Neural Transition-based String Transduction for Limited-Resource Setting in Morphology

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

Abstract

We present a neural transition-based model that uses a simple set of edit actions (copy, delete, insert) for morphological transduction tasks such as inflection generation, lemmatization, and reinflection. In a large-scale evaluation on four datasets and dozens of languages, our approach consistently outperforms state-of-the-art systems on low and medium training-set sizes and is competitive in the high-resource setting. Learning to apply a generic copy action enables our approach to generalize quickly from a few data points. We successfully leverage minimum risk training to compensate for the weaknesses of MLE parameter learning and neutralize the negative effects of training a pipeline with a separate character aligner.

1 Introduction

Morphological string transduction involves mapping one word form into another, possibly given a feature specification for the mapping, and comprises such inflectional morphology tasks as reinflection and lemmatization (Figure 1), and related problems such as normalization of historical texts. Traditionally, this task has been solved with weighted finite state transducers (Mohri, 2004; Eisner, 2002, WFST). Recently, it has been approached with neural sequence-to-sequence (seq2seq) methods (Faruqui et al., 2016; Kann and Schütze, 2016), inspired by the advances in neural machine translation (Sutskever et al., 2014; Bahdanau et al., 2014). Albeit offering a general solution to a special case of the string-to-string mapping problem, seq2seq models are highly data-intensive. The long tradition of modeling for morphology offers insights into the specifics of the task, suggesting models that would exploit inductive biases and thereby attain lower sample complexity. Recent works in seq2seq morphology model full input string context and unbounded dependencies in the output, but also propose conditioning generation on the context-enriched representation of one input character at a time (Aharoni and Goldberg, 2017; Yu et al., 2016). This and constraining character alignment to be monotonic bring this line of work close to traditional WFST approaches, which monotonically modify a string by performing local changes.

fliegen
{VERB, PAST TENSE,
3RD PERSON, SINGULAR} ▷ flog   na hainmneacha ▷ ainm

Figure 1: Morphological inflection generation in German (left). Lemmatization in Irish (right).

Having as our starting point the hard monotonic attention model of Aharoni and Goldberg (2017, HA), our goal is to improve seq2seq morphological processing by explicitly modeling local string edits commonly studied in traditional approaches. Our contributions are as follows:

- First, we explain HA as a neural transition-based system over edit actions. Alternative models are then available, differing in the choice of edit actions. We argue that extending HA with the COPY edit action is crucial and supported by the nature of the problem, accounting for large performance gains especially in the low-resource setting.

- Second, trained with the original MLE procedure, HA relies on gold action sequences computed by a separate character aligner. As a result, the overall approach is a pipeline. We propose enabling exploration at training (e.g. via expected risk minimization (MRT) or reinforcement learning-style training), thereby allowing the model to prefer alternative actions that also lead to the correct output sequence and neutralizing negative effects of the pipelined architecture. Additionally, this approach benefits from directly optimizing a sequence-level performance metric.

- Third, we conduct extensive experiments on the morphological inflection generation, reinflection and lemmatization tasks, showing that our approaches come near to or improve on the state-of-the-art results. We make our code and model predictions publicly available.[1]

2 Model Description

In our approach, we seek the most probable sequence of edit actions for a given input string and an optional feature specification for the transduction. Unlike traditional WFST approaches to this problem, we abandon the explicit modeling of all possible edit sequences via latent alignments in favor of a greedy, representationally rich RNN-powered transition-based architecture. When training with the MLE criterion following Aharoni and Goldberg (2017), our overall set-up is a pipeline of a character aligner followed by a greedy neural string transducer. Character alignments generated by the aligner are mapped to gold action sequences, whose conditional likelihood the neural transducer then learns to maximize. Under training with exploration, the neural transducer no longer relies on gold action sequences. Instead, the parameters are adjusted to directly maximize the model's accuracy of producing training-set output sequences.

Let Σ_x, Σ_y, and Σ_a be alphabets of input characters, output characters, and edit actions, respectively. Let $\mathbf{x} = x_1, \dots, x_n$, $x_i \in \Sigma_x$ denote an input sequence, $\mathbf{y} = y_1, \dots, y_p$, $y_j \in \Sigma_y$ an output sequence, and $\mathbf{a} = a_1, \dots, a_m$, $a_t \in \Sigma_a$ an action sequence. Let $\{f_h\}_{h=1}^{H}$ be the set of morpho-syntactic features.

seq2seq state-transition system We build a greedy transition-based string transducer that uses a seq2seq neural network to model arbitrary dependencies in the input sequence, the unbounded action history, and the non-deterministic choice of the next action. The system operates a buffer filled with RNN-encoded input characters, and a decoder RNN, which implements a push-only stack. The configuration of the system is given by the decoder state. Transitions are scored based on the output of the decoder, which takes as input the encoded character from the top of the buffer. Here, we elaborate on the model architecture.

