UZH at CoNLL-SIGMORPHON 2018 Shared Task on Universal Morphological Reinflection

Makarov, Peter ; Clematide, Simon

Originally published at:
Makarov, Peter; Clematide, Simon (2018). UZH at CoNLL-SIGMORPHON 2018 Shared Task on Universal Morphological Reinflection. In: CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection, Brussels, Belgium, 31 October 2018. Association for Computational Linguistics, 69-75.

UZH at CoNLL–SIGMORPHON 2018 Shared Task on Universal Morphological Reinflection

Peter Makarov **Simon Clematide**
Institute of Computational Linguistics
University of Zurich, Switzerland
makarov@cl.uzh.ch simon.clematide@cl.uzh.ch

Abstract

This paper presents the submissions by the University of Zurich to the CoNLL–SIGMORPHON 2018 Shared Task on Universal Morphological Reinflection. Our system is based on the prior work on neural transition-based transduction (Makarov and Clematide, 2018b; Aharoni and Goldberg, 2017). Unlike the prior work, we train the model in a fully end-to-end fashion—without the need for an external character aligner—within the framework of imitation learning. In the type-level morphological inflection generation challenge (Task I), our five-strong ensemble outperforms all competitors in all three data-size settings. In the token-level inflection generation challenge (Task II), our single model achieves the best results on three out of four sub-tasks that we have participated in.

1 Introduction

The CoNLL–SIGMORPHON 2018 Shared Task on Universal Morphological Reinflection (Cotterell et al., 2018) focuses on inflection generation at the type level (Task I) and in context (Task II). Both tasks feature three settings depending on the maximum number of training examples: low, medium, and high. The team from the University of Zurich has taken part in both tasks with submissions featuring neural transition-based transducers (Aharoni and Goldberg, 2017; Makarov and Clematide, 2018b). The model transduces a string by a sequence of traditional character edit operations. The neuralized transducer, which conditions edits on the entire input string and captures unbounded dependencies in the output, has proven very effective in the past editions of the SIGMORPHON shared task (Aharoni et al., 2016; Makarov et al., 2017). Typically, this model is trained by maximizing the likelihood of gold action sequences that come from a separate character aligner. This year, we train with an imitation learning method (Makarov and Clematide, 2018a) that enforces optimal alignment in the loss and additionally supports action-space exploration and the optimization of a task-specific objective. Our method entirely eliminates the need for a character aligner and results in substantially stronger models, at the expense of slight increase in training time. The resulting models evaluate favorably on both CoNLL–SIGMORPHON 2018 tasks. On Task I, our five-strong ensemble `uzh-02` outperforms the nearest competitor by over 1% absolute accuracy in the high setting (24.4% error reduction) and over 2% absolute in the medium setting and 4% absolute in the low setting (13.9% and 8.6% error reduction, respectively). The larger ensemble `uzh-01` further improves the result slightly in the high and medium settings (1% and 2% error reduction, respectively). For Task II, we submit a single model to only the low and medium settings. The single model dominates the low setting, being also the only system that beats the predict-the-lemma baseline. The model comes second in the track 1 medium setting with almost 4% absolute accuracy behind the winner (8.5% error increase), and is the best in the track 2 medium setting with almost 4% absolute above the runner-up (6.7% error reduction).

2 Task description

The now classic Task I requires mapping a lemma form (e.g. “Schlüssel” meaning “key” in Swiss German) to an inflected form (“Schlüssle”) given a morpho-syntactic description (N;DAT;PL). The new Task II requires mapping a lemma (“Schlüssel”) to an inflected form (“Schlüssle”) given sentential context (“Du muesch de _____ Sorg gäh.” / “You need to take care of the keys.”).

This year’s edition of Task I features an unprecedented 102 languages. As in 2017, the low

Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection, pages 69–75,
Brussels, Belgium, October 31, 2018.

setting offers just 100 training samples per language, and it is at most 1,000 and 10,000 samples in the medium and high settings, respectively.

