Leveraging Principal Parts for Morphological Inflection

Ling Liu and **Mans Hulden**
Department of Linguistics
University of Colorado
first.last@colorado.edu

Abstract

This paper presents the submission by the CU Ling team from the University of Colorado to SIGMORPHON 2020 shared task 0 on morphological inflection. The task is to generate the target inflected word form given a lemma form and a target morphosyntactic description. Our system uses the Transformer architecture. Our overall approach is to treat the morphological inflection task as a paradigm cell filling problem and to design the system to leverage principal parts information indirectly for better morphological inflection when the training data is limited. We train one model for each language separately without external data. The overall average performance of our submission ranks the first in both average accuracy and Levenshtein distance from the gold inflection among all submissions including those using external resources.

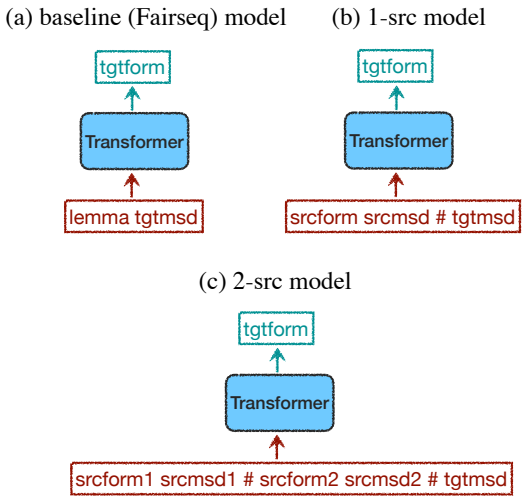

Figure 1: Illustration of general model architectures. All three of our models use the Transformer architecture for inflection. They are different from each other by the input to the Transformer model.

1 Introduction

The task of morphological inflection is to generate a target inflected word form (henceforth *tgtform*) given a lemma form (henceforth *lemma*) and a target morphosyntactic description (henceforth *tgtmsd*). In the SIGMORPHON 2020 shared task 0 on morphological inflection (Vylomova et al., 2020) and previous years' SIGMORPHON shared tasks on morphological inflection (Cotterell et al., 2016, 2017a, 2018; McCarthy et al., 2019), the training data is provided in the format of tab-separated lemma-tgtmsd-tgtform triples, and participating systems are expected to predict the missing target forms in the test data released shortly before prediction submission.

The sequence-to-sequence (henceforth *seq2seq*) architecture has been very successful in dealing with morphological inflection, especially when there are abundant labeled data for training. The accuracies and Levenshtein distances on the development data inflected by 9 baseline models are provided for the 45 typologically and genealogically diversified development languages: a non-neural model based on lemma-tgtform alignment and transformation, a per-language Transformer model, a per-language-family Transformer model, a per-language Transformer model with data augmentation, a per-language-family Transformer model with data augmentation, LSTM seq2seq models with exact hard monotonic attention (Wu and Cotterell, 2019) trained per language, per language family, per language with data augmentation, and per language family with data augmentation respectively. The data augmentation method used by the baseline models is from Anastasopoulos and Neubig (2019). The baseline numbers indicate that the Transformer model for character-level transduction (Wu et al., 2020) is very competitive, achieving the highest average accuracy and lowest average edit distance and best performance on most lan-

guages (33 out of 45) when the model is trained per language. Therefore, we adopt the Transformer architecture (Vaswani et al., 2017) for all three of our models (see Figure 1) which are different from each other by the input and output to the Transformer model, as will be presented in section 3.

Though not explicitly organized as a paradigm cell filling problem (PCFP) (Ackerman et al., 2009) task, the shared task is closely related to and can largely be seen as a computational instance of it (Malouf, 2016, 2017; Cotterell et al., 2017a; Silfverberg et al., 2018; Silfverberg and Hulden, 2018), where some slots are given in the paradigms as training data and others are to be inflected as development data or test data.[1] The data format of the shared task privileges the lemma as the source form (henceforth *srcform*) which all tgtforms are inflected from. However, the lemma form may not be the only and the most informative srcform to inflect all other slots from in the same paradigm. Morphologists refer to a lexeme's principal parts (Finkel and Stump, 2007) as the minimum subset of paradigm slots which, if known, provide all the information needed to generate the other slots in its paradigm. The principal parts which best predict an inflected form in a lexeme's paradigm do not necessarily include the lemma, and more than one of the principal parts may be needed to generate an inflected form reliably (see examples in Table 1 analyzed in section 3.2). Considering this, we convert the shared task of morphological inflection to the paradigm cell filling problem, and incorporate the principal part intuition into the inflection system. Our approaches achieve better or equally good performance compared to the official baselines for most (19 out of 24) relatively low-resource languages we experimented with.

