Exact Hard Monotonic Attention for Character-Level Transduction

Shijie Wu[ə] and **Ryan Cotterell**[ə,ﬁ]
[ə]Department of Computer Science, Johns Hopkins University
[ﬁ]Department of Computer Science and Technology, University of Cambridge
`shijie.wu@jhu.edu`, `rdc42@cam.ac.uk`

Abstract

Many common character-level, string-to-string transduction tasks, e.g. grapheme-to-phoneme conversion and morphological inflection, consist almost exclusively of monotonic transduction. Neural sequence-to-sequence models with soft attention, which are non-monotonic, often outperform popular monotonic models. In this work, we ask the following question: Is monotonicity really a helpful inductive bias in these tasks? We develop a hard attention sequence-to-sequence model that enforces strict monotonicity and learns a latent alignment jointly while learning to transduce. With the help of dynamic programming, we are able to compute the exact marginalization over all monotonic alignments. Our models achieve state-of-the-art performance on morphological inflection. Furthermore, we find strong performance on two other character-level transduction tasks. Code is available at `https://github.com/shijie-wu/neural-transducer`.

1 Introduction

Many tasks in natural language can be treated as character-level, string-to-string transduction. The current dominant method is the neural sequence-to-sequence model with soft attention (Bahdanau et al., 2015; Luong et al., 2015). This method has achieved state-of-the-art results in a plethora of tasks, for example, grapheme-to-phoneme conversion (Yao and Zweig, 2015), named-entity transliteration (Rosca and Breuel, 2016) and morphological inflection generation (Cotterell et al., 2016). While soft attention is very similar to a traditional alignment between the source characters and target characters in some regards, it does not explicitly a distribution over alignments. On the other hand, neural sequence-to-sequence models with hard alignment (Xu et al., 2015; Wu et al., 2018) are analogous to the latent alignment in the classic IBM models for machine translation, which do model the alignment distribution explicitly (Brown et al., 1993).

The standard versions of both soft and hard attention are non-monotonic. However, if we look at the data in grapheme-to-phoneme conversion, named-entity transliteration, and morphological inflection—examples are shown in Fig. 1—we see that the tasks require almost exclusively monotonic transduction. Yet, counterintuitively, the state of the art in high resource morphological inflection is held by non-monotonic models (Cotterell et al., 2017)! Indeed, in a recent controlled experiment, Wu et al. (2018) found non-monotonic models (with either soft attention or hard alignment) outperform popular monotonic models (Aharoni and Goldberg, 2017) in the three above mentioned tasks. However, the inductive bias of monotonicity, if correct, should help learn a better model or, *at least*, learn the same model.

In this paper, we hypothesize that the underperformance of monotonic models stems from the lack of joint training of the alignments with the transduction. Generalizing the model of Wu et al. (2018) to enforce monotonic alignments, we show that, for all three tasks considered, monotonicity is a good inductive bias and jointly learning a monotonic alignment improves performance. We provide an exact, cubic-time, dynamic-programming inference algorithm to compute the log-likelihood and an approximate greedy decoding scheme. Empirically, our results indicate that, rather than the pipeline systems of Aharoni and Goldberg (2017) and Makarov et al. (2017), we should jointly train monotonic alignments with the transduction model, and, indeed, we set the single model state of the art on the task of morphological inflection.[1]

---

[1] The state of the art for morphological inflection is held by ensemble systems, much like parsing and other structured

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1530–1537
Florence, Italy, July 28 - August 2, 2019.

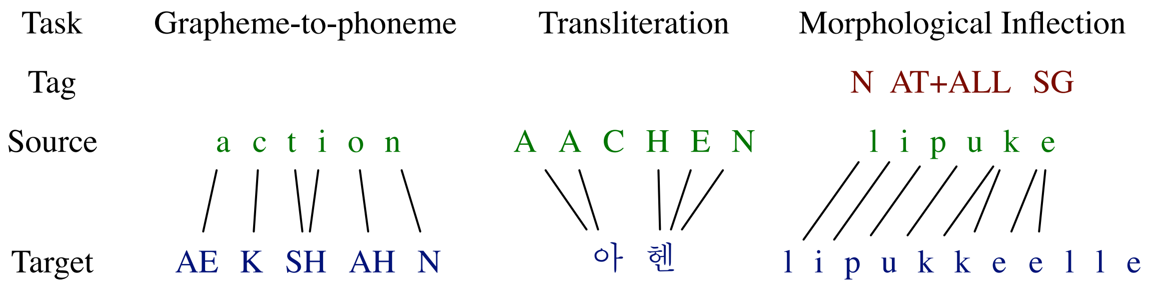

Figure 1: Example of source and target string for each task. Tag guides transduction in morphological inflection.