Task II features six Indo-European languages (German, English, Swedish; Spanish, French; Russian) and one Uralic language, Finnish. Track 1 of Task II additionally provides the morpho-syntactic specifications of all context words.

3 State of the Art

Over the past two years, type-level inflection generation in the high setting has been dominated by general sequence-to-sequence models (seq2seq) with soft attention (Kann and Schütze, 2016; Bergmanis et al., 2017). Neuralized counterparts of traditional transition-based transducers have proven highly competitive, being particularly effective in lower-resource settings (Aharoni et al., 2016; Makarov et al., 2017). Inflection generation in context is a novel SIGMORPHON challenge and, generally, a less studied problem. Recently, Wolf-Sonkin et al. (2018) propose a context-aware deep generative graphical model that generates sequences of inflected words.

4 Methods

Our model is a version of the transition-based transducer with a designated copy action (Makarov and Clematide, 2018b). The model edits the lemma into the inflected form by a sequence of single-character edit actions (DELETE, COPY, INSERT(c) for any output character c). Through the use of a recurrent neural network, the choice of edit is conditioned on the globally contextualized representation of an input character and the full history of edits. Concretely, at timestep t, the probability distribution over edits a_t is computed with a softmax classifier:

$$P(a_t \mid \mathbf{a}_{<t}, \mathbf{x}) = \text{softmax}\big(\mathbf{W} \cdot \mathbf{s}_t + \mathbf{b}\big), \quad (1)$$

where $\mathbf{W}$ and $\mathbf{b}$ are classifier weights and $\mathbf{s}_t$ is the output of the long short-term memory cell (Hochreiter and Schmidhuber, 1997, LSTM):

$$\mathbf{s}_t = \text{LSTM}(\mathbf{c}_{t-1}, \big[E(a_{t-1}); \mathbf{h}_i; \mathbf{f}\big]). \quad (2)$$

Vector $\mathbf{c}_{t-1}$ denotes the previous hidden state, $E(a_{t-1})$ is the embedding of the previous edit, $\mathbf{f}$ is the embedding of the morpho-syntactic description (MSD), and $\mathbf{h}_i$ is the encoding of an input character x_i with a bidirectional LSTM (Graves and Schmidhuber, 2005). Given the set $\{f_h\}_{h=1}^{H}$ of all morpho-syntactic features seen at training, we compute the embedding $\mathbf{f}$ of an MSD as a concatenation of individual feature embeddings $[F(f_1); \ldots; F(f_H)]$ where we use a designated embedding $F(0)$ instead of $F(f_h)$ if f_h does not occur in the MSD. Also, we use the same embedding for input character c and action INSERT(c).

Our submission differs from the previous work in the way we train this model.

4.1 Task I: Type-level Inflection Generation

The transducer is typically trained to maximize the conditional likelihood of a gold edit action sequence given a lemma (Aharoni and Goldberg, 2017). Gold actions are generated by a character alignment algorithm, such as minimal edit distance (Levenshtein, 1966), applied to lemma-word pairs. The performance of the transducer, therefore, hinges on the quality of character alignments that, in turn, might depend on the amount of training data (e.g. if we employ a statistical alignment model such as a Bayesian nonparametric aligner). On the one hand, it is unsatisfactory to have to choose aligners for different settings. On the other, multiple edit sequences are often equally likely, and yet the choice of a single gold sequence does not depend on the MSD. This leads to the learning of a suboptimal transducer.

Our solution, therefore, is to remove the aligner entirely and instead optimize a sequence-level loss that enforces optimal alignment. Concretely, our loss is a weighted sum of (i) the minimal edit distance between the target word $\mathbf{y}$ and prediction $\hat{\mathbf{y}}$ and (ii) the cost of the sequence of edits from lemma $\mathbf{x}$ to prediction $\hat{\mathbf{y}}$:

$$\ell(\mathbf{a}, \mathbf{x}, \mathbf{y}) = \beta \text{ distance}(\mathbf{y}, \hat{\mathbf{y}}) + \text{cost}(\mathbf{a}) \quad (3)$$

The first term is the task objective. The second term pushes the model to achieve the objective with the least number of edits. To compute the terms, we use unit costs for DELETE and INSERT(c) for any output character c and zero cost for COPY. $\beta \geq 1$ is a penalty for unit distance.