To generate inflected forms for the test data for submission, our system uses the same input-output format as the baselines for high-resource languages, and includes two slightly different approaches of leveraging principal parts information for low-resource languages. The evaluation results indicate that the Transformer model augmented with principal parts information can handle morphological inflection very well for typologically and genealogically diverse languages, whether it has been tuned on the language or not, even when the training data is limited.

[1] This does not hold perfectly—some languages have held-out data that come from paradigms where no form is ever witnessed in the training data, but these are a minority. We overcome this problem by adding an additional slot (tagged as `POS;CANONICAL`) for the lemma in the paradigm.

2 Task and data description

The SIGMORPHON 2020 shared task 0 (Vylomova et al., 2020) is a typical morphological inflection task. Compared to previous years' SIGMORPHON shared tasks on morphological inflection, this year's task highlights the distinction between development languages and surprise languages and the inflection model's ability to generalize to new languages that may be genetically related or unrelated to the languages according to which it is developed. In the development phase, 45 languages from 5 language families were provided, and these languages are development languages. In the generalization phase, 45 surprise languages from 16 language families were released. In the evaluation phase, test data include both development languages and surprise languages.

Deviating from previous years' tasks, this year's task did not feature different (low/medium/high) data settings for the languages (Cotterell et al., 2017a, 2018) or manipulate the data size of genetically related language pairs (McCarthy et al., 2019). Instead, each language comes with different amount of training, development and test data, corresponding to the reality of data availability for the language. Of the total 90 languages from 18 language families, 44 have 5,000 or more lemma-tgtmsd-tgtform training triples and 46 have fewer than 5,000. Of the 45 development languages, 24 have fewer than 5,000 training examples. In this paper, we refer to languages with 5,000 or more training triples as high-resource and those with fewer than 5,000 training triples as low-resource.

3 System description

All our models use the self-attention Transformer architecture (Vaswani et al., 2017) as implemented in the Fairseq (Ott et al., 2019) tool, a PyTorch-based sequence modeling toolkit. Both the encoder and decoder have 4 layers with 4 attention heads, an embedding size of 256 and hidden layer size of 1024. Models are trained with the Adam algorithm (Kingma and Ba, 2014) for optimization with an initial learning rate of 0.001, a batch size of 400, 0.1 label smoothing, the gradient clip threshold as 1.0, and 4,000 warmup updates. The models are trained for a maximum of 20,000 or 30,000 optimizer updates depending on the amount of input-

ID	MSD	Lexeme1	Lexeme2	Lexeme3	Lexeme4	Lexeme5
1	V;CANONICAL	pahinga	bayad	pukpok	linlang	gáling
2	V;AGFOC;LGSPEC1	–	magbabayad	manumukpok	lanlilinlang	gagáling
3	V;IPFV;AGFOC	?	nagbabayad	namumukpok	nanlilinlang	gumagáling
4	V;IPFV;PFOC	∗	binabayaran	pinupukpok	nililinlang	iginagáling
5	V;NFIN	pahinga	bayad	pukpok	linlang	gáling
6	V;PFOC;LGSPEC1	∗	babayaran	pupukpukin	?	igagáling
7	V;PFV;AGFOC	nagpahinga	nagbayad	namukpok	nanlinlang	gumáling
8	V;PFV;PFOC	∗	binayaran	pinukpok	nilinlang	igináling

Table 1: Example of reconstructed paradigms from Tagalog data. – are slots in the development set, ? are slots in the test set, ∗ are slots which didn't appear in the shared task data, and other slots which are filled with inflected forms are slots in the training set.

output tuples for training, with checkpoints saved every 10 epochs. The checkpoint with the smallest loss and the last checkpoint are also saved. The model with the best parameters was selected from all the saved checkpoints based on the accuracy on the development data. Beam search is used at decoding time with a beam width of 5.