2 Hard Attention

2.1 Preliminary

We assume the source string $\mathbf{x} \in \Sigma_{\mathtt{x}}^*$ and the target string $\mathbf{y} \in \Sigma_{\mathtt{y}}^*$ have finite vocabularies $\Sigma_{\mathtt{x}} = \{x_1, \ldots, x_{|\Sigma_{\mathtt{x}}|}\}$ and $\Sigma_{\mathtt{y}} = \{y_1, \ldots, y_{|\Sigma_{\mathtt{y}}|}\}$, respectively. In tasks where the tag is provided, i.e., labeled transduction (Zhou and Neubig, 2017), we denote the tag as an ordered set $\mathbf{t} \in \Sigma_{\mathtt{t}}^*$ with a finite tag vocabulary $\Sigma_{\mathtt{t}} = \{t_1, \ldots, t_{|\Sigma_{\mathtt{t}}|}\}$. We define the set $A = \{1, \ldots, |\mathbf{x}|\}^{|\mathbf{y}|}$ to be set of all alignments from $\mathbf{x}$ to $\mathbf{y}$ where an alignment aligns each target character y_i to exactly one source character in $\mathbf{x}$. In other words, it allows zero-to-one[2] or many-to-one alignments between $\mathbf{x}$ and $\mathbf{y}$. For an $\mathbf{a} \in A$, $a_i = j$ refers to the event that $\mathbf{y}_i$ is aligned to $\mathbf{x}_j$, the i^{th} character of $\mathbf{y}$ and the j^{th} character of $\mathbf{x}$.

2.2 0^{th}-order Hard Attention

Hard attention was first introduced to the literature by Xu et al. (2015). We, however, follow Wu et al. (2018) and use a tractable variant of hard attention and model the probability of a target string $\mathbf{y}$ given an input string $\mathbf{x}$ as the following:

$$\begin{aligned} p(\mathbf{y} \mid \mathbf{x}) &= \sum_{\mathbf{a} \in A(\mathbf{x},\mathbf{y})} p(\mathbf{y}, \mathbf{a} \mid \mathbf{x}) \\ &= \underbrace{\sum_{\mathbf{a} \in A} \prod_{i=1}^{|\mathbf{y}|} p(\mathbf{y}_i \mid a_i, \mathbf{y}_{<i}, \mathbf{x})\, p(a_i \mid \mathbf{y}_{<i}, \mathbf{x})}_{\textit{exponential number of terms}} \\ &= \underbrace{\prod_{i=1}^{|\mathbf{y}|} \sum_{a_i=1}^{|\mathbf{x}|} p(\mathbf{y}_i \mid a_i, \mathbf{y}_{<i}, \mathbf{x})\, p(a_i \mid \mathbf{y}_{<i}, \mathbf{x})}_{\textit{polynomial number of terms}} \end{aligned} \tag{1}$$

where we show how one can rearrange the terms to compute the function in polynomial time.

The model above is exactly an 0^{th}-order neuralized hidden Markov model (HMM). Specifically, $p(\mathbf{y}_i \mid a_i, \mathbf{y}_{<i}, \mathbf{x})$ can be regarded as an emission distribution and $p(a_i \mid \mathbf{y}_{<i}, \mathbf{x})$ can be regarded as a transition distribution, which *does not* condition on the previous alignment. Hence, we will refer to this model as 0^{th}-order hard attention. The likelihood can be computed in $\mathcal{O}(|\mathbf{x}| \cdot |\mathbf{y}| \cdot |\Sigma_{\mathtt{y}}|)$ time.