While there exist many techniques that minimize a non-differentiable loss, most of them require initialization with a pretrained model (Ranzato et al., 2016; Bahdanau et al., 2017; Shen et al., 2016). They assume that at each specific timestep, the gold transition is unknown. Training

signal comes from the entire sequence of transitions, which is typically too sparse for a cold start (Leblond et al., 2018). This assumption is not valid for morphological string transduction, and it is easy to design an efficient procedure that returns edits that result in the lowest sequence-level loss (assuming that all future transitions are, too, selected optimally). Specifically, such a procedure should return an edit that leads to the completion of the suffix of target $\mathbf{y}$ using a transition sequence with the lowest edit cost. For example, if the partial prediction $\hat{\mathbf{y}}$ for the lemma-word pair ("Schlüssel", "Schlüssle") is "Schlüss" and the model attends to $x_6 = e$, the procedure returns DELETE. Using this technique, we collect per-timestep losses that reflect the impact of single edits on the sequence-level loss.

Our training method is thus an instance of imitation learning (Daumé III et al., 2009; Ross et al., 2011; Chang et al., 2015), also familiar from transition-based dependency parsing (Goldberg and Nivre, 2012). For the submission, we train the model by maximizing the marginal log-likelihood (Riezler et al., 2000; Goldberg, 2013; Ballesteros et al., 2016) with a variant of stochastic gradient ascent:

$$\mathcal{L}(\mathbf{x},\mathbf{y},\Theta) = \sum_{t=1}^{m} \log \sum_{a \in A_t} P(a \,|\, \mathbf{a}_{<t}, \mathbf{x}, \Theta) \quad (4)$$

Θ are the model parameters and A_t is the set of optimal edits at timestep t. To train with exploration, which addresses the exposure bias, we use a roll-in schedule and model roll-outs (Chang et al., 2015). In the *roll-in* phase, the next edit a_{t+1} is either sampled uniformly at random from the set A_t of optimal edits (expert roll-in) or from the current model's distribution over valid edits (model roll-in). This choice is controlled by a Bernoulli random variable that depends on the training epoch η. We use a roll-in schedule that gradually adds sampling from the model as training proceeds. In the *roll-out* phase, we estimate the future effect of some next action a_{t+1} on the sequence-level loss $l(\mathbf{a}, \mathbf{x}, \mathbf{y})$ by either using the optimal procedure outlined above (expert roll-out) or running the model for the rest of the input (model roll-out). Again, this choice is controlled by a Bernoulli random variable. The purpose of β from Eq. 3 is to cut down the number of expensive model roll-outs: In case a_{t+1} results in accuracy error (e.g. by inserting a letter that does not occur in the target $\mathbf{y}$), we set the sequence-level loss associated with this edit to β.

Hyperparameter	Value
char. & action embedding (E)	100
feature embedding (F)	20
context char. embedding (Task II)	20
context feat. embedding (Task II)	10
batch size	1
epochs / patience (high)	30 / 5
epochs / patience (medium)	50 / 15
epochs / patience (low)	60 / 20
optimization	ADADELTA
β	5
roll-in	$\frac{k}{k+\exp(\eta/k)}, k = 12$
roll-out	0.5
beam width	4

Table 1: Model hyperparameters.

The model hyperparameters and optimization details are given in Table 1.

4.2 Task II: Inflection Generation in Context

Our submission involves a minor change to the model described above. Similar in spirit to the baseline of Task II, we compress the immediate context into context vector $\mathbf{g}$ and use it in place of the feature vector. For track 2 of Task II, the context vector is a concatenation of the character LSTM encodings of the words to the immediate right and left. For track 1 of Task II, the context vector also includes the embeddings of their MSDs, which we compute just as the feature vector $\mathbf{f}$ in the type-level model. We use smaller dimensions in the computation of context vectors (Table 1). We also considered larger context windows. For French and Swedish in the medium setting, our submission uses as context two words to the left and two words to the right since we observe substantial gains in accuracy over the two-word context window.