Our submission is an ensemble of predictions from three types of models: baseline (Fairseq), 1-src, and 2-src. These three types of models have identical model architecture for inflection and are different from each other in the input and output. As the varied baseline results trained per language family provided by the organizers did not show consistent improvements compared to training languages separately, we train all the models per language without using external resources. We made our code publicly available.[2]

3.1 Baseline (Fairseq) model

The baseline (Fairseq) model (see Figure 1(a)) is very similar to the unaugmented per-language Transformer baseline (Wu et al., 2020) provided by the shared task organizers, except that the Fairseq implementation is used and that beam search rather than greedy search is used at decoding time. The inputs to this model are the individual characters of the lemma followed by the individual subtags of the tgtmsd. For example, for the English training triple `(look, looks, V;SG;3;PRS)`, the input to the model is `l o o k V SG 3 PRS` and the gold standard output is `l o o k s`. Our submissions for languages with 5,000 or more training triples are generated with this model. The model is trained for a maximum of 20,000 optimizer updates for languages with 5,000 to 20,000 training triples, and for a maximum of 30,000 updates for languages with over 20,000 training triples.

3.2 Principal parts of a paradigm

The classical notion of "principal parts of a paradigm" is the minimal subset of paradigm slots that provides enough information according to which the inflection forms for other slots in the same paradigm can be correctly generated (Finkel and Stump, 2007). The principal part may be different for different slots in the same paradigm, and more than one principal part may be necessary in order to inflect for some slots correctly. For example, for each Tagalog lexeme in Table 1, slots 2 and 3 are very informative source forms for each other, which are different by the first consonant, or the presence or absence of `um` in the prefix. Slot 3 can predict slot 7 very well, and slot 8 can be easily generated from slot 4. Inflection of slot 6 is the most complex in the paradigms, for which slot 4 together with the lemma, i.e. slot 1, can be informative but not sufficient. Therefore, the lemma is not always a good choice as the source to generate all other slot forms from, and we can expect the morphological inflection system to be more effective and efficient if the principal parts information is incorporated.

The 1-src model (see Figure 1(b)) and the 2-src model (see Figure 1(c)) leverage the idea of paradigm principal parts. To do this, we first reconstruct the paradigm for each lexeme in the shared task data, from which we prepare input and output data for the inflection models.

We assume that each part-of-speech (henceforth *POS*) in a language has its own set of morphosyntactic descriptions (henceforth *MSD*s), which can be obtained by collecting the tgtmsd types in the training, development and test data for the lan-

[2] https://github.com/LINGuistLIU/principal_parts_for_inflection

guage. Each slot in the paradigm of a lexeme locates an inflected word form, which can be considered a combination of a lexeme and an MSD. In this paper, slot is used to refer to both the inflected form and the corresponding MSD it locates, slot form refers to the inflected forms only, and slot MSD refers to the corresponding morphosyntactic description. If a slot contains both the MSD and the inflected form, it is a filled slot, while an empty slot needs to be filled with the corresponding inflected form. The slot MSD can be determined by the set of MSDs we collect for each POS, and we can fill in the slot if it appears in the training data and mark it if the inflected form is to be generated in the development or test data, or does not appear in the shared task data at all. In addition, the shared task data format has the first element in the triple as the lemma form, i.e. the canonical, or citation, form of the lexeme. We add an additional slot in the paradigm for the lemma form, and tag the slot as *POS*;CANONICAL where the *POS* in the tag is determined by the POS of the lemma. As a result, we create a paradigm for each lexeme in the shared task data and the reconstructed paradigm for each lexeme has at least one filled slot. Table 1 provides 5 example paradigms reconstructed from the Tagalog data, where – marks slots with tgtforms to be predicted in the development set, ? are slots in the test data and ∗ indicates slots which are not found in the shared task data,[3] and other slots which are filled with inflected word forms are data in the training set. In cases where slots have alternative forms in the data, only one form is kept. For example, there are two alternative forms for thanda V;SG;1;PRS in the Zulu training data: ngithanda and ngiyathanda, and our conversion only kept ngiyathanda.

