Cross-lingual morphological inflection with explicit alignment

Çağrı Çöltekin
University of Tübingen
Department of Linguistics
ccoltekin@sfs.uni-tuebingen.de

Abstract

This paper describes two related systems for cross-lingual morphological inflection for SIGMORPHON 2019 Shared Task participation. Both sets of results submitted to the shared task for evaluation are obtained using a simple approach of predicting transducer actions based on initial alignments on the training set, where cross-lingual transfer is limited to only using the high-resource language data as additional training set. The performance of the system does not reach the performance of the top two systems in the competition. However, we show that results can be improved with further tuning. We also present further analyses showing that the cross-lingual gain is rather modest.

1 Introduction

Morphological inflection generation is the task of generating a word based on its lemma and morphological features. For example, given the German lemma *aufgeben* 'to give up' and the morphological tags `{V.PTCP, PST}`, the task is to predict the inflected form *aufgegeben* (morphological tags are described in McCarthy et al., 2019; Kirov et al., 2018). Traditionally, finite-state methods (Koskenniemi, 1985) are used for morphological generation (and analysis). Since such systems typically require man-months of expert work, and difficult to maintain and adapt to changes in the language, data driven approaches to inflection generation have recently become popular (Durrett and DeNero, 2013; Nicolai et al., 2015; Ahlberg et al., 2015; Faruqui et al., 2016). The task is further popularized by the past three SIGMORPHON morphological (re)inflection shared tasks (Cotterell et al., 2016, 2017, 2018). The primary focus of the task tackled in this paper, the task 1 of the present SIGMORPHON shared task (McCarthy et al., 2019), is the cross-lingual transfer learning of the inflection generation.

The dominant approach to morphological inflection has been sequence-to-sequence neural networks with attention (e.g., Kann and Schütze, 2016; Makarov et al., 2017; Makarov and Clematide, 2018). Furthermore, there seems to be a shift from soft attention models towards models with monotonic attention (Ahlberg et al., 2015; Makarov and Clematide, 2018; Wu and Cotterell, 2019), which indicate that the predictions of the decoder benefit most from a (short) window in the output. Although we do not use an encoder-decoder architecture, the simple systems presented here are similar to hard-monotonic attention models in the sense that they predict the transduction actions based on a window on the input and output. The method presented here is much simpler, however. The predictions are not conditioned on any hidden (continuous or discrete) state or variable.

A particular reason of interest for data-driven approaches to morphological inflection generation is to avoid the considerable amount of expert time required for building rule-based finite-state systems. This is particularly important for low-resource languages, where experts, and maybe even native speakers, are hard to come by. As past SIGMORPHON shared tasks demonstrated, however, satisfactory results in the morphological inflection task requires relatively large amount of data. The low-resource settings in earlier SIGMORPHON shared tasks often resulted in much worse accuracy compared to the high-resource settings. A potential solution to this problem, the focus of the current inflection shared task, is cross-lingual or transfer learning, which has been demonstrated to be useful a number of language processing tasks (e.g., Yarowsky et al., 2001; Faruqui and Kumar, 2015; Johnson et al., 2017; Barnes et al., 2018). In cross-lingual learn-

Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology, pages 71–79
Florence, Italy. August 2, 2019 ©2019 Association for Computational Linguistics

ing, the data or resources that exist for a related language are leveraged to improve the learning in low-resource setting. The method we use for cross-lingual learning is rather simple. We only use the (related) high-resource language as additional training data.

2 The method

The inflection systems in this study operate by predicting a number of transduction actions based on current position in the lemma, morphological tags, and the output produced so far. The general idea is similar to transition-based parsers (Yamada and Matsumoto, 2003; Nivre et al., 2004) where the aim is to predict the parsing action in a given state of the parser. The similar ideas were used in the past for morphological inflection generation as well. The system presented here is most similar to the baseline system of SIGMORPHON 2016 shared task (Cotterell et al., 2016), and also shares many aspects of the inflection generation systems that follow an align-and-transduce strategy in the earlier SIGMORPHON shared tasks (e.g., Alegria and Etxeberria, 2016; Nicolai et al., 2016; Liu and Mao, 2016). Our current models do not make use of any hidden representations, such as the parser state in transition based parsing, or hidden representations learned in a recurrent neural network.

