN-2 |

Table 3: The best baseline(s) and submission(s) WERs for each language.

|  | WER | PER |
|---|---|---|
| Pair n-gram | 22.00 | 4.92 |
| LSTM | 16.84 | 3.99 |
| Transformer | 17.51 | 4.30 |
| CLUZH | 14.13 | 2.82 |
| CU-1 | 14.52 | 3.24 |
| CUZ | 20.87 | 5.23 |
| DeepSPIN-3 | 14.15 | 2.92 |
| IMS | **13.81** | **2.76** |
| NSU-1 | 63.56 | 20.76 |
| UA-2 | 17.47 | 4.26 |
| UBCNLP-1 | 14.99 | 3.30 |
| UZH-3 | 16.34 | 3.27 |

Table 4: Macro-averaged results for the baselines and the best submission from each team.

training. These techniques are commonly used in shared tasks and are essentially task-agnostic. However, we were surprised that few teams made use of task-specific resources such as the PHOIBLE phonemic inventories and feature specifications (Moran and McCloy 2019) or rule-based G2P systems like Epitran (Mortensen et al. 2018). Nor do any of the submissions make use of morphological analyzers or lexicons, which were found to be helpful in earlier work (e.g., Coker et al. 1990, Demberg et al. 2007). We speculate that such resources might further improve performance. Finally we note that submissions make use of unsupervised tokenization techniques such as byte-pair encoding (Schuster and Nakajima 2012).

Finally, we note that several participants expressed interest in a low-resource version of this challenge, and two teams simulated a low-resource setting. We leave the design of a low-resource task for future work.

## 9 Conclusion

SIGMORPHON, under whose auspices this task was conducted, was once known as SIGPHON and was primarily focused on computational phonetics and phonology. The shared task on multilingual grapheme-to-phoneme conversion, a uniquely phonological problem, thus represents something of a return to the roots of this special interest group. In this task, nine teams submitted 23 G2P systems for fifteen languages and achieved substantial improvements over the provided baselines. The results suggest many directions for improving G2P systems and the pronunciation dictionaries used to train them.

## Acknowledgements

We thank the contributors to WikiPron—especially Jackson Lee—and to Wiktionary itself. We also thank all those who met for lunch at Florence's *mercato centrale* to plan this task.

## References

Emily M. Bender. 2009. Linguistically naïve != language independent: why NLP needs linguistic typology. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pages 26–32, Athens.

Maximilian Bisani and Hermann Ney. 2008. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5):434–451.

Johannes Bjerva and Isabella Augenstein. 2018. From phonology to syntax: unsupervised linguistic typology at different levels with language embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 907–916, New Orleans.

Alan W. Black, Kevin Lenzo, and Vincent Pagel. 1998. Issues in building general letter to sound rules. In *Third ESCA/COCOSDA Workshop on Speech Synthesis*, pages 77–80, Jenolan, Australia.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. 2016. Listen, attend and spell: a neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4960–4964, Shanghai.

Chung-Cheng Chiu, Tara N. Sainath, Yonghei Wu, Rohit Prabhavalkar, Patrick Nyugen, Zhifeng Chen, Anjuli Kannan, Ron J. Weiss, Kanishka Rao, Ekaterina Gonina, Navdeep Jaitly, Bo Li, Chorowski, and Michiel Bacchiani. 2018. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4774–4778, Calgary.

Cecil H. Coker, Kenneth W. Church, and Mark Y. Liberman. 1990. Morphology and rhyming: two powerful alternatives to letter-to-sound rules for speech synthesis. In *ESCA Workshop on Speech Synthesis*, pages 83–86, Autrans, France.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017. CoNLL–SIGMORPHON 2017 shared task: universal morphological reinflection in 52 languages. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 1–30, Vancouver.

John DeFrancis. 1989. *Visible speech: the diverse oneness of writing systems*. University of Hawaii Press.

Vera Demberg, Helmut Schmid, and Gregor Möhler. 2007. Phonological constraints and morphological preprocessing for grapheme-to-phoneme conversion. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 96–103, Prague.