1-src model In order to train the 1-src model, the reconstructed paradigm is organized so that each of the known slots is given as a srcform from which we predict every other known slot as the tgtform. The symbol # is inserted between the srcmsd and tgtmsd. For example, six input-output tuples (see Figure 2) are constructed from the Tagalog Lexeme1 paradigm example provided in Table 1. When only one slot is filled in the reconstructed paradigm, we make the slot predict

[3]The ∗ slots may be invalid in the language. For example, the English noun cattle does not have a single form, and the single slot would be marked by ∗ in the paradigm for the lexeme cattle reconstructed by our method.

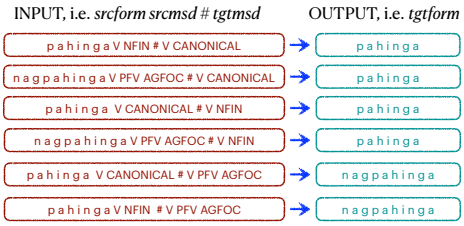

Figure 2: Input-output tuples for the 1-src model for Tagalog Lexeme1 (*pahinga* "rest") example paradigm

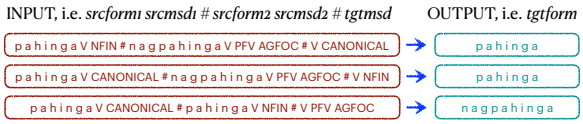

Figure 3: Input-output tuples for the 2-src model for Tagalog Lexeme1 (*pahinga* "rest") example paradigm

itself (i.e. input as *lemma POS*;CANONICAL # *POS*;CANONICAL and output as *lemma*) for training. All given srcform-srcmsd slots are used to predict the tgtform for each tgtmsd in the development and test data respectively. Consequently, for the Tagalog Lexeme1 example, each tgtsmd in the development and test sets will be predicted by three different source forms with the corresponding morphosyntactic description specified, rather than being predicted only by the lemma. This is the model we use to generate our submission predictions for 39 languages with fewer than 5,000 training triples. The languages aka, ben, cly, cre, kan, kir, kon, liv, lld, lug, nya, pus, sna, and swa are trained for 30,000 maximum updates, and other languages are trained for 20,000 maximum updates.

2-src model The 2-src model generates predictions for the remaining 7 low-resource languages (czn, frr, gsw, izh, mlt, mwf, zpv), because we only trained the 2-src model for languages with fewer than 2,000 training examples due to time constraints and because the 2-src model generates significantly better predictions for these 7 languages on the development data than the 1-src model. During training, the inputs to the 2-src model are all possible known two-slot combinations followed by the MSD for the slot to be filled; the output is the known inflected form for the target slot. The symbol # is inserted between the first srcmsd and the second srcform as well as between the second srcmsd and tgtmsd. For example, three input-output tuples (see Figure 3) are constructed from the Tagalog Lexeme1 example. When only one slot form is given in the paradigm, the given slot is made

to predict itself by taking as input the *lemma* and *POS*;CANONICAL repeated twice together with the tgtmsd as *POS*;CANONICAL, and the output is the *lemma* form. When only two slots are filled in the paradigm, each slot form is treated as the tgtform and the other slot is repeated twice together with the MSD for the slot to be predicted as input to the model. For the development and test data, every two-slot combination of given slots is used as input to predict the tgtform corresponding to the tgtmsd. Therefore, each test and development tgtmsd in the Tagalog Lexeme1 example will be predicted by three different inputs, respectively.

Prediction selection Because of the input and output construction for the 1-src and 2-src models, each tgtmsd may be predicted multiple times by different inputs which may generate more than one inflected form for the same tgtmsd. Two mechanisms are employed to pick the best prediction, both of which implicitly employ the principal parts intuition. The first mechanism is to select the prediction generated by most inputs, i.e. by majority vote for predictions by different inputs. The second mechanism is to select the prediction which gets the highest average log-likelihood, i.e. by averaging the scores for each prediction by different inputs. The intuition behind this mechanism is that the most informative source slots should be most confident about the inflection for the target slot. Unless the majority vote mechanism produces significantly higher accuracy on the development data for the language, the prediction with the highest average log-likelihood is selected as the final prediction for the target slot.

4 Experiments

Considering the time constraints and the already strong performance of the baseline models—especially when training data is abundant—we focused our experiments on the 24 low-resource development languages in the development phase, for which we attempted to augment the Transformer model for inflection by reorganizing the data into paradigms and making use of the principal parts morphology idea in different ways.