2.1 Alignment

During training, we need to determine the gold-standard transduction actions, which requires aligning the lemmas and word forms. Better sequence alignment is one of the concerns for the similar inflection systems cited above, as well as the sequence-to-sequence models that operate with hard monotonic attention. Better alignments are also a common concern and studied extensively in other areas of computational linguistics such as dialectometry (Wieling et al., 2009; Prokić, 2010) and historical linguistics (List, 2012; Jäger, 2013). Standard alignment algorithms that use equal penalties for edit operations often fail to capture the similarities and differences between characters (or phonetic segments). As a result, often a weighted method is used such that similar characters in one of the sequences are more likely to be aligned with the similar characters in the other. The weights are most often learned from the data using an unsupervised method. The data-driven weights are found to be more effective than manually assigned weights based on linguistic knowledge/intuitions (Sofroniev and Çöltekin, 2018). We tried a few of these more informed weighted alignment methods. However, in preliminary experiments, a simple alignment mechanism based on longest common substring (LCS) worked best. Hence, in all the experiments reported here, alignments are performed first by finding the longest common substring of lemma and the word from, and aligning the two strings such that the LCS is aligned correctly. The rest of the characters are aligned disregarding whether they match or not. The method introduces gaps only at the beginning and end of the sequences. If there are two matching substrings of equal length, we pick the first sequence.

Figure 1: Example alignments of two lemma–form pairs from Turkish (top) and German (bottom). In each box, upper part shows the alignment based on longest common substring, while lower part shows minimum edit distance solution.

Figure 1 presents two example alignments based on the LCS and the edit distance. In the first example (top figure), minimum edit distance aligns substring `ek`, part of the infinitive marker `mek` used in verbal lemmas in the data set, to a substring string that matches accidentally in the word form. The intuition here is that even if we do not have a good reason for aligning the infinitive marker `mek` with the initial part of the suffix, doing this consistently facilitates learning. The example from German (and in general infixes) is a potentially problematic case for the LCS-based aligner. Minimum edit distance here produces an alignment that is intuitively better. However, since in most cases we expect a

Lemma (input)	Form (output)	Action
s	s	copy
c	c	copy
h	h	copy
r	r	copy
e	i	replace(i)
i	e	replace(e)
b	b	copy
e	e	copy
n	s	replace(s)
#	t	insert(t)
#	#	copy

Table 1: The sequence of actions mapping the German lemma *schreiben* 'to write' to its second person singular past form *schriebst*.

limited number prefixes to precede 'infixed' material, the LCS solution still provides reasonably regular patterns to predict.

2.2 Transduction actions

The inflection systems use four character-to-character transduction actions:

copy copies the current character of the lemma to the word form, and advances to the next character on the lemma

replace(c) inserts the character c to the word form, and advances to the next character on the lemma

insert(c) inserts the character c to the word form, without advancing the current lemma pointer

delete deletes the current character of the lemma, and advances to the next character on the lemma

All actions are character-to-character operations based on one-to-one alignments, and each action is represented individually, i.e., we do not compress consecutive actions of the same type to a single complex action. Table 1 demonstrates the series of transductions for an example lemma–word form pair. Both lemmas and words are appended with a special end-of-sequence symbol (indicated with '#' in Table 1). The decoding stops when any of the actions predict the end-of-sequence symbol.

2.3 Classifiers

Given the gold standard action sequences extracted from a training set as described above, we can use any multi-class classification method for predicting the next action. We experimented both with traditional linear classifiers, in particular SVMs, and feed-forward neural network classifiers. Regardless of the classification method, however, the features are based on the morphological tags, characters within a local window around the current lemma character, and the last few (predicted) characters of the word form. During training we use the gold-data for extracting features from the word forms.