2.3 1^{st}-order Hard Attention

To enforce monotonicity, hard attention with conditionally independent alignment decisions is not enough: The model needs to know the previous alignment position when determining the current alignment position. Thus, we allow the transition distribution to condition on previous one alignment $p(a_i \mid a_{i-1}, \mathbf{y}_{<i}, \mathbf{x})$ and it becomes a 1^{st}-order neuralized HMM. We display this model as a graphical model in Fig. 2. We will refer to it as 1^{st}-order hard attention. Generalizing the 0^{th}-order model, we define 1^{st}-order extension as:

$$\begin{aligned} p(\mathbf{y} \mid \mathbf{x}) &= \sum_{\mathbf{a} \in A(\mathbf{x},\mathbf{y})} p(\mathbf{y}, \mathbf{a} \mid \mathbf{x}) \\ &= \underbrace{\sum_{\mathbf{a} \in A} \prod_{i=1}^{|\mathbf{y}|} p(\mathbf{y}_i \mid a_i, \mathbf{y}_{<i}, \mathbf{x})\, p(a_i \mid a_{i-1}, \mathbf{y}_{<i}, \mathbf{x})}_{\textit{exponential number of terms}} \\ &= \underbrace{\prod_{i=1}^{|\mathbf{y}|} \sum_{a_{i-1}=1}^{|\mathbf{x}|} \sum_{a_i=1}^{|\mathbf{x}|} p(\mathbf{y}_i \mid a_i)\, p(a_i \mid a_{i-1}) \alpha(a_{i-1})}_{\textit{polynomial number of terms}} \end{aligned} \tag{2}$$

where $\alpha(a_{i-1})$ is the forward probability, calculated using the forward algorithm (Rabiner, 1989) with $\alpha(a_0, \mathbf{y}_0) = 1$, and $p(a_1 \mid a_0) = p(a_1 \mid$ `<BOS>`$, \mathbf{x})$ is the initial alignment distribution. For simplicity, we drop $\mathbf{y}_{<i}$ and $\mathbf{x}$ in $p(\mathbf{y}_i \mid a_i)$ and $p(a_i \mid a_{i-1})$. For completeness, we include the

---

prediction tasks. We present the new *best individual* system.

[2] Zero in the sense of non-character like `BOS` or `EOS`

recursive definition of the forward probability:

$$\alpha(a_i) = p(\mathbf{y}_i \mid a_i) \sum_{a_{i-1}=1}^{|\mathbf{x}|} p(a_i \mid a_{i-1})\, \alpha(a_{i-1})$$
$$\alpha(a_1) = p(\mathbf{y}_1 \mid a_1)\, p(a_1 \mid a_0)\, \alpha(a_0)$$

Thus, computation of the likelihood in our 1^{st}-order hard attention model is $\mathcal{O}(|\mathbf{x}|^2 \cdot |\mathbf{y}| \cdot |\Sigma_{\mathrm{y}}|)$.

Decoding at test time, however, is hard and we resort to a greedy scheme, described in Alg. 1. To see why it is hard, note that the dependence on $\mathbf{y}_{<i}$ means that we have a neural language model scoring the target string as it is being transduced. Because the dependence is unbounded, there will be no dynamic program that allows for efficient computation.

3 A Neural Parameterization with Enforced Monotonicity

The goal of this section is to take the 1^{st}-order model of §2 and show how we can straightforwardly enforce monotonic alignments. We will achieve this by adding structural zeros to the distribution, which will still allow us to perform efficient inference with dynamic programming. We follow the neural parameterization of Wu et al. (2018). The source string $\mathbf{x}$ is represented by a sequence of character embeddings vectors, which are fed into an encoder bidirectional LSTM (Hochreiter and Schmidhuber, 1997) to produce hidden state representations $\mathbf{h}^e_j$. The emission distribution $p(\mathbf{y}_i \mid a_i, \mathbf{y}_{<i}, \mathbf{x})$ depends on these encodings $\mathbf{h}^e_j$ and the decoder hidden states $\mathbf{h}^d_i$, produced by

$$\mathbf{h}^d_i = \text{LSTM}([\mathbf{e}^d(\mathbf{y}_{i-1}); \mathbf{h}^t], \mathbf{h}^d_{i-1})$$

where $\mathbf{e}^d$ encodes target characters into character embeddings. The tag embedding $\mathbf{h}^t$ is produced by

$$\mathbf{h}^t = \text{ReLU}(\mathbf{Y}\, [\mathbf{e}^t(\mathbf{t}_1); \ldots ; \mathbf{e}^t(\mathbf{t}_{|\Sigma_{\mathrm{t}}|})])$$

where $\mathbf{e}^t$ maps the tag $\mathbf{t}_k$ into tag embedding $\mathbf{h}^t_k \in \mathbb{R}^{d_t}$ or zero vector $\mathbf{0} \in \mathbb{R}^{d_t}$, depends on whether the tag $\mathbf{t}_k$ is presented. Note that $\mathbf{Y} \in \mathbb{R}^{d_t \times |\Sigma_{\mathrm{t}}|\, d_t}$ is a learned parameter. Also $\mathbf{h}^e_j \in \mathbb{R}^{2d_h}$, $\mathbf{h}^d_i \in \mathbb{R}^{d_h}$ and $\mathbf{h}^t \in \mathbb{R}^{d_t}$ are hidden states.