Following the baseline, we train on verbs, nouns, and adjectives for track 1 and all words paired with lemmas for track 2. We do not exploit any knowledge of the development and test sets[1] and assume that test sentences contain multiple gaps as in the development set. We do, however, use the fact that the official evaluation script ignores case distinctions. We lowercase all the

[1] At least for some languages (e.g. Russian and English), the development and test sets differ in the types of lexical categories to be predicted.

Task I

	uzh-01	uzh-02	#	avg	baseline	compet.
development set						
H	96.18	**96.29**	10	94.98	78.12	–
M	87.60	**87.87**	12	85.66	63.05	–
L	57.52	**58.30**	15	54.73	38.81	–
test set						
H	**96.00**	95.97		–	77.42	94.66
M	**86.64**	86.38		–	63.53	84.19
L	57.18	**57.21**		–	38.89	53.22

Table 2: Overview of Task I results. H, M, L=high, medium, low settings; #=number of models that the average is taken over; compet.=nearest competitor.

Task II

	UZH--01--2	baseline	predict lemma	compet.
development set				
M 1	53.05	41.44	35.09	–
M 2	49.53	33.55		–
L 1	41.75	1.61		–
L 2	38.40	1.58		–
test set				
M 1	53.02	44.09	36.62	**56.70**
M 2	**48.88**	38.56		45.18
L 1	**42.42**	1.85		29.86
L 2	**38.60**	2.19		33.38

Table 3: Overview of Task II results.

low-setting data, which results in an improvement.

4.3 `uzh-01` and `uzh-02`: Ensembling Strategies for Task I

Ensembling addresses the effects of random initialization when the objective function is not concave (Reimers and Gurevych, 2017). Both our Task I submissions are majority-vote ensembles. For each language, we compute 15 models for the low setting, 12 for the medium setting, and 10 for the high setting. For each language, run `uzh-02` is an ensemble over five models with the highest development set accuracies. Run `uzh-01` is an ensemble over all computed models. Ties are broken at random.

The Task II submission contains only a single model for each language.

All models are decoded with beam search.

5 Results and Discussion

5.1 Task I Results and Discussion

In Task I, our five-model majority-vote ensemble (`uzh-02`) outperforms the nearest competitor in the high setting (`mbe-02`) by 1.3% absolute accuracy (24.4% error reduction), and by 2.2% absolute in the medium setting (`iitbhu-iiith-02`) and 4.0% in the low setting (`ua-08`) (13.8% and 8.5% error reduction, respectively). The larger ensemble `uzh-01` is somewhat stronger in the medium and high settings than `uzh-02` and only marginally weaker in the low setting (Table 2). Ensembling adds consistent improvement over the single-model average, which suggests an uncomplicated way to improve our Task II results.

We also compare the performance of `uzh-02` to the highest accuracy achieved for each language and data setting by any of our competitors (Figure 1).[2] In the low setting, `uzh-02` is occasionally much behind the best achieved result (e.g. for Latin, Old English, Hebrew, Norwegian, and Danish) and behind the average 59.22% with a 4.9% relative error increase. For low-setting Lithuanian, `uzh-02` fails to improve over the baseline whereas `ua-08` and `ua-06` beat it by a large margin. In the medium and high settings, there are very few languages in which `uzh-02` is beaten by a competitor. `uzh-02` is more accurate on average (86.38% vs 84.79% in the medium setting and 95.97% vs 95.43% in the high setting) with error reductions of 10.4% and 12.0%, respectively.

Thus, with an appropriate training method, the neural transition-based model can be very strong in the high data setting. This is in line with the results for the SIGMORPHON 2016 dataset in Makarov and Clematide (2018a). On the other hand, we expect that gains can be made with the general soft-attention seq2seq model (or any latent-alignment model), by applying the same training method or other existing alternatives (Edunov et al., 2017).