Since the linear methods cannot represent non-linear combinations of the features, we use the following feature (combinations).

- The current lemma character.
- The varying-length, overlapping n-grams to the left and right of the current lemma character. For example, at the fourth step in Table 1, with current lemma character `r`, and assuming a window size of three, we include `h`, `ch`, and `sch` as n-grams before the current point, and `e`, `ei`, and `eib` as n-grams after the current point.
- The varying-length, overlapping n-grams of the last part of the output already predicted. For the same position, this would amount to n-grams `h`, `ch`, and `sch`.
- Morphological tags, including a special tag indicating the language, and all binary combination of tags. For example, with input tags `{V, PST, S}` we also include `{V-PST, V-S, PST-S}` as additional tag features.
- Cross product of all tag features with the other features.

All features for the linear classifiers are combined as a single sparse feature vector. The features are weighted using TF-IDF, but no pruning or any other feature selection step is employed.

As well as the choice of the window size, the choice of the features and feature combinations clearly is important for the linear models. Variations on this feature scheme, e.g., also including interactions between the n-grams, and including skip-grams may improve model's predictions.

However, they also increase the feature set size, resulting in an increases in the time it takes to tune the classifiers. The choice above was a compromise between accuracy and demands on computation (which may be an important factor when tuning models for 100 language pairs).

Intuitively, and also shown in similar tasks earlier, the neural models here have an expected advantage as they can learn useful combinations of arbitrary features automatically. The features for the neural classifier used in this study are based on the same set of characters and the tags, but without explicit combinations of the features.

For both type classifiers, a straightforward option is to train a single multi-class classifier is predicting all possible actions (including the composite actions such as replace(i)). Alternatively, one can first predict one of the four action types, and then predict the parameter of the action if the action is replace or insert. For the linear classifier, this means training three separate classifiers, and applying two of them in correct order at prediction time. For the neural model, a similar approach is used. The model first predicts the action, and then the parameter of the action, for which action is also given as an additional predictor. The different parts of the network are trained jointly, and share some of the weights which may provide additional benefits. We experimented with both approaches. Initial experiments produced mixed results, one or the other option performing better in different data sets. The results presented here are based on the two-level classifiers, chosen somewhat arbitrarily.

At prediction time we decode the sequence greedily, choosing the single-best action according to the model at each step. However, both systems can produce multiple outputs with minor modification to the decoding algorithm.

2.4 Cross lingual transfer

The main focus of the present task is cross-lingual transfer. Although we entertained a few ideas, including the use of cross-lingual character embeddings and translation of transition sequences, the approach used at the end was straightforward inclusion of the high-resource language data in training the models. During cross-lingual training, however, we include an additional hyperparameter that determines the weight training instances belonging to the source language. This way the model also learns 'how much to learn from the source language' during tuning.

Since a sizable number of language pairs do not use the same writing system, a learner relying on categorical character inputs cannot learn from the source language data. Even when the script used by both languages are the same, there are often differences in the writing traditions that make the transfer difficult. Without success from our preliminary experiments with cross-lingual character embeddings, we used the inputs as is, only experimenting with transliteration of the source language input to the target language input for a limited set of language pairs.

Performing the correct, or useful, transliterations for this task seems difficult. There are no standard transliteration methods defined for most language pairs in the data. The standard transliteration methods for some languages exists, where the standard typically defines how to transliterate a language written with a non-Latin script to some version of the Latin script. However, the standard methods are often designed for easy reading/phonetization by English speakers. Even in cases of target languages that use a version of the Latin script, there are significant differences to hinder cross-lingual learning considerably. As a result, we report below some of the experiments with transliterations between Latin and Cyrillic scripts for only eight language pairs (all Turkic languages) to demonstrate the potential gains that can be obtained with transliteration. The transliteration method follows Çöltekin and Barnes (2019). The method does not follow any transliteration standards (e.g., one set by ISO), but tries to maximize the similarities of the writing traditions in these particular languages.