In addition to the 1-src and 2-src models described in section 3.2, other approaches we experimented with included 2-random-src, 3-random-src and 4-random-src models where we randomly pick two, three or four given slots as input which will be translated to the tgtform corresponding to the tgtmsd, as well as all-src-tgtform and all-src-allform models, where the concatenation of all given slots followed by the tgtmsd are input to the inflection model which predicts the corresponding tgtform or all srcforms and the tgtform. Though these models produced better performance for one or two languages that we experimented with initially, we did not see consistent performance improvement proportional to the increasing model complexity over the 1-src and 2-src models. We also experimented with warming up the 1-src model with an additional copying task following the practice suggested by Anastasopoulos and Neubig (2019), but did not see improvements. Therefore, we focused exclusively on the 1-src and 2-src models after initial experiments.

Further experiments with the 1-src model were conducted on the 24 development languages with fewer than 5,000 training triples, and further experiments with the 2-src model were conducted on the 17 development languages, each of which has fewer than 2,000 training triples. The performance of the two selected models will be presented and discussed in the next section.

5 Results and discussion

The average inflection accuracy of development data for the 24 languages by the 1-src model is 91.72%, which is 1.3% higher than the unaugmented per-language Transformer baseline and 0.55% higher than the best performance of all baseline models. The 1-src model achieved higher or equal accuracy on 18 languages compared to the unaugmented per-language Transformer baseline and 17 languages compared to the best performance of all baselines. The 2-src models for the 17 languages we experimented with achieve an average accuracy of 91.63% and their performance on 7 languages (czn, frr, gsw, izh, mlt, mwf, zpv) is better than the 1-src model.

Figure 4 plots the difference in the accuracy on the development set for each language by the 1-src for 2-src model from that by the unaugmented per-language Transformer baseline. Figures 4(a) and 4(c) depict the relationship between this difference and the number of training triples. Figures 4(b) and 4(d) show the relationship between this difference and the completeness of the paradigms seen in training. The filled percentage of each paradigm is calculated by dividing the number of given slots by the number of all slots in the paradigm, and the