We encode input sequence $\mathbf{x}$ with a bidirectional LSTM (Graves and Schmidhuber, 2005)

$$\mathbf{h}_1, \dots, \mathbf{h}_n = \text{BiLSTM}(E(x_1), \dots, E(x_n)), \tag{1}$$

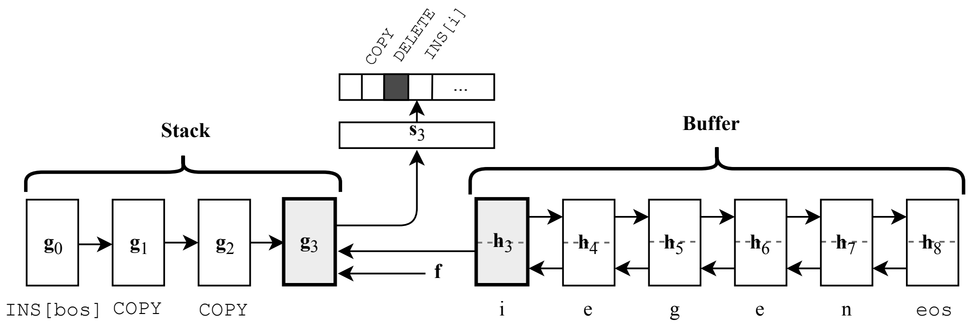
where E returns the embedding for x_i. Vector $\mathbf{h}_i$ is thus the representation of x_i in the context of the entire sequence $\mathbf{x}$. We push $\mathbf{h}_1, \dots, \mathbf{h}_n$ in reversed order into the buffer. Transduction begins with the full buffer and the empty decoder state.

The decoder LSTM keeps track of the past actions and—through conditioning at each step on $\mathbf{h}_i$—knows of character x_i at the top of the buffer and the full contents of the buffer. From the latest state of the decoder $\mathbf{c}_{t-1}$, we compute the configuration of the system:

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

where the input is the concatenation of (i) the embedding of the previous action (given by A), (ii) $\mathbf{h}_i$ from the top of the buffer indicating the current position in $\mathbf{x}$, and—optionally—(iii) feature vector $\mathbf{f}$,

[1] `https://github.com/ZurichNLP/coling2018-neural-transition-based-morphology`



	Transition a_t	**Output y**	**Stack**	**Buffer**
0	INSERT(BOS)	[]	[INSERT(BOS)]	[f, l, i, e, g, e, n, EOS]
1	COPY	[f]	[COPY, INSERT(BOS)]	[l, i, e, g, e, n, EOS]
2	COPY	[f, l]	[COPY, COPY, . . .]	[i, e, g, e, n, EOS]
3	DELETE	[f, l]	[DELETE,COPY, . . .]	[e, g, e, n, EOS]
4	DELETE	[f, l]	[DELETE, DELETE, . . .]	[g, e, n, EOS]
5	INSERT(o)	[f, l, o]	[INSERT(o), DELETE, . . .]	[g, e, n, EOS]
6	COPY	[f, l, o, g]	[COPY, INSERT(o), . . .]	[e, n, EOS]
7	DELETE	[f, l, o, g]	[DELETE, COPY, . . .]	[n, EOS]
8	DELETE	[f, l, o, g]	[DELETE, DELETE, . . .]	[EOS]
9	INSERT(EOS)	[f, l, o, g]		

Figure 2: Transduction of "fliegen" to "flog". (Above) Visualization of the system as it chooses a_3 = DELETE. (Below) Full transition sequence. Action a_0 is always fixed to INSERT(BOS).

which is the concatenation of the embedded morpho-syntactic features $\phi \subseteq \{f_h\}_{h=1}^{H}$ associated with this transduction: $\mathbf{f} = [F(f_1) \; ; \; \cdots \; ; \; F(f_H)]$ and $F(f_h) = F(0)$ if $f_h \notin \phi$.

To compute probabilities of transitions $\mathbf{a}_t$, we feed $\mathbf{s}_t$ through a softmax classifier:

$$P(a_t = k \mid \mathbf{a}_{<t}, \mathbf{x}, \Theta) = \text{softmax}_k(\mathbf{W} \cdot \mathbf{s}_t + \mathbf{b}) \tag{3}$$

Model parameters Θ include softmax classifier parameters $\mathbf{W}$ and $\mathbf{b}$, the embedding parameters, and the parameters of the encoder and decoder.

Edit actions Traditional transducers edit input sequence $\mathbf{x}$ into output sequence $\mathbf{y}$ by a sequence of single-character edit actions from the following set (Cotterell et al., 2014):

- DELETE: Read x_i and write nothing.
- SUBST(c) for $c \in \Sigma_y$: Read x_i and write c.
- INSERT(c) for $c \in \Sigma_y$: Write c and read nothing.
- COPY: Read x_i and write x_i.

Let INSERTS$_y$ be the set of all insertions with respect to Σ_y. We consider the following two action alphabets: Σ_a^{HA} = INSERTS$_y \cup \{$DELETE$\}$ and $\Sigma_a^{CA} = \Sigma_a^{HA} \cup \{$COPY$\}$.