3 Experimental setup

3.1 Data and preprocessing

The shared task data used in this study consists of 100 language pairs, which is described in detail in McCarthy et al. (2019). Here we only provide a basic overview that is relevant to our discussion below. All language pairs feature a high-resource training set from the source language, a low-resource training set and a development set, both from the target language. Number of unique source and target languages are both 44. The number of source languages for a target language, and number of target languages for a source language differ. Some languages also appear as both source and target languages in dif-

ferent pairs. Most source languages have 10 000 training instances, with a few exceptions (notably Uzbek with only 1 060 word forms). The number of training instances for all target languages is 100, with a single exception of Telugu with 61 word forms. Development set sizes vary more between 50, 100 and 1 000 word forms. The number of unique lemmas and tag combinations also vary among different training and development sets.

The relation of language pairs also differ. Most pairs have shared ancestry, ranging from very close (e.g., Turkish–Azeri) to rather far modern relatives (e.g., Russian–Portuguese), or historical relatives (e.g., Polish–Old Church Slavonic). There are also a few pairs where the relation is rather through geographical contact (e.g., Italian–Maltese). As noted above, one obstacle for cross-lingual learning is the different writing systems used in these languages. The data set includes 11 different scripts, and 30 of language pairs do not use the same script. It should be noted, however, that the use of common script does not necessarily solve all the problems regarding mapping character sequences across languages reliably. Even when they use the same script, e.g., Latin or Cyrillic, the differences adopted in the writing tradition of each languages may still introduce difficulties.

To overcome the differences in scripts, we transliterated source language data in eight pairs (Bashkir–Azeri, Bashkir–Crimean-Tatar, Bashkir–Tatar, Bashkir–Turkmen, Turkish–Kazakh, Turkish–Khakas, Uzbek–Kazakh, and Uzbek–Khakas) into the script used by the target language. The transliteration method used tries to maximize the similarity of the transliterations with the writing system of the target language.

3.2 Classifier tuning

For both classifiers we performed a random search through the hyperparameter space, which included the weight of the source language instances. Hence, both classifiers are tuned to make use of the source language based on their usefulness. The other common parameter for both the linear and the neural classifier included window size, which determines the number of characters to the left and right of the current lemma position, the number of characters from the end of the word form predicted so far. For linear models the only other hyperparameter we tune is the regularization constant.

For the neural model we fixed the architecture after some initial experimentation, where the action classifier had two hidden layers of 100 units with ReLU activation, followed by a softmax classifier. The part of the network that predict the parameter of insert and replace actions had one layer with ReLU activation followed by a softmax classifier. The input to the parameter classifier was the output of the hidden layer of the activity classifier, as well as the activity prediction. We used early stopping, stopping when the mean edit distance on the development set did not improve. We use only a single-best system, without any weight averaging or ensembling.

Both models are tuned on the training sets of target language, and both source and target combination with a parameter controlling the weight of the source language instances. Random search for transfer model includes the best model parameters tuned on mono-lingual low-resource setting with a source weight parameter set to 0. Hence, the transfer models for each language pair does at

All linear models were implemented in scikit-learn Python library (Pedregosa et al., 2011) using liblinear back end (Fan et al., 2008). The neural model was implemented with Tensorflow (Abadi et al., 2015) using Keras API (Chollet et al., 2015).

4 Results

The official results obtained by our linear and neural model alongside the best and worst baseline results published by the organizers presented in Table 2. The organizers offer a large number of baseline results. We only present the best (unpublished transformer model) and the worst ('untuned' monotonic alignment model). The best system in the competition (CMU-03) achieves an accuracy of 58.79 and mean edit distance of 1.52.