The Emission Distributon. All of our hard-attention models employ the same emission distribution parameterization, which we define below

$$p(\mathbf{y}_i \mid a_i, \mathbf{y}_{<i}, \mathbf{x}) = \text{softmax}\left(\mathbf{W}\, \mathbf{f}(\mathbf{h}^d_i, \mathbf{h}^e_{a_i})\right)$$
$$\mathbf{f}(\mathbf{h}^d_i, \mathbf{h}^e_{a_i}) = \tanh\left(\mathbf{V}\, [\mathbf{h}^d_i; \mathbf{h}^e_{a_i}]\right)$$

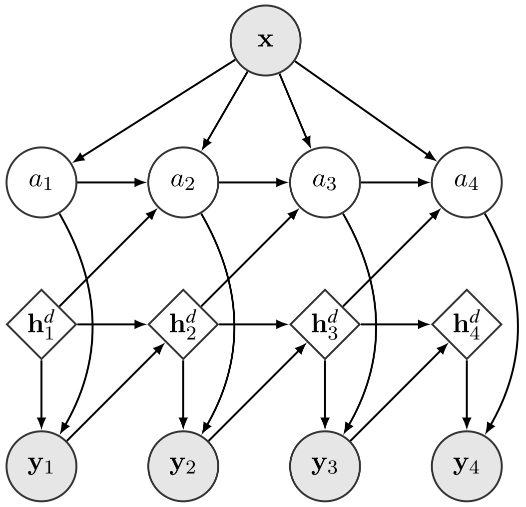

Figure 2: Our monotonic hard-attention model viewed as a graphical model. The circular nodes are random variables and the diamond nodes deterministic variables. We have omitted arcs from $\mathbf{x}$ to y_1, y_2, y_3 and y_4 for clarity (to avoid crossing arcs).

where $\mathbf{V} \in \mathbb{R}^{3d_h \times 3d_h}$ and $\mathbf{W} \in \mathbb{R}^{|\Sigma_{\mathrm{y}}| \times 3d_h}$ are learned parameters.

0^{th}-order Hard Attention. In the case of the 0^{th}-order model, the distribution is computed by a bilinear attention function with eq. (1)

$$p(a_i = j \mid \mathbf{y}_{<i}, \mathbf{x}) = \frac{\exp(\mathbf{h}^{d\top}_i\, \mathbf{T}\, \mathbf{h}^e_j)}{\sum_{j'=1}^{|\mathbf{x}|} \exp(\mathbf{h}^{d\top}_i\, \mathbf{T}\, \mathbf{h}^e_{j'})}$$

where $\mathbf{T} \in \mathbb{R}^{d_h \times 2d_h}$ is a learned parameter.

0^{th}-order Hard Monotonic Attention. We may enforce string monotonicity by zeroing out any non-monotonic alignment *without* adding any additional parameters, which can be done through adding structural zeros to the distribution as follows

$$p(a_i = j \mid a_{i-1} = j', \mathbf{y}_{<i}, \mathbf{x}) = \frac{\mathbb{1}\{j \geq j'\} \exp(\mathbf{h}^{d\top}_i\, \mathbf{T}\, \mathbf{h}^e_j)}{\sum_{j'=1}^{|\mathbf{x}|} \mathbb{1}\{j \geq j'\} \exp(\mathbf{h}^{d\top}_i\, \mathbf{T}\, \mathbf{h}^e_{j'})}$$

These structural zeros prevent the alignments from jumping backwards during transduction and, thus, enforce monotonicity. The parameterization is identical to the 0^{th}-order model up to the enforcement of the hard constraint with eq. (2).