Omnia ElSaadany and Benjamin Suter. 2020. Grapheme-to-phoneme conversion with a multilingual transformer model: a contribution to the SIGMORPHON 2020 Shared Task 1. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Seattle.

Daan van Esch, Mason Chua, and Kanishka Rao. 2016. Predicting pronunciations with syllabification and stress with recurrent neural networks. In *INTERSPEECH 2016: 17th Annual Conference of the International Speech Communication Association*, pages 2841–2845, San Francisco.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the Leipzig Corpora Collection: from 100 to 200 languages. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 759–765, Istanbul.

Kyle Gorman. 2016. Pynini: a Python library for weighted finite-state grammar compilation. In *Proceedings of the SIGFSM Workshop on Statistical NLP and Weighted Automata*, pages 75–80, Berlin.

Kyle Gorman, Gleb Mazovetskiy, and Vitaly Nikolaev. 2018. Improving homograph disambiguation with supervised machine learning. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, pages 1349–1352, Miyazaki, Japan.

Kyle Gorman, Arya D. McCarthy, Ryan Cotterell, Ekaterina Vylomova, Miikka Silfverberg, and Magdalena Markowska. 2019. Weird inflects but OK: making sense of morphological generation errors. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 140–151, Hong Kong.

Bradley Hauer, Amir Ahmad Habibi, Yixing Luan, Arnob Mallik, and Grzegorz Kondrak. 2020. Low-resource G2P and P2G conversion with synthetic training data. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Seattle.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: a method for stochastic optimization. In *3rd International Conference on Learning Representations: Conference Track Proceedings*, San Diego.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Arya D. McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. UniMorph 2.0: universal morphology. In *Proceedings of the 11th Language Resources and Evaluation Conference*, pages 1868–1873, Miyazaki, Japan.

Gari K. Ledyard. 1966. *The Korean language reform of 1446: the origin, background, and early history of the Korean alphabet*. Ph.D. thesis, University of California, Berkeley.

Jackson L. Lee, Lucas F.E. Ashby, M. Elizabeth Garza, Yeonju Lee-Sikka, Sean Miller, Alan Wong, Arya D. McCarthy, and Kyle Gorman. 2020. Massively multilingual pronunciation mining with WikiPron. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4216–4221, Marseille.

Patrick Lehnen, Alexandre Allauzen, Thomas Lavergne, François Yvon, Stefan Hahn, and Hermann Ney. 2013. Structure learning in hidden conditional random fields for grapheme-to-phoneme conversion. In *INTERSPEECH 2013: 14th Annual Conference of the International Speech Communication Association*, pages 2326–2330, Lyon.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon.

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

Peter Makarov and Simon Clematide. 2020. CLUZH at 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*, Seattle.

Arya D. McCarthy, Liezl Puzon, and Juan Pino. 2020. SkinAugment: auto-encoding speaker conversions for automatic speech translation. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 7924–7928, Barcelona.

Sabrina J. Mielke, Ryan Cotterell, Kyle Gorman, Brian Roark, and Jason Eisner. 2019. What kind of language is hard to language-model? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4975–4989, Florence.

Steven Moran and Michael Cysouw. 2018. *The Unicode cookbook for linguists: managing writing systems using orthography profiles*. Language Science Press.

Steven Moran and Daniel McCloy. 2019. *PHOIBLE 2.0*. Max Planck Institute for the Science of Human History.

David R. Mortensen, Siddharth Dalmia, and Patrick Littell. 2018. Epitran: precision G2P for many languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, pages 2710–2714, Miyazaki, Japan.

Garrett Nicolai, Saeed Najafi, and Grzegorz Kondrak. 2018. String transduction with target language models and insertion handling. In *Proceedings of the Fifteenth Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 43–53, Brussels.

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

Rolf Noyer. 1998. Vietnamese 'morphology' and the definition of word. *University of Pennsylvania Working Papers in Linguistics*, 5(2):65–89.

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.

Ben Peters, Jon Dehdari, and Josef van Genabith. 2017. Massively multilingual neural grapheme-to-phoneme conversion. In *Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems*, pages 19–26, Copenhagen.