Alphabet Σ_a^{HA} is from Aharoni and Goldberg (2017) and includes only the INSERT and DELETE actions. Both substitution and copying of c are expressed as an INSERT(c) followed by a DELETE.

Alphabet Σ_a^{CA} adds a designated COPY action to Σ_a^{HA}. Thus, copying x_i to the output sequence can be executed by one single action. This results in shorter and simpler action sequences dominated by COPY actions, following the observation that inflectional changes are typically small and most of $\mathbf{x}$ is preserved in $\mathbf{y}$.[2]

Action execution Operationally, reading x_i corresponds to popping its representation $\mathbf{h}_i$ from the top of the buffer. The transducer terminates when the buffer is empty and the latest action a_t is INSERT(EOS), where EOS is the end-of-sequence character. If we constrain the number of successive insertions to at most q, the transducer runs in $O(n)$ time, where n is the length of input $\mathbf{x}$.[3]

[2]We also experimented with extending Σ_a^{HA} and Σ_a^{CA} with actions for character substitutions. The resulting models perform similarly to models without substitutions, and so we do not report them here.

[3]In practice, we simply cap the number of actions at 150.

$$\begin{array}{ccccccc} \mathrm{f} & \mathrm{l} & \mathrm{i} & \mathrm{e} & \mathrm{g} & \mathrm{e} & \mathrm{n} \\ | & | & | & | & | & | & | \\ \mathrm{f} & \mathrm{l} & \mathrm{o} & \mathrm{g} & \epsilon & \epsilon & \epsilon \end{array} \qquad\qquad \begin{array}{ccccccc} \mathrm{f} & \mathrm{l} & \mathrm{i} & \mathrm{e} & \mathrm{g} & \mathrm{e} & \mathrm{n} \\ | & | & | & | & | & | & | \\ \mathrm{f} & \mathrm{l} & \mathrm{o} & \epsilon & \mathrm{g} & \epsilon & \epsilon \end{array}$$

Figure 3: Longest Common Substring (LCS, left) and Chinese Restaurant Process (CRP, right) character alignments for the same $\mathbf{x}$ and $\mathbf{y}$. Input sequence $\mathbf{x}$ is at the top, output sequence $\mathbf{y}$ at the bottom. A CRP aligner recovers this alignment given sufficient training data and number of iterations.

MLE training The model is trained to maximize the conditional log-likelihood of the data $D = \{(\mathbf{x}^{(l)}, \mathbf{a}^{(l)})\}_{l=1}^{N}$, which is an everywhere differentiable function of parameters Θ:

$$\mathcal{L}(D, \Theta) = \sum_{l=1}^{N} \sum_{t=1}^{m} \log P(a_t^{(l)} \mid \mathbf{a}_{<t}^{(l)}, \mathbf{x}^{(l)}, \Theta) \tag{4}$$

The gold action sequences $\mathbf{a}^{(l)}$ are computed by a deterministic algorithm from some character alignment: $\mathbf{a}^{(l)} = C_{\Sigma_a}^{Align}(\mathbf{x}^{(l)}, \mathbf{y}^{(l)})$. Figure 3 illustrates different character alignment algorithms that we use in our experiments. A simple procedure for the generation of gold actions from alphabet Σ_a^{CA} would call the following subroutine d on each pair of character alignment $(b_1, c_1), \ldots, (b_r, c_r)$ between input $\mathbf{x}$ and output $\mathbf{y}$, where $b_k \in \Sigma_x \cup \{\epsilon\}$ and $c_k \in \Sigma_y \cup \{\epsilon\}$ but not $b_k = c_k = \epsilon$:

$$d(b, c) = \begin{cases} \text{COPY}, & \text{if } b = c, \\ \text{DELETE}, & \text{if } c = \epsilon, \\ \text{INSERT(c)}, & \text{if } b = \epsilon, \\ \text{DELETE, INSERT(c)} & \text{otherwise} \quad \textit{\% substitution} \end{cases}$$

Applying this procedure to e.g. the CRP alignment from Figure 3, we obtain the following gold action sequence: COPY, COPY, DELETE, INSERT(o), DELETE, COPY, DELETE, DELETE.

Learning with exploration Training with MLE comes with a number of limitations. First, the model is not exposed to its own errors at training time: It makes predictions conditioned on gold-action histories, which is at odds with test time when the model has to condition on predicted actions. Second, MLE training increases the model's per-action likelihood, although at test time, the model's performance is assessed with sequence-level accuracy or edit distance. Both constitute well-known MLE training biases—the exposure bias and the loss-evaluation mismatch (Wiseman and Rush, 2016). Finally, we would like the model to be less dependent on the gold actions generated by the aligner, which is uninformed of the downstream task, and that at training, the model can choose alternative action sequences leading to correct predictions, if that helps it generalize.