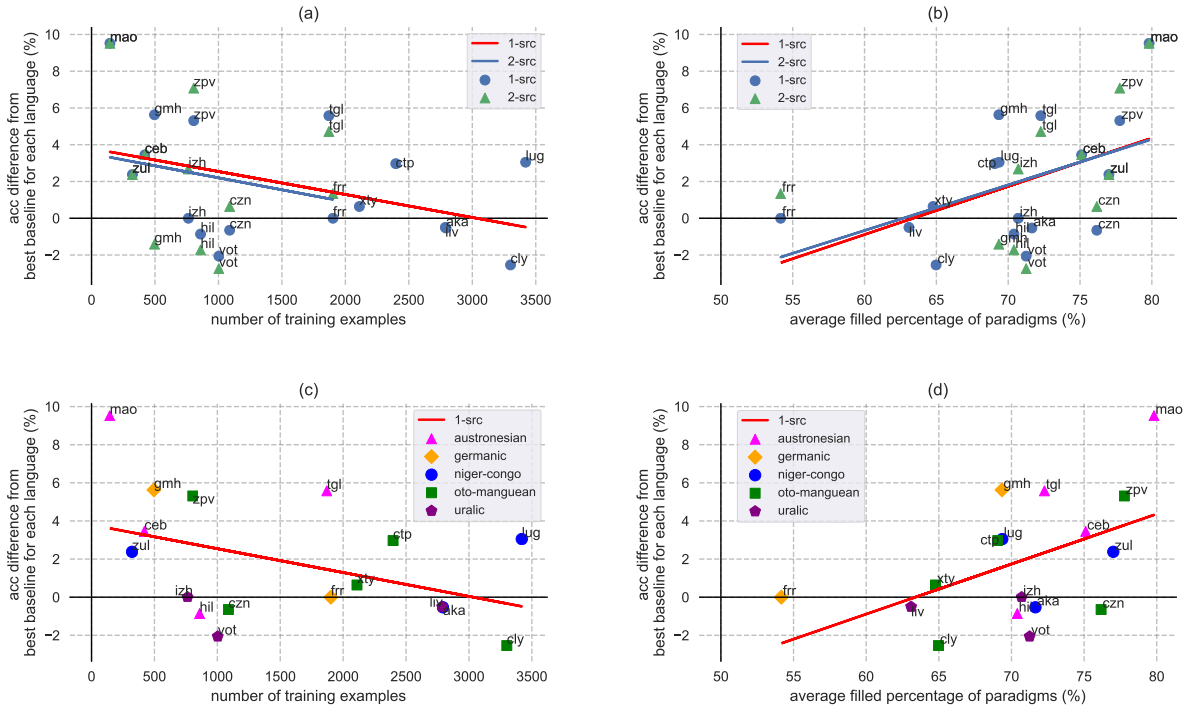

Figure 4: Scatter plots with trend lines for the difference in accuracy between the 1-src or 2-src model and the **per-language Transformer baseline without data augmentation** on low-resource dev languages: (a) 1-src vs 2-src: Size vs Δacc_1 or Δacc_2, (b) 1-src vs 2-src: Percentage vs Δacc_1 or Δacc_2, (c) 1-src and genealogy: Size vs Δacc_1, (d) 1-src and genealogy: Percentage vs Δacc_1. (Size: training data size, Percentage: average percentage of slots per paradigm in training data, $\Delta acc_1^i = acc_{\text{1-src}}^i - acc_{\text{per-lang-unaug-transformer-baseline}}^i$, $\Delta acc_2^j = acc_{\text{2-src}}^j - acc_{\text{per-lang-unaug-transformer-baseline}}^j$)

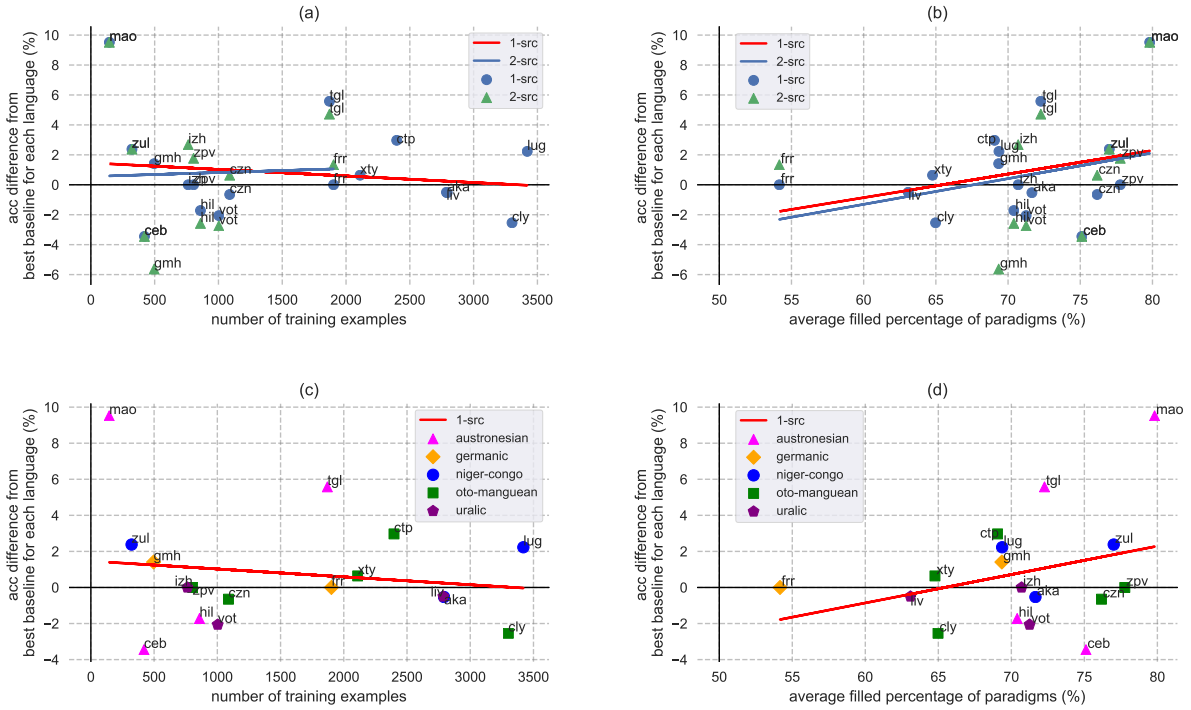

Figure 5: Scatter plots with trend lines for the difference in accuracy between the 1-src or 2-src model and the **best performance of all baselines** on low-resource dev languages: (a) 1-src vs 2-src: Size vs Δacc_3 or Δacc_4, (b) 1-src vs 2-src: Percentage vs Δacc_3 or Δacc_4, (c) 1-src and genealogy: Size vs Δacc_3, (d) 1-src and genealogy: Percentage vs Δacc_3. (Size: training data size, Percentage: average percentage of slots per paradigm in training data, $\Delta acc_3^i = acc_{\text{1-src}}^i - acc_{\text{best-baseline}}^i$, $\Delta acc_4^j = acc_{\text{2-src}}^j - acc_{\text{best-baseline}}^j$)