Besides the official results, we also present results obtained after the competition in Table 2. The row labeled '*Linear' presents the results obtained with the same linear classifier after fixing a bug in feature extraction and further tuning. The row marked as 'target only', presents results that are obtained using only the target language, without any attempt of cross-lingual learning. Both scores are obtained using the official evaluation script on the test data released after the end of the evaluation period.

The neural predictor performed worse than the linear predictor, within the (rather limited) effort and computational resources put into developing it. Although the performance is still below the top

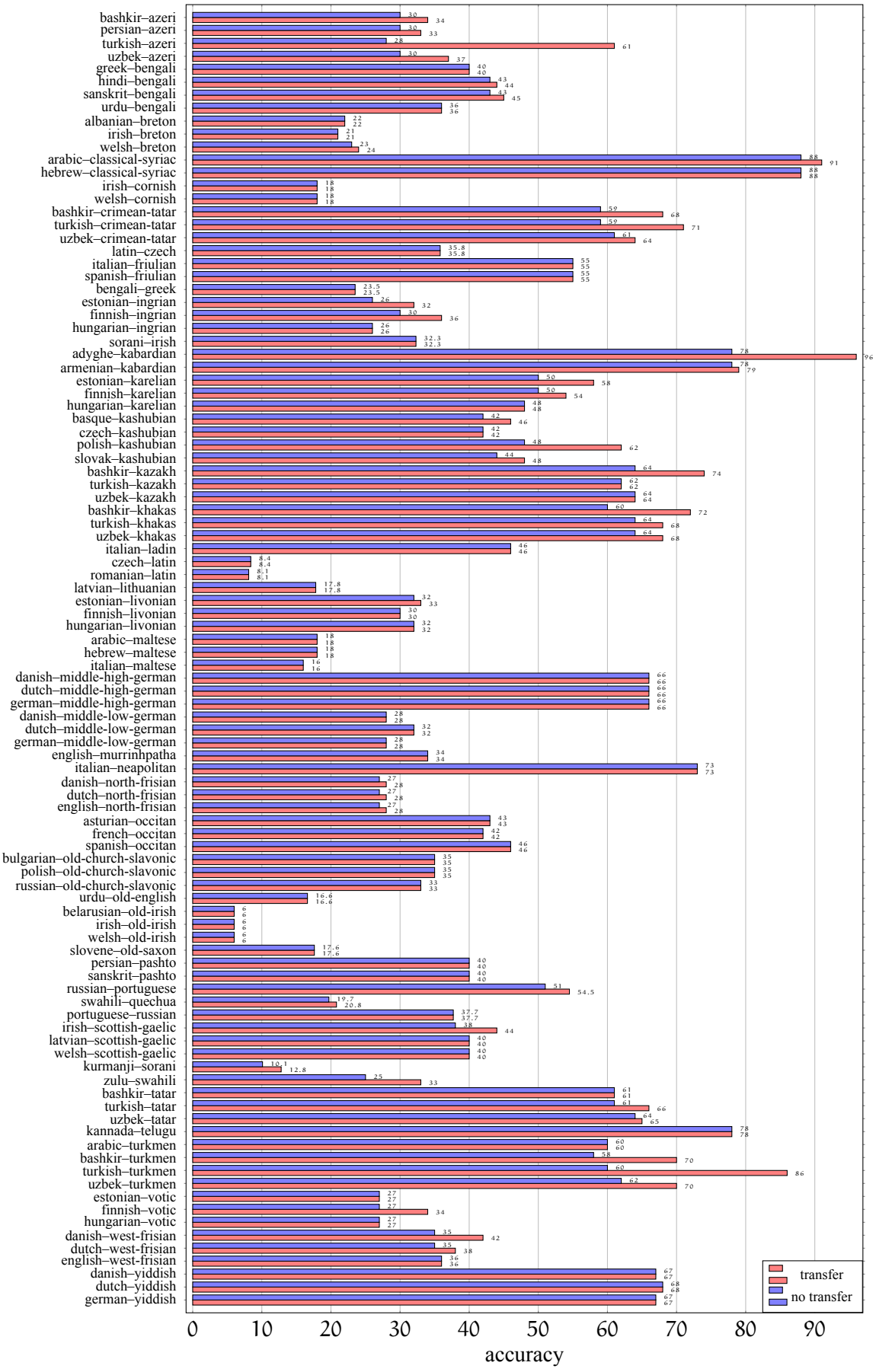

Figure 2: Detailed accuracy scores obtained using the linear predictor with and without source language data. The language pairs are sorted by the target language.

System	Accuracy	MED
Linear	34.49	1.88
Neural	20.86	2.36
*Linear	43.67	1.43
*Linear, target only	41.00	1.50
Baseline (worst)	28.76	2.07
Baseline (best)	54.25	1.13

Table 2: Overall results obtained by our systems in comparison to the official state-of-the-art baseline (Wu and Cotterell, 2019). The scores are word-form accuracy and mean edit distance (MED) averaged over all 100 language pairs. The rows marked with asterisk indicate post-evaluation scores obtained using the linear predictor, after fixing a bug and further tuning.

performing systems, the post-evaluation fixes and tuning results in a dramatic increase in the performance of the linear model. The more interesting result, however, is the small difference between the transfer learning results and the 'target only' results. We present the target-only and transfer accuracy scores for each language pair in Figure 2. In general, the gains from cross-lingual learning are modest. There is no improvement at all for 59 of the language pairs. As expected, this includes all 30 pairs with writing system mismatch, excluding some of the language pairs for which we transliterated the source data. The effect of transliteration is rather modest as well, yielding an improvement between 4 to 12 percentage points of accuracy for four of the eight language pairs where it was used. The effect of the transliteration to the overall score is a 0.29 % increase in accuracy. Not surprisingly, the highest increases due to cross-lingual learning are obtained when source and target languages are closely related. The highest increase is obtained from Turkish to Azeri with 33 %, followed by Turkish–Turkmen and Adyghe–Kabardian with 26 % and 18 % respectively.