To address all these issues at once, we train the model by minimizing the expected risk (Och, 2003; Smith and Eisner, 2006) of the actual training data $T = \{(\mathbf{x}^{(l)}, \mathbf{y}^{(l)})\}_{l=1}^{N}$:

$$\mathcal{R}(T, \Theta) = \sum_{l=1}^{N} \mathbb{E}_{\mathbf{a}|\mathbf{x}^{(l)};\Theta} \left[\Delta(\mathbf{y}, \mathbf{y}^{(l)})\right], \tag{5}$$

where $\mathbf{y}$ is computed from $\mathbf{a}$ and $\mathbf{x}$, and the risk is given by a combination of normalized Levenshtein distance (NLD) and accuracy:

$$\Delta(\mathbf{y}, \mathbf{y}^{(l)}) = \text{NLD}(\mathbf{y}, \mathbf{y}^{(l)}) - \mathbb{1}\{\mathbf{y} = \mathbf{y}^{(l)}\} \tag{6}$$

Thus, an action sequence $\mathbf{a}$ attains the lowest risk of -1 if its corresponding output sequence $\mathbf{y}$ is identical to $\mathbf{y}^{(l)}$ of the training sample and the highest risk of $+1$ if the number of edits from $\mathbf{y}$ to $\mathbf{y}^{(l)}$ equals the maximum of their lengths.

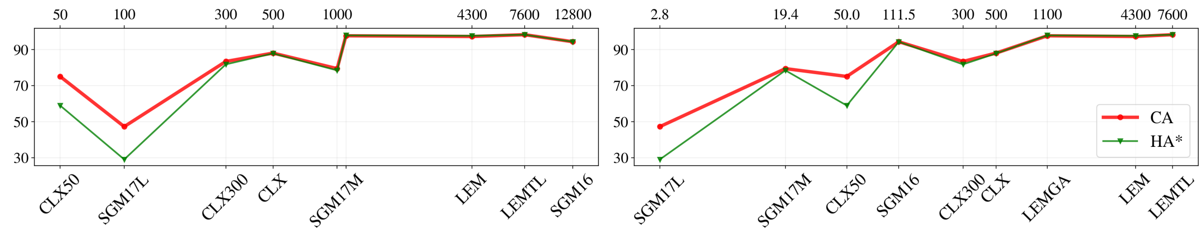


Figure 4: Accuracy as a function of dataset size (left) and the ratio of dataset size to the number of unique transformations (right) for selected experiments. A log scale is used for the X axis. CLX50/CLX300=average scores on CELEX with 50/300 samples (Figure 5), SGM16 =SIGMORPHON2016, SGM17L/SGM17M=SIGMORPHON2017-low/medium, LEM=average scores on lemmatization, LEMGA/LEMTL=lemmatization Irish/Tagalog.

Following Shen et al. (2016), we approximate the expectation under the posterior distribution $P(\mathbf{a} \mid \mathbf{x}^{(l)}; \Theta)$ with ancestral sampling from the model and re-normalize the sampled probability scores to get a new distribution Q:

$$\mathcal{R}(D, \Theta) \approx \sum_{l=1}^{N} \sum_{\mathbf{a} \in S(\mathbf{x}^{(l)})} Q(\mathbf{a} \mid \mathbf{x}^{(l)}; \Theta, \alpha)\; \Delta(\mathbf{y}, \mathbf{y}^{(l)}) \tag{7}$$

$$Q(\mathbf{a} \mid \mathbf{x}^{(l)}; \Theta, \alpha) = \frac{P(\mathbf{a} \mid \mathbf{x}^{(l)}; \Theta)^{\alpha}}{\sum_{\mathbf{a}' \in S(\mathbf{x}^{(l)})} P(\mathbf{a}' \mid \mathbf{x}^{(l)}; \Theta)^{\alpha}} \tag{8}$$

Here, $S(\mathbf{x}^{(l)})$ denotes the set of samples from $P(\mathbf{a} \mid \mathbf{x}^{(l)}; \Theta)$ and $\alpha \in \mathbb{R}$ is a hyper-parameter that controls for the peakedness of the new distribution Q.

3 Experiments

In the following experiments, we evaluate the performance of our model with an explicit copy action (referred to as CA) and show how it further improves with exploration training (-MRT).

Unless stated otherwise, our MLE models are trained on gold actions computed using Mans Hulden's Chinese Restaurant Process string-pair aligner (indicated as CRP)[4] and decoded with beam search. On some problems, we find a simple strategy, which heuristically maximizes the number of COPY actions, to work surprisingly well: The Longest Common Substring aligner (LCS) first aligns the longest common substring of $\mathbf{x}$ and $\mathbf{y}$ and then pads both strings to the same length.

MRT models are initialized with the corresponding MLE models and decoded with beam search. We found the best value of $\alpha = 1$ from $\{1, 0.1, 0.05\}$ on the CELEX-ALL task (§ 3.1) and used that for all other datasets as well.