paradigm completion rate of a language is calculated by taking the average of the filled percentages of all the reconstructed paradigms. For instance, the completion rate of the Tagalog Lexeme1 paradigm is 37.5%, and the average completion rate of all the Tagalog example paradigms in Table 1 is 85%. The low-resource development languages have average completion rates between 54.16% (frr) and 79.81% (mao). Figure 5 plots the same relationships, but the difference is between the 1-src or 2-src model and the best performance of all baseline models. Languages for which both the baseline models and our models achieve 100% accuracy are excluded from the plots, because such languages have the potential to skew the performance comparison. Such languages include one Austronesian language: mlg and six Niger-Congo langauges: gaa, kon, lin, nya, sot and swa.

Model performance and training data size The improvements by the 1-src and 2-src models over the unaugmented Transformer baseline trained per language show the same tendency with relationship to the training data size: The more training data there is available, the less advantage our models have. This is shown in Figure 4(a). The baseline model begins to catch up with these improvements as is shown in Figure 5(a), where the 1-src model accuracy still has a decreasing trend as the training data increases while the 2-src model accuracies turn into a slightly increasing trend.

Model performance and paradigm completion rate The good performance of our models relies on the high completion rate of paradigms. The performance for both the 1-src and 2-src models tends to be better if the reconstructed paradigm contains a higher proportion of known slots. This is true whether our models are compared to the single unaugmented per-language Transformer baseline model or to the ensemble of all baseline models. This relationship is illustrated in Figure 4(b) and Figure 5(b). An extreme case of a low paradigm completion rate in the shared task languages is Ludic, where only 5.64% of the slots are known, and our best model for this language is the 1-src approach with average score selection, which generates an accuracy of 48.78% on the development data. This relationship supports the use of principal parts for morphological inflection, because given a random sampling, the more complete a paradigm is, the more likely it is that the principal parts are included in the paradigm.

Model performance and genealogy Subplots (c) and (d) in Figure 4, and subplots (c) and (d) in Figure 5, show the performance of the 1-src model on languages with language family information. Uralic languages are challenging to our models. This is to be expected from the fact that Uralic languages usually have large inflection paradigms and therefore tend to have more incomplete slots on average given the same amount of data, and may hence be missing a principal part.

6 Related work

Morphological inflection is one of the natural language processing tasks which achieve great improvement by applying neural network models, especially sequence to sequence models, which initially outperformed other approaches by a large margin on high-resource languages (Cotterell et al., 2016; Kann and Schütze, 2016; Aharoni et al., 2016) and have been improved and augmented later to achieve state-of-the-art performance on low-resource languages as well (Aharoni and Goldberg, 2017; Cotterell et al., 2017a; Makarov and Clematide, 2018; Wu et al., 2018; Cotterell et al., 2018; Wu and Cotterell, 2019; McCarthy et al., 2019; Anastasopoulos and Neubig, 2019).

Subtask 2 of the CoNLL-SIGMORPHON 2017 shared task (Cotterell et al., 2017a) was about paradigm cell filling, and received submissions of neural network systems (Kann and Schütze, 2017; Silfverberg et al., 2017). There is also other work which targets the paradigm cell filling problem (Cotterell et al., 2017b; Silfverberg et al., 2018; Silfverberg and Hulden, 2018). Cotterell et al. (2017b) models the principal parts idea with graphical models to generate all the missing slots in paradigms. Our 1-src model has an input-output format similar to Silfverberg and Hulden (2018). Our work is also closely related to Kann et al. (2017) on multi-source inflection which is also motivated by a principal parts analysis. Cotterell et al. (2019) use an explicit neural model that organizes paradigm slots in their most predictable order to investigate measures of morphological complexity, an instantiation of the principal parts idea in another context.