1^{st}-order Hard Monotonic Attention. We may also generalize the 0^{th}-order case by *adding more parameters*. This will equip the model with a more expressive transition function. In this case, we take

Algorithm 1 Greedy decoding. (N is the maximum length of target string.)

```
for $i = 1, \cdots, N$ do
    if $i = 1$ then
        $y_i^* = \text{argmax}_{y_i} \sum_{a_i=1}^{|\mathbf{x}|} p(y_i \mid a_i) p(a_i \mid a_{i-1}) \, \alpha(a_0)$        ▷ Greedy decoding
        $\alpha(a_1) = p(y_1^* \mid a_1) \, p(a_1 \mid a_0) \, \alpha(a_0)$        ▷ Forward probability
    else
        $y_i^* = \text{argmax}_{y_i} \sum_{a_i=1}^{|\mathbf{x}|} p(y_i \mid a_i) \sum_{a_{i-1}=1}^{|\mathbf{x}|} p(a_i \mid a_{i-1}) \, \alpha(a_{i-1})$        ▷ Greedy decoding
        $\alpha(a_i) = p(y_i^* \mid a_i) \sum_{a_{i-1}=1}^{|\mathbf{x}|} p(a_i \mid a_{i-1}) \, \alpha(a_{i-1})$        ▷ Forward probability
    if $y_i^* = \texttt{EOS}$ then
        return $\mathbf{y}^*$
```

the 1^{st}-order hard attention to be an offset-based transition distribution similar to Wang et al. (2018):

$$p(a_i \mid a_{i-1}, \mathbf{y}_{<i}, \mathbf{x}) = \begin{cases} \text{softmax}(\mathbf{U}[\mathbf{h}_i^d; \mathbf{T}\, \mathbf{h}_{a_{i-1}}^e]) & 0 \leq \Delta \leq w \\ 0 & \text{otherwise} \end{cases}$$

where $\Delta = a_i - a_{i-1}$ is relative distance to previous attention position and $\mathbf{U} \in \mathbb{R}^{(w+1) \times 2d_h}$, a learned parameter. Note that, as before, we also enforce monotonicity as a hard constraint in this parameterization.

4 Related Work

There have been previous attempts to look at monotonicity in neural transduction. Graves (2012) first introduced the monotonic neural transducer for speech recognition. Building on this, Yu et al. (2016) proposes using a separated `shift/emit` transition distribution to allow more expressive model. Like us, they also consider morphological inflection and outperform a (weaker) soft attention baseline. Rastogi et al. (2016) offer a neural parameterization of a finite-state transducer, which implicitly encodes monotonic alignments. Instead of learning the alignments directly, Aharoni and Goldberg (2017) take the monotonic alignments from an external model (Sudoh et al., 2013) and train the neural model with these alignments. In follow-up work, Makarov et al. (2017) show this two-stage approach to be effective, winning the CoNLL-SIGMORPHON 2017 shared task on morphological inflection (Cotterell et al., 2017). Raffel et al. (2017) propose a stochastic monotonic transition process to allow sample-based *online* decoding.

5 Experiments

5.1 Experiments Design

Tasks. We consider three character-level transduction tasks: grapheme-to-phoneme conversion (Weide, 1998; Sejnowski and Rosenberg, 1987), named-entity transliteration (Zhang et al., 2015) and morphological inflection in high-esource setting (Cotterell et al., 2017).

Empirical Comparison. We compare (i) soft attention without input-feeding (SOFT) (Luong et al., 2015), (ii) 0^{th}-order hard attention (0-HARD) (Wu et al., 2018), (iii) 0^{th}-order monotonic hard attention (0-MONO) and (iv) 1^{st}-order monotonic hard attention (1-MONO). The SOFT, 0-HARD and 0-MONO models have an *identical* number of parameters, but the 1-MONO has more. All of them have approximately 8.6M parameters. Experimental details and hyperparameters may be found in App. A.

5.2 Experimental Findings

Finding #1: Morphological Inflection. The first empirical finding in our study is that we achieve single-model, state-of-the-art performance on the CoNLL-SIGMORPHON 2017 shared task dataset. The results are shown in Tab. 2. We find that the 1-MONO ties with the 0-MONO system, indicating the additional parameters do not add much. Both of these monotonic systems surpass the non-monotonic system 0-HARD and SOFT. We also report comparison to other top systems at the task in Tab. 1. The previous state-of-the-art model, Bergmanis et al. (2017), is a non-monotonic system that outperformed the monotonic system of Makarov et al. (2017). However, Makarov et al. (2017) is a pipeline system that took alignments from an existing aligner; such a system has no manner, by which it can recover from poor initial

Morphological Inflection	ACC
Silfverberg et al. (2017)	93.0
SOFT	93.4
Makarov et al. (2017)	93.9
0-HARD	94.5
Bergmanis et al. (2017)	94.6
Makarov and Clematide (2018)	94.6
0-MONO	**94.8**
1-MONO	**94.8**

Table 1: Average dev performance on morphological inflection of our models against single models from the 2017 shared task. All systems are single model, i.e., *without* ensembling. Why dev? No participants submitted single-model systems for evaluation on test and the best systems were not open-sourced, constraining our comparison. Note we report numbers from their paper.[3]

alignment. We show that *jointly* learning monotonic alignments lead to improved results.