We use the same embedding parameters for characters and insertion actions (i.e. $A(\text{INSERT}(b)) = E(b)$) to match closely the set-up of Aharoni and Goldberg (2017). In all our systems, the dimension of the character and action embeddings is 100, LSTM hidden layers are of size 200, and all LSTMs are single-layer. We use LSTMs with peephole connections and coupled input and forget gates (Greff et al., 2016). We optimize with ADADELTA (Zeiler, 2012) and update parameters at a single training sample (=batch size 1) during MLE training. For MRT, we build sets $S(\mathbf{x}^{(l)})$ by drawing twenty samples per training example. We mini-batch using these sets as batches. We include the gold action sequence (generated for MLE training) into the batch. We implement our models using DyNet (Neubig et al., 2017).

All experiments report exact accuracies. They are mean accuracies over single runs with different initializations, unless the model is an ensemble (marked with an -E suffix). The ensembles are built with majority voting over differently initialized runs of the same model.

[4]https://github.com/ryancotterell/sigmorphon2016/blob/master/src/baseline/align.c

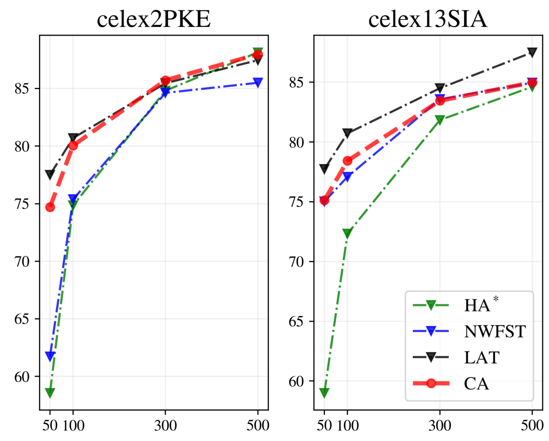


Figure 5: Learning curves on the CELEX dataset.

Model	13SIA	2PIE	2PKE	rP	Avg.
CELEX-BY-TASK					
LAT	**87.5**	93.4	87.4	84.9	88.3
NWFST	85.1	94.4	85.5	83.0	87.0
HA*	84.6	93.9	**88.1**	85.1	87.9
CA	85.0	94.5	88.0	84.9	88.1
HA*-MRT	84.8	94.0	88.1	85.2	88.0
CA-MRT	85.6	**94.6**	88.0	**85.3**	**88.4**
CELEX-ALL (ensembles)					
MED	83.9	95.0	87.6	84.0	87.2
HA	85.8	**95.1**	**89.5**	87.2	89.5
HA*-E	85.3	94.8	88.9	87.4	89.1
CA-E	85.8	94.9	88.8	86.7	89.1
HA*-MRT-E	85.8	95.0	89.2	**87.7**	89.4
CA-MRT-E	**86.7**	94.9	89.3	87.1	**89.5**

Table 1: Results on the CELEX dataset.

We evaluate our approaches on four standard morphological datasets and compare to the following published systems: (HA) the ensemble of five MLE models over Σ_a^{HA} of Aharoni and Goldberg (2017) as well as our re-implementation of a single model marked as HA*; (MED) the ensemble of five soft-attentional models of Kann and Schütze (2016) and an alternative implementation of the soft-attention approach, SOFT, by Aharoni and Goldberg (2017); (NWFST) the neural WFST model of Rastogi et al. (2016) and (LAT) the non-neural WFST with latent variables of Dreyer et al. (2008).

3.1 Morphological reinflection

The task is to map an inflected form $\mathbf{x}$ into another form $\mathbf{y}$ of that word given a feature specification ϕ for this transformation.

CELEX This dataset of German verbal morphology transformations was compiled by Dreyer et al. (2008) from the CELEX database (Baayen et al., 1993). It comprises four transformations (13SIA↦13SKE, 2PIE↦13PKE, 2PKE↦z, rP↦pA),[5] featuring such morphological phenomena as circumfixation, infixation, and irregular stem changes. The data are split into five folds, each with 500 training samples per transformation. We conduct two types of evaluation on these data. In the original experiment, which we call CELEX-BY-TASK, models are trained on each transformation separately, and scores are averaged over the folds. In the second experiment, CELEX-ALL, five single models are trained on all the 2,000 samples of one fold and then ensembled. Again, scores are averaged over the folds. As part of CELEX-BY-TASK, we additionally evaluate how our models perform on even fewer—50, 100, and 300—training samples on two tasks, 2PKE and 13SIA. CELEX could be considered a relatively simple dataset as the ratio of the number of training samples to the number of unique transformations is high, even though the overall training-data size is modest. On the other hand, most CELEX tasks require learning complex lexical properties such as the distinction between strong and weak verbs or prefix types.

3.2 Morphological inflection generation

Given a feature specification ϕ and a base form $\mathbf{x}$, the task is to generate the corresponding inflected form $\mathbf{y}$.