Following a reviewer's request, we also compare the performance of the new model to that of the copy-enhanced variant of the hard attention model trained by maximizing the conditional likelihood of separately derived gold edits (Makarov and Clematide, 2018a, CA). (This model outperformed all competitors in last year's low setting.) Due to limited resources, we only compute low-setting CA models. The new model makes substantial gains: `uzh-01` and `uzh-02` achieve 8.5% and 7.8% error reduction over their CA ensemble counterparts (53.21% vs 57.18% and 53.62% vs 57.21%, respectively). This is con-

[2] Thus, this does not include the results from run `uzh-01`.

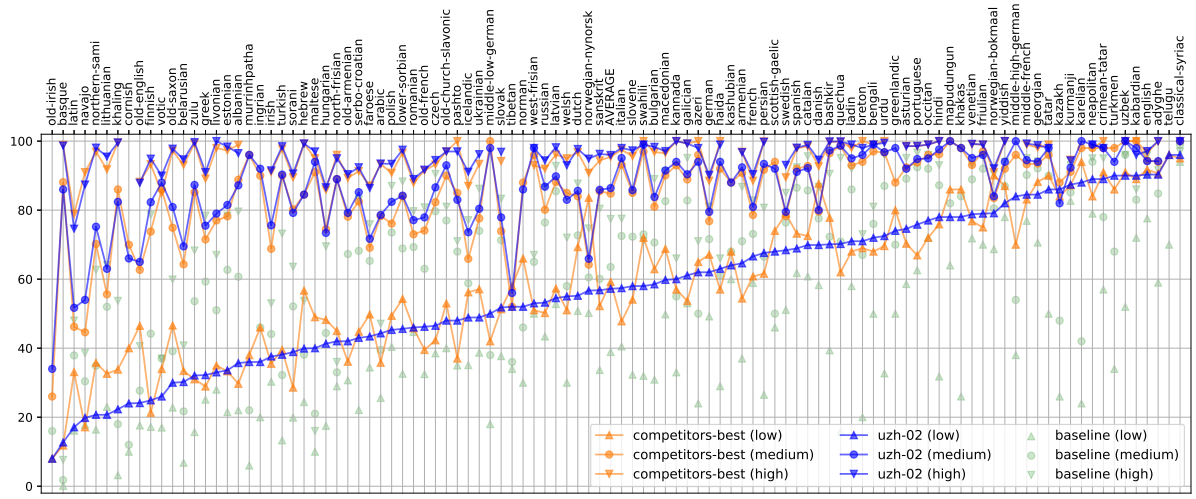

Figure 1: Test set accuracies for all languages and data-size settings of Task I: `uzh-02` (blue), the best result by any of the competitors (orange), and the official baseline (green).

sistent with the improvements that Makarov and Clematide demonstrate with the new model on the 2017 SIGMORPHON shared task data.

5.2 Task II Results and Discussion

We submit one model (`UZH--02--1`) to only the medium and low settings. The model comes second in the medium setting (track 1), behind `COPENHAGEN--01--2` with an absolute accuracy difference of 3.7% and an error increase of 8.5%. It outperforms by 3.7% absolute accuracy and an error reduction of 6.7% the nearest competitor (`COPENHAGEN--01--2`) in the medium setting (track 2) and by 12.6% (track 1, `CUBoulder--02--2`) and 5.2% absolute (track 2, `NYU--01--2`) in the low setting (17.9% and 7.8% error reduction, respectively) (Table 3).

Our model is the only system in the low setting that beats the baseline that makes prediction by copying over the lemma string ("predict lemma" in Table 3). Moreover, for individual languages, it is always as good as or better than this baseline, with the largest improvements for Spanish (8.8% and 6.7% absolute accuracy in tracks 1 and 2), French (11.2% in track 1), and Russian (18.6% and 5.9%).

We take a closer look at the Russian and French data to better understand the task that our system solves. About 98% of the gaps in the Russian development data correspond to nouns, and it is 100% verbs for French. We sample uniformly at random 100 development set examples with MSDs (track 1) for each language and limit the context to two words to the left and two words to the right. A native speaker (with a linguistic background) attempts to predict the correct word form; additionally, they indicate whether their prediction is fully determined by the local context (and the MSD).