Finding #2: Effect of Strict Monotonicity. The second finding is that by comparing SOFT, 0-HARD, 0-MONO in Tab. 2, we observe 0-MONO outperforms 0-HARD and 0-HARD in turns outperforms SOFT in all three tasks. This shows that monotonicity should be enforced strictly since strict monotonicity does not hurt the model. We contrast this to the findings of Wu et al. (2018), who found the non-monotonic models outperform the monotonic ones; this suggests strict monotonicity is more helpful when the model is allowed to learn the alignment distribution jointly.

Finding #3: Do Additional Parameters Help? The third finding is that 1-MONO has a more expressive transition distribution and, thus, outperforms 0-MONO and 0-HARD in G2P. However, it performs as well as or worse on the other tasks. This tells us that the additional parameters are not always necessary for improved performance. Rather, it is the hard constraint that matters—not the more expressive distribution. However, we remark that enforcing the monotonic constraint does come at an additional computational cost: an additional factor $\mathcal{O}(|\mathbf{x}|)$.

6 Conclusion

	Trans		G2P		MorInf	
	ACC	**MFS**	**WER**	**PER**	**ACC**	**MLD**
SOFT	40.4	0.893	29.3	0.071	92.9	0.157
0-HARD	$41.1^{\star}$	0.894	$29.2^{\star}$	0.070	$93.8^{\star}$	0.126
0-MONO	$\mathbf{41.2}^{\star}$	**0.895**	$29.0^{\star\times}$	0.072	$\mathbf{94.4}^{\star\times}$	**0.113**
1-MONO	40.8	0.893	$\mathbf{28.2}^{\star\times\dagger}$	**0.069**	$\mathbf{94.4}^{\star\times}$	0.116

Table 2: Average test performance of namded-entity transliteration (Trans), grapheme-to-phoneme conversion (G2P) and morphological inflection (MorInf). First group has exactly same number of parameter while the second group has slightly more parameter. $\star$, $\times$ and $\dagger$ indicate statistical significant improvement against SOFT, 0-HARD and 0-MONO on language-level paired permutation test ($p < 0.05$).

We expand the hard-attention neural sequence-to-sequence model of Wu et al. (2018) to enforce monotonicity. We show, empirically, that enforcing monotonicity in the alignments found by hard attention models helps significantly, and we achieve state-of-the-art performance on the morphological inflection using data from the CoNLL-SIGMORPHON 2017 shared task. We isolate the effect of monotonicity in a controlled experiment and show monotonicity is a useful hard constraint for three tasks, and speculate previous underperformance is due to a lack of joint training.

Acknowledgments

The final author acknowledges a Facebook Fellowship.

References

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

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*, volume abs/1409.0473.

Toms Bergmanis, Katharina Kann, Hinrich Schütze, and Sharon Goldwater. 2017. Training data augmentation for low-resource morphological inflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 31–39, Vancouver. Association for Computational Linguistics.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

---

[3] Some numbers are obtained by contacting authors.

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*, Vancouver, Canada. 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. Association for Computational Linguistics.

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

Sepp Hochreiter and Jürgen Schmidhuber. 1997. *Neural Computation*, 9(8):1735–1780.

Diederick P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

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

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

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

Lawrence R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Colin Raffel, Minh-Thang Luong, Peter J. Liu, Ron J. Weiss, and Douglas Eck. 2017. Online and linear-time attention by enforcing monotonic alignments. In *International Conference on Machine Learning (ICML)*, pages 2837–2846.

Pushpendre Rastogi, Ryan Cotterell, and Jason Eisner. 2016. Weighting finite-state transductions with neural context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 623–633, San Diego, California. Association for Computational Linguistics.

Mihaela Rosca and Thomas Breuel. 2016. Sequence-to-sequence neural network models for transliteration. *arXiv preprint arXiv:1610.09565*.