Sigmorphon 2017 The low (100 training samples) and medium (1,000 training samples) settings of the SIGMORPHON 2017 shared task data (Cotterell et al., 2017) feature fifty-two languages. The datasets contain extremely diverse language material and morphological transformations. Unlike CELEX, input $\mathbf{x}$ is always a dictionary form, however morphological changes are unrestricted. The low setting constitutes a very hard learning problem, with the ratio of training samples to unique transformations being 2.8 on

[5]Glossary: 13SIA=1st/3rd person singular indicative past; 13SKE=1st/3rd person singular subjunctive present; 2PIE=2nd person plural indicative present; 13PKE=1st/3rd plural subjunctive present; 2PKE=2nd person plural subjunctive present; z="zu" infinitive; rP=plural imperative; pA=past participle.

Model (averages)		**low**	**medium**	**Model** (ensembles)		**low**	**medium**
baseline		37.9	64.7			37.9	64.7
HA*	lcs	29.1	78.5	HA*-E	lcs	31.5	80.2
CA	lcs	47.3	79.5	CA-E	lcs	48.8	81.0
HA*	crp	23.9	75.4	HA*-E	crp	26.1	77.8
CA	crp	42.5	78.9	CA-E	crp	44.0	80.6
HA*-MRT	lcs	30.2	79.6	HA*-MRT-E	lcs	33.1	81.5
CA-MRT	lcs	**48.1**	80.3	CA-MRT-E	lcs	49.9	81.9
HA*-MRT	crp	25.3	78.1	HA*-MRT-E	crp	28.1	80.5
CA-MRT	crp	43.6	**81.1**	CA-MRT-E	crp	45.7	**82.9**
				HACM-E7		46.8	81.8
				HAEM-E7		48.5	80.3
				HA[EC]M-E15		**50.6**	82.8

Table 2: Results on the SIGMORPHON 2017 dataset.

average (SD $= 2.9$). In the medium setting, the mean number of unique transformations rises to 19.8 (SD $= 29.3$), with a minimum of 1.4 observed for Basque and a maximum of 200 for English.

For this dataset, we also show the results for the official baseline, a ruled-based system that is particularly strong in the low setting,[6] and the best systems of the shared task (Makarov et al., 2017).

Sigmorphon 2016 The SIGMORPHON 2016 shared task dataset (Cotterell et al., 2016) is the largest dataset. It comprises ten languages with about 12,800 training examples on average. The number of samples per transformation varies from 6 for Maltese to 198 for Hungarian, being 112 samples per transformation on average (SD $= 51.3$). In both SIGMORPHONS, we train five single models for each language.

3.3 Lemmatization

Given an inflected word form $\mathbf{x}$ (without any feature specification), the task is to predict the correct dictionary form $\mathbf{y}$. Following Dreyer (2011) and Rastogi et al. (2016), we evaluate our approach on a subset of the dataset by Wicentowski (2002). The data, split into ten folds, comprise four languages, with per-fold training sizes ranging on average from 1,100 for Irish to 7,635 for Tagalog. For each language, we train a separate model for each fold and then average the scores over the folds.

4 Results and Discussion

Generally, comparing the performance of CA and HA (or HA*), we observe that CA achieves great performance gains on small-sized problems while matching HA in the higher-resource setting (Figure 4).

4.1 Morphological reinflection

CA is a very competitive model on both CELEX-BY-TASK and CELEX-ALL, and adding exploration (CA-MRT) results in the strongest performance in both evaluations (Table 1). In contrast to HA*, in very low settings (Figure 5), CA performs not much worse than the only non-neural model, LAT. HA* and NWFST need around 300 training examples to start catching up, and the extremely low-resource conditions (50, 100) on 13SIA are especially troublesome for HA*. On CELEX-ALL, even with more training data, soft-attentional ensemble MED is typically much weaker, including tasks with infixation (2PKE) and circumfixation (rP).

Advancing further on most CELEX tasks is difficult due to morphological irregularities. As an example, examining the predictions of CA-MRT on one fold of the rP task reveals that the system largely fails to predict strong-verb participles (71% of the errors), conjugating 67% of them as if they were regular.

[6] `https://github.com/sigmorphon/conll2017/tree/master/baseline`

Model	RU	DE	ES	KA	FI	TR	HU	NV	AR	MT	Avg.
HA*	**91.32**	95.91	98.63	97.69	94.75	96.99	98.44	**90.57**	**93.93**	85.28	94.35
CA	90.81	**95.97**	**98.75**	**97.97**	**95.59**	**97.11**	**98.64**	89.74	93.59	**85.77**	**94.39**
						ensembles					
MED	91.46	95.80	98.84	98.50	95.47	98.93	96.80	91.48	**99.30**	**88.99**	95.56
SOFT	92.18	96.51	98.88	**98.88**	**96.99**	**99.37**	97.01	**95.41**	**99.30**	88.86	**96.34**
HA	**92.21**	**96.58**	**98.92**	98.12	95.91	97.99	96.25	93.01	98.77	88.32	95.61
HA*-E	91.95	96.28	98.85	97.90	95.78	97.55	98.77	92.14	95.08	87.82	95.21
CA-E	91.87	96.36	98.84	98.35	96.50	97.74	**98.90**	92.14	94.63	87.66	95.30

Table 3: Results on the SIGMORPHON 2016 dataset: ru=Russian, de=German, es=Spanish, ka=Georgian, fi=Finnish, tr=Turkish, hu=Hungarian, nv=Navaho, ar=Arabic, mt=Maltese.