5 Summary and outlook

We presented a simple inflection system based on predicting transduction actions. Of the predictors we tried, the linear predictor performs reasonably well. Although its performance is lower than the top performing systems in the shared task, the system is far from being well-tuned, and as demonstrated above simple improvements may have a major effects on the performance. Furthermore, the linear predictor has the advantage of requiring relatively less computational resources, which may be advantageous in some cases. One further advantage is the ease of analyses of linear learners. What the linear model learns is often much simpler to understand and interpret, and although the need for crafting feature combinations is one of its weaknesses, it may also provide further insight through more interpretable ablation studies. Our neural predictor did not perform as well as the linear predictor. This, however, is by no means a conclusive result. If tuned well, neural networks should in fact work well in this task because of their capability of learning arbitrary combinations of their inputs.

On the cross-lingual side of the problem, the improvements we get are rather modest. In fact, there is a only small overall improvement due to cross-lingual learning over learning only from low-resource target language. Since our relatively simple system can get up to 40 % accuracy by learning only from the small target language training sets, there is also a good chance that more successful systems are also relying more on the target language data rather than benefiting from transfer learning. Some of the reasons for low success is probably the make up of the data. Not all language pairs are close enough to facilitate the transfer learning. However, there are many possible directions for exploiting the cross-lingual signal better. The simple method used in this study can be improved in many ways. Although our initial experiments were not successful. We believe cross-lingual character embeddings, and 'translating' transduction actions from source language to target language may be potential ways to get a better cross-lingual input.

Acknowledgments

Some of the experiments reported here were run on a Titan Xp donated by the NVIDIA Corporation.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Tal-

war, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Malin Ahlberg, Markus Forsberg, and Mans Hulden. 2015. Paradigm classification in supervised learning of morphology. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1024–1029, Denver, Colorado. Association for Computational Linguistics.