Human accuracy is 78% for Russian and 72% for French. Local context determines exactly 39% and 29% of examples. A good choice of the default prediction can be very effective: The upper bound formed by predicting based on local context and otherwise copying over the lemma is 57% for Russian (and 29% for French, which is unsurprising since the French infinitive is a relatively infrequent verb form). Except for long-range dependencies (e.g. conjunction, sequence of tenses), whose frequency is fairly low for nominal categories, bridging the gap to human performance primarily requires the knowledge of the word's lexical properties (e.g. being an uncountable noun) and usage, rather than morpho-syntactic information about other words in the sentence.

6 Conclusion

We use an imitation learning method to train a neural transition-based transduction model, which has previously been shown to be highly competitive on inflection generation and other morphological tasks and particularly strong in the limited-resource setting. The new training method elimi-

nates the need for an external character aligner, integrating alignment into the training objective and thereby avoiding error propagation due to suboptimal alignments. Further improvement comes from optimizing the task metric and performing action-space exploration. Importantly, the new training method produces very strong models in the high data-size setting, which has previously been dominated by general soft-attention seq2seq models. In type-level inflection generation, our five-model majority-vote ensemble outperforms all competitors in all three data settings. Our single model submission comes out on top in three out of six challenges in inflection generation in context.

Acknowledgment

We would like to thank the organizers for the great effort. We thank Anne Göhring and Michi Amsler for their linguistic help and the reviewers for their interest and comments. Peter Makarov is supported by European Research Council Grant No. 338875. Simon Clematide is supported by the Swiss National Science Foundation under grant CRSII5_173719.

References

Roee Aharoni and Yoav Goldberg. 2017. Morphological inflection generation with hard monotonic attention. In *ACL*.

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 *14th Annual SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology at ACL 2016*.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. In *ICLR*.

Miguel Ballesteros, Yoav Goldberg, Chris Dyer, and Noah A Smith. 2016. Training with exploration improves a greedy stack-LSTM parser. In *EMNLP*.

Toms Bergmanis, Katharina Kann, Hinrich Schütze, and Sharon Goldwater. 2017. Training data augmentation for low-resource morphological inflection. *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*.

Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daume III, and John Langford. 2015. Learning to search better than your teacher. In *ICML*.

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*, Brussels, Belgium. Association for Computational Linguistics.

Hal Daumé III, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine learning*, 75(3).

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2017. Classical structured prediction losses for sequence to sequence learning. In *NAACL*.

Yoav Goldberg. 2013. Dynamic-oracle transition-based parsing with calibrated probabilistic output. In *Proc. of IWPT*.

Yoav Goldberg and Joakim Nivre. 2012. A dynamic oracle for arc-eager dependency parsing. In *COLING*.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8).

Katharina Kann and Hinrich Schütze. 2016. MED: The LMU system for the SIGMORPHON 2016 shared task on morphological reinflection. In *14th Annual SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology at ACL 2016*.

Rémi Leblond, Jean-Baptiste Alayrac, Anton Osokin, and Simon Lacoste-Julien. 2018. SEARNN: Training RNNs with global-local losses. In *ICLR*.

Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10(8).

Peter Makarov and Simon Clematide. 2018a. Imitation learning for neural morphological string transduction. In *To appear in EMNLP*.

Peter Makarov and Simon Clematide. 2018b. Neural transition-based string transduction for limited-resource setting in morphology. In *COLING*.

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*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *ICLR*.

Nils Reimers and Iryna Gurevych. 2017. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *EMNLP*.

Stefan Riezler, Jonas Kuhn, Detlef Prescher, and Mark Johnson. 2000. Lexicalized stochastic modeling of constraint-based grammars using log-linear measures and EM training. In *ACL*.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *ACL*.

Lawrence Wolf-Sonkin, Jason Naradowsky, Sebastian J Mielke, and Ryan Cotterell. 2018. A structured variational autoencoder for contextual morphological inflection. In *ACL*.