4.2 Morphological inflection generation

Table 2 summarizes the results on the SIGMORPHON 2017 dataset. In the low setting, CA easily beats the baseline system, whereas HA* fails to do so. Our simple majority-vote ensemble CA-MRT-E over five models comes very close to the complex 15-strong ensemble HA[EC]M-E15 of Makarov et al. (2017), the best system of the shared task. Under a paired permutation test, the latter system is statistically significantly better ($p < 0.05$) on only twenty one languages.

In the medium setting, CA maintains the advantage, although the performance gap from HA* is much smaller. CA-MRT-E even outperforms the shared task's best system, although the gain is statistically significant for only ten languages. In both settings, MRT consistently improves the performance of both the HA* and CA models.

In the high-resource scenario of SIGMORPHON 2016 (Table 3), HA* and CA attain virtually identical results, occasionally outperforming the soft-attentional ensembles. Unlike HA, we use the same set of hyper-parameters (the dimension of embeddings, the number of hidden LSTM layers, etc.) for all of our experiments, which might explain that both our reimplementation HA* and CA perform less strongly here. Due to computational restrictions, we could not apply MRT to this dataset.

4.3 Lemmatization

On the lemmatization task (Table 4), CA strongly outperforms WFST models LAT and NWFST on average. Yet, the HA* reimplementation consistently delivers the best results on every language. The error analysis for English in Rastogi et al. (2016) mentions the tendency of their system, NWFST, to simply copy the inflected word over, which accounts for 25% of English-language errors. Given that CA also has a dedicated copy action, one might suspect that the inferior performance of CA compared to HA* for English and Basque would be due to excessive copying. An inspection of the incorrectly predicted lemmas reveals that both systems produce virtually the same number of copy errors. The difference in error counts is actually due to cases where the system modifies the inflected word form. For English, errors typically occur in strong verbs and verbs with graphemic alternations, as e.g. "oozing" gets incorrectly lemmatized as "ooz". The scores of over 97% on every language and the kind of unsolved cases, likely requiring external resources, suggest that this task should be considered solved.

As a final remark, we note that with the datasets at hand, performance attribution is often hampered by the lack of explicit characterization of morphological phenomena or lexical properties at the example level (we have derived some of these meta-data for the CELEX rP task). Given the difficulties interpreting neural models, computational morphology could arguably profit from challenge sets that have recently been gaining popularity in machine translation (Sennrich, 2017; Avramidis et al., 2018).

5 Related Work

Traditional models for morphological string transduction are discriminatively trained WFSTs (Cotterell et al., 2014; Dreyer et al., 2008; Eisner, 2002). The transducer defines eligible edit sequences for $\mathbf{x}$ (each implying a different monotonic character alignment), and its weights are expressed in terms of hand-crafted features. Rastogi et al. (2016) employ RNNs to parametrize the weights of a globally normalized

Model		basque	english	irish	tagalog	Avg.
Size		4.7K	3.9K	1.1K	7.6K	4.3K
LAT		93.6	96.9	**97.9**	88.6	94.2
NWFST		91.5	94.5	**97.9**	97.4	95.3
HA*	lcs	**97.0**	97.5	**97.9**	**98.3**	**97.7**
CA	lcs	96.3	96.9	97.7	**98.3**	97.3
HA*	crp	96.2	**97.7**	97.3	97.9	97.3
CA	crp	96.1	96.7	96.8	97.6	96.8

Table 4: Results on the lemmatization dataset.

WFST, thereby conditioning on global context. The powerful approach of Dreyer et al. (2008) adds latent variables to a globally normalized log-linear WFST to learn task-specific properties: a word's paradigm class and approximate morphological segmentation.

Enabling soft character alignment via a deterministic function of inputs (Kann and Schütze, 2016) has proven crucial to the success of seq2seq models first proposed for this task in Faruqui et al. (2016). In line with the traditional simplification of the task, other neural-network approaches treat hard monotone character alignment as a latent variable that the model marginalizes out using dynamic programming, while enabling unbounded dependencies in the output and permitting online generation (Yu et al., 2016; Graves, 2012). An appealing alternative to latent alignment is to learn from supervised alignment, an idea explored to train soft-attention models (Mi et al., 2016). For hard-attention models (Aharoni and Goldberg, 2017), training with an observed alignment is particularly simple as it results in learning from a single gold action sequence.

A state-transition system is an elegant, linear-time model for morphological string transduction, in which eligible monotonic edit sequences are implied by the semantics of the actions. As demonstrated on other tasks (Dyer et al., 2015; Andor et al., 2016), when provided with global context via RNNs, the model overcomes the limitations of a locally normalized conditional distribution, while retaining computational efficiency.