7 Conclusion

We have presented the system for our submission to the SIGMORPHON 2020 shared task 0 on mor-

phological inflection. It achieved the highest average accuracy and smallest average Levenshtein distance across all the 90 languages from 18 language families. The standard deviation of our submission is the lowest for accuracy and the second lowest (0.004 higher than the lowest) for edit distance.

Our work indicates that the self-attention Transformer architecture can perform well for the morphological inflection task for a genealogically and typologically diverse group of languages. The architecture has a strong generalization ability and can inflect new languages as effectively as the languages it is tuned on. We augment the Transformer model by converting the morphological inflection task to the paradigm cell filling problem and leveraging the principal parts of paradigms in indirect ways, which turns out to be helpful, especially when the training data is limited and the reconstructed paradigms have a high completion rate. Our primary strategy to incorporate principal parts information in this work is to use each given slot in the reconstructed paradigm to predict the target form and select the final prediction from predictions generated by different slots by highest average score or majority vote. Another strategy is to use all possible two-slot combinations to predict the target form.

According to principal parts morphology, the number of principal parts may vary between paradigms and languages, and different slots may require different numbers of principal parts to inflect correctly, indicating that uniformly using every slot individually or every two-slot combination may not always be the best choice. Future work is needed to explore how to use principal parts information more effectively, perhaps tuning the number and choice of forms on a per-language basis or developing strategies to explicitly determine principal parts for the paradigms.

References

Farrell Ackerman, James P. Blevins, and Robert Malouf. 2009. Parts and wholes: Implicative patterns in inflectional paradigms. In James P. Blevins and Juliette Blevins, editors, *Analogy in grammar: Form and acquisition*, pages 54–82. Oxford University Press.

Roee Aharoni and Yoav Goldberg. 2017. Morphological inflection generation with hard monotonic attention. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2004–2015, Vancouver, Canada. Association for Computational Linguistics.

Roee Aharoni, Yoav Goldberg, and Yonatan Belinkov. 2016. Improving sequence to sequence learning for morphological inflection generation: The BIU-MIT systems for the SIGMORPHON 2016 shared task for morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 41–48, Berlin, Germany. Association for Computational Linguistics.

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.

Ryan Cotterell, Christo Kirov, Mans Hulden, and Jason Eisner. 2019. On the complexity and typology of inflectional morphological systems. *Transactions of the Association for Computational Linguistics*, 7:327–342.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sebastian Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick Xia, Manaal Faruqui, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2017a. 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. 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.

Ryan Cotterell, John Sylak-Glassman, and Christo Kirov. 2017b. Neural graphical models over strings for principal parts morphological paradigm completion. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages

759–765, Valencia, Spain. Association for Computational Linguistics.

Raphael Finkel and Gregory Stump. 2007. Principal parts and morphological typology. *Morphology*, 17(1):39–75.

Katharina Kann, Ryan Cotterell, and Hinrich Schütze. 2017. Neural multi-source morphological reinflection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 514–524, Valencia, Spain. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2016. MED: The LMU system for the SIGMORPHON 2016 shared task on morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 62–70, Berlin, Germany. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2017. The LMU system for the CoNLL-SIGMORPHON 2017 shared task on universal morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 40–48, Vancouver. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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, Belgium. Association for Computational Linguistics.

Robert Malouf. 2016. Generating morphological paradigms with a recurrent neural network. *San Diego Linguistic Papers*.

Robert Malouf. 2017. Abstractive morphological learning with a recurrent neural network. *Morphology*, 27(4):431–458.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sebastian J. Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244, Florence, Italy. Association for Computational Linguistics.

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.

Miikka Silfverberg and Mans Hulden. 2018. An encoder-decoder approach to the paradigm cell filling problem. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2883–2889, Brussels, Belgium. Association for Computational Linguistics.

Miikka Silfverberg, Ling Liu, and Mans Hulden. 2018. A computational model for the linguistic notion of morphological paradigm. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1615–1626, Santa Fe, New Mexico, USA. 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.

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 and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1530–1537, Florence, Italy. Association for Computational Linguistics.

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

Shijie Wu, Pamela Shapiro, and Ryan Cotterell. 2018. Hard non-monotonic attention for character-level transduction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4425–4438, Brussels, Belgium. Association for Computational Linguistics.