Iñaki Alegria and Izaskun Etxeberria. 2016. EHU at the SIGMORPHON 2016 shared task. a simple proposal: Grapheme-to-phoneme for inflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 27–30, Berlin, Germany. Association for Computational Linguistics.

Jeremy Barnes, Roman Klinger, and Sabine Schulte im Walde. 2018. Bilingual sentiment embeddings: Joint projection of sentiment across languages. In *Proceedings of the 56th Annual Meeting of the Association for Comutational Linguistics (Volume 1: Long Papers)*, pages 2483–2493.

François Chollet et al. 2015. Keras. `https://github.com/fchollet/keras`.

Çağrı Çöltekin and Jeremy Barnes. 2019. Neural and linear pipeline approaches to cross-lingual morphological analysis. In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 153–164, TOBEFILLED-Ann Arbor, Michigan. Association for Computational Linguistics.

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. 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. 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. 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.

Greg Durrett and John DeNero. 2013. Supervised learning of complete morphological paradigms. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1185–1195, Atlanta, Georgia. Association for Computational Linguistics.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Manaal Faruqui and Shankar. Kumar. 2015. Multilingual open relation extraction using cross-lingual projection. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pages 1351–1356.

Manaal Faruqui, Yulia Tsvetkov, Graham Neubig, and Chris Dyer. 2016. Morphological inflection generation using character sequence to sequence learning. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 634–643, San Diego, California. Association for Computational Linguistics.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Gerhard Jäger. 2013. Phylogenetic Inference from Word Lists Using Weighted Alignment with Empirically Determined Weights. *Language Dynamics and Change*, 3(2):245–291.

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.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick

Xia, Manaal Faruqui, Sebastian J. Mielke, Arya 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*, Miyazaki, Japan. European Language Resource Association.

Kimmo Koskenniemi. 1985. Compilation of automata from morphological two-level rules. In *Papers from the Fifth Scandinavian Conference of Computational Linguistics*, page 143–149.

Johann-Mattis List. 2012. SCA: Phonetic Alignment based on sound classes. *New Directions in Logic, Language and Computation*, pages 32–51.

Ling Liu and Lingshuang Jack Mao. 2016. Morphological reinflection with conditional random fields and unsupervised features. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 36–40, Berlin, Germany. Association for Computational Linguistics.

Peter Makarov and Simon Clematide. 2018. UZH at CoNLL–SIGMORPHON 2018 shared task on universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 69–75, Brussels. Association for Computational Linguistics.

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

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

Garrett Nicolai, Colin Cherry, and Grzegorz Kondrak. 2015. Inflection generation as discriminative string transduction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 922–931, Denver, Colorado. Association for Computational Linguistics.

Garrett Nicolai, Bradley Hauer, Adam St Arnaud, and Grzegorz Kondrak. 2016. Morphological reinflection via discriminative string transduction. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 31–35, Berlin, Germany. Association for Computational Linguistics.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CoNLL)*, pages 49–56.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jelena Prokić. 2010. *Families and resemblances*. Ph.D. thesis, University of Groningen.

Pavel Sofroniev and Çağrı Çöltekin. 2018. Phonetic vector representations for sound sequence alignment. In *Proceedings of the Fifteenth Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 111–116, Brussels, Belgium. Association for Computational Linguistics.

Martijn Wieling, Jelena Prokić, and John Nerbonne. 2009. Evaluating the pairwise string alignment of pronunciations. In *Proceedings of the EACL 2009 workshop on language technology and resources for cultural heritage, social sciences, humanities, and education*, pages 26–34.

Shijie Wu and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. *arXiv preprint arXiv:1905.06319v1*.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of 8th international workshop on parsing technologies (IWPT)*, pages 195–206.

David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the first international conference on Human language technology research*, pages 1–8. Association for Computational Linguistics.