Using a single designated copy action in not new in morphological string transduction, e.g. the SIGMORPHON 2016 feature-based state-transition baseline uses COPY$[n]$, where n is the number of characters to copy. Biasing towards copy edits is crucial to the performance of the model of Rastogi et al. (2016). An alternative to the copy action is to introduce a binary latent variable that signals whether y_i is copied from x_i or generated (Gu et al., 2016; Gulcehre et al., 2016; See et al., 2017). Extending models with alignment variables with such a copying mechanism is simple as the the choice of which x_i has to be copied need not be modeled (Makarov et al., 2017): The copy variable points to the x_i that y_j is aligned with. This alternative requires learning additional model parameters, which could explain its somewhat worse performance on smaller-sized problems.

Minimum risk training (Smith and Eisner, 2006; Och, 2003) is one simple solution enabling exploration and addressing the loss-evaluation mismatch. The approach of Shen et al. (2016) closely relates to classical policy gradient methods in reinforcement learning (Edunov et al., 2018). A number of alternative methods have recently been proposed to address the MLE training biases in the context of seq2seq models (Andor et al., 2016; Wiseman and Rush, 2016; Ranzato et al., 2016; Rennie et al., 2017).

6 Conclusion

In a large-scale evaluation on different morphological tasks and languages, we show that a neural transition-based system over edit actions consistently outperforms state-of-the-art systems on morphological string transduction tasks in low- and medium-resource settings and is competitive on large training sets. Crucially, adding a designated action to copy the input character over to the output string helps the transition model generalize quickly from very few data points. Using a training procedure that enables exploration of the action space (e.g. minimum risk training) consistently improves the performance of our models as they are exposed to action sequences other than those proposed by the character aligner underlying the static oracle in the MLE training procedure.

Acknowledgements

We would like to thank Tatyana Ruzsics, Tanja Samardžić, Mathias Müller, Roee Aharoni, Pushpendre Rastogi, and the anonymous reviewers. Peter Makarov has been supported by European Research Council Grant No. 338875.

References

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

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *ACL*.

Eleftherios Avramidis, Vivien Macketanz, Arle Lommel, and Hans Uszkoreit. 2018. Fine-grained evaluation of quality estimation for machine translation based on a linguistically-motivated test suite. In *Proceedings of the 1st Workshop on Translation Quality Estimation and Automatic Post-Editing*. AMTA.

RH Baayen, R Piepenbrock, and H Van Rijn. 1993. The CELEX lexical database. LDC.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*.

Ryan Cotterell, Nanyun Peng, and Jason Eisner. 2014. Stochastic contextual edit distance and probabilistic FSTs. In *ACL*.

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 2016 Meeting of SIGMORPHON*.

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. The CoNLL-SIGMORPHON 2017 shared task: Universal morphological reinflection in 52 languages. In *Proceedings of the CoNLL-SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*. ACL.

Markus Dreyer, Jason R Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *ACL*.

Markus Dreyer. 2011. *A non-parametric model for the discovery of inflectional paradigms from plain text using graphical models over strings*. Ph.D. thesis, The Johns Hopkins University.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *ACL*.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. In *NAACL-HLT*.

Jason Eisner. 2002. Parameter estimation for probabilistic finite-state transducers. In *ACL*.

Manaal Faruqui, Yulia Tsvetkov, Graham Neubig, and Chris Dyer. 2016. Morphological inflection generation using character sequence to sequence learning. In *NAACL-HLT*.

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

Alex Graves. 2012. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*.

K. Greff, R. K. Srivastava, J. Koutnk, B. R. Steunebrink, and J. Schmidhuber. 2016. LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, PP(99).

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *CoRR*.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. In *ACL*.

Katharina Kann and Hinrich Schütze. 2016. Single-model encoder-decoder with explicit morphological representation for reinflection. In *ACL*.

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*. ACL.

Haitao Mi, Zhiguo Wang, and Abe Ittycheriah. 2016. Supervised attentions for neural machine translation. In *EMNLP*.

Mehryar Mohri. 2004. Weighted finite-state transducer algorithms. an overview. In *Formal Languages and Applications*, volume 148 of *Studies in Fuzziness and Soft Computing*. Springer Berlin Heidelberg.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. Dynet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980*.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *ACL*.

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

Pushpendre Rastogi, Ryan Cotterell, and Jason Eisner. 2016. Weighting finite-state transductions with neural context. In *NAACL-HLT*.

Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *CVPR 2017*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *ACL*.

Rico Sennrich. 2017. How grammatical is character-level neural machine translation? assessing mt quality with contrastive translation pairs. In *EACL*.

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*.

David A Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. In *COLING/ACL*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Richard Wicentowski. 2002. *Modeling and learning multilingual inflectional morphology in a minimally supervised framework*. Ph.D. thesis, Johns Hopkins University.

Sam Wiseman and Alexander M Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *EMNLP*.

Lei Yu, Jan Buys, and Phil Blunsom. 2016. Online segment to segment neural transduction. In *EMNLP*.

Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. arXiv:1212.5701.