Ben Peters and André F. T. Martins. 2020. One-size-fits-all multilingual models. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Seattle.

Ben Peters, Vlad Niculae, and André F.T. Martins. 2019. Sparse sequence-to-sequence models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1504–1519, Florence.

Juan Pino, Liezl Puzon, Jiatao Gu, Xutai Ma, Arya D. McCarthy, and Deepak Gopinath. 2019. Harnessing indirect training data for end-to-end automatic speech translation: tricks of the trade. In *16th International Workshop on Spoken Language Translation*.

Nikhil Prabhu and Katharina Kann. 2020. Frustratingly easy multilingual grapheme-to-phoneme conversion. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Seattle.

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*, pages 4225–4229, Brisbane.

Brian Roark, Richard Sproat, Cyril Allauzen, Michael Riley, Jeffrey Sorensen, and Terry Tai. 2012. The OpenGrm open-source finite-state grammar software libraries. In *Proceedings of the ACL 2012 System Demonstrations*, pages 61–66, Jeju Island, Korea.

Henry Rogers. 2005. *Writing systems: a linguistic approach*. Blackwell.

Zach Ryan and Mans Hulden. 2020. Data augmentation for transformer-based G2P. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Seattle.

Avedis Sanjian. 1996. The Armenian alphabet. In Peter T. Daniels and Williams Bright, editors, *The World's Writing Systems*, pages 356–357. Oxford University Press.

Michael Schuster and Kaisuke Nakajima. 2012. Japanese and Korean voice search. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5149–5152, Kyoto.

Jose Sotelo, Soroush Mehri, Kundan Kumar, João Felipe Santos, Kyle Kastner, Aaron C. Courville, and Yoshua Bengio. 2017. Char2Wav: end-to-end speech synthesis. In *5th International Conference on Learning Representations: Workshop Track Proceedings*, Toulon.

Richard Sproat. 1996. Multilingual text analysis for text-to-speech synthesis. *Natural Language Engineering*, 2(4):369–380.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, Las Vegas.

Paul Taylor. 2005. Hidden Markov models for grapheme to phoneme conversion. In *INTERSPEECH 2005–EUROSPEECH 2005: 9th European Conference on Speech Communication and Technology*, pages 1973–1976, Lisbon.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uzskoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008, Long Beach.

Kaili Vesik, Muhammad Abdul-Mageed, and Miikka Silfverberg. 2020. One model to pronounce them all: multilingual grapheme-to-phoneme conversion with a transformer ensemble. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Seattle.

Robert A. Wagner and Michael J. Fischer. 1974. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173.

Oliver Watts, Adriana Stan, Robert Clark, Yoshitaka Mamiya, Mircea Giurgiu, Junichi Yamagishi, and Simon King. 2013. Unsupervised and lightly-supervised learning for rapid construction of TTS systems in multiple languages from 'found' data: evaluation and analysis. In *Eighth ISCA Workshop on Speech Synthesis*, pages 101–106, Barcelona.

Ke Wu, Cyril Allauzen, Keith Hall, Michael Riley, and Brian Roark. 2014. Encoding linear models as weighted finite-state transducers. In *INTERSPEECH 2014: 15th Annual Conference of the International Speech Communication Association*, pages 1258–1262, Singapore.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2020. Applying the transformer to character-level transduction. `arXiv:2005.10213`.

Kaisheng Yao and Geoffrey Zweig. 2015. Sequence-to-sequence neural net models for grapheme-to-phoneme conversion. In *INTERSPEECH 2015: 16th Annual Conference of the International Speech Communication Association*, pages 3330–3334, Dresden.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, MA.

Sevinj Yolchuyeva, Géza Németh, and Bálint Gyires-Tóth. 2019. Transformer based grapheme-to-phoneme conversion. In *INTERSPEECH 2019: 20th Annual Conference of the International Speech Communication Association*, pages 2095–2099, Graz, Austria.

Xiang Yu, Ngoc Thang Vu, and Jonas Kuhn. 2020. Ensemble self-training for low-resource languages: grapheme-to-phoneme conversion and morphological inflection. In *Proceedings of the 17th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Seattle.