Terrence J. Sejnowski and Charles R. Rosenberg. 1987. Parallel networks that learn to pronounce english text. *Complex Systems*, 1.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99, Vancouver. Association for Computational Linguistics.

Katsuhito Sudoh, Shinsuke Mori, and Masaaki Nagata. 2013. Noise-aware character alignment for bootstrapping statistical machine transliteration from bilingual corpora. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 204–209.

Weiyue Wang, Derui Zhu, Tamer Alkhouli, Zixuan Gan, and Hermann Ney. 2018. Neural hidden Markov model for machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 377–382. Association for Computational Linguistics.

R.L. Weide. 1998. The Carnegie Mellon pronouncing dictionary.

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

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the 32nd International Conference on Machine Learning, ICML*, pages 2048–2057.

Kaisheng Yao and Geoffrey Zweig. 2015. Sequence-to-sequence neural net models for grapheme-to-phoneme conversion. In *INTERSPEECH 2015*, pages 3330–3334, Dresden, Germany.

Lei Yu, Jan Buys, and Phil Blunsom. 2016. Online segment to segment neural transduction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1307–1316.

Min Zhang, Haizhou Li, Rafael E. Banchs, and A. Kumaran. 2015. Whitepaper of news 2015 shared task on machine transliteration. In *NEWS@ACL*.

Chunting Zhou and Graham Neubig. 2017. Multi-space variational encoder-decoders for semi-supervised labeled sequence transduction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 310–320, Vancouver, Canada. Association for Computational Linguistics.

A Experimental Details

A.1 Tasks.

We ask the authors of Wu et al. (2018) for the split data of grapheme-to-phoneme conversion (CMUDict (Weide, 1998) and NetTalk (Sejnowski and Rosenberg, 1987)) and NEWS 2015 shared task on named-entity transliteration. In named-entity transliteration, we only run experiments on 11 language pairs.[4]

Grapheme-to-Phoneme Conversion is evaluated by word error rate (WER) and phoneme error rate (PER) (Yao and Zweig, 2015), where PER is the edit distance divided by the length of the phonemes. Named-entity transliteration is evaluated by word accuracy (ACC) and mean F-score (MFS) (Zhang et al., 2015). F-score is computed by

$$\begin{aligned}\mathrm{LCS}(c,r) &= \frac{1}{2}(|c|+|r|-\mathrm{ED}(c,r))\\ R_i &= \frac{\mathrm{LCS}(c_i,r_i)}{|r_i|}\\ P_i &= \frac{\mathrm{LCS}(c_i,r_i)}{|c_i|}\\ \mathrm{FS}_i &= 2\frac{R_i\times P_i}{R_i+P_i}\end{aligned}$$

where r_i and c_i is the i-th reference and prediction and $ED(c,r)$ is the edit distance between c and r. Morphological inflection is evaluated by word accuracy (ACC) and average edit distance (MLD) (Cotterell et al., 2017).

A.2 Parameterization.

For completeness, we also include the parameterization of soft attention.

$$\begin{aligned}p(\mathbf{y}_i \mid \mathbf{y}_{<i},\mathbf{x}) &= \mathrm{softmax}\left(\mathbf{W}\,\mathbf{f}(\mathbf{h}_i^d,\mathbf{c}_i)\right)\\ \mathbf{c}_i &= \sum_{j=1}^{|\mathbf{x}|}\alpha_{ij}\,\mathbf{h}_j^e\\ \alpha_{ij} &= \frac{\exp(e_{ij})}{\sum_{j=1}^{|\mathbf{x}|}\exp(e_{ij})}\\ e_{ij} &= \mathbf{h}_i^{d\top}\,\mathbf{T}\,\mathbf{h}_j^e\end{aligned}$$

The dimension of character and tag embedding are 200 and 40, respectively. The encoder and decoder LSTM both have 400 hidden dimensions (d_h). We also have a 2 layer encoder LSTM. We have 0.4 dropout in embedding and encoder LSTM. The w in 1st-order hard monotonic attention model is 4.

A.3 Optimization.

The model is trained with Adam (Kingma and Ba, 2015) and the learning rate is 0.001. We halve the learning rate whenever the development log-likelihood increase and we stop early when the learning rate reaches 0.00001. We apply gradient clipping with maximum gradient norm 5. The models are selected by development evaluation metric and decoded greedily since no improvements are observed when using beam search (Wu et al., 2018).

---

[4] Ar–En, En–Ba, En–Hi, En–Ja, En–Ka, En–Ko, En–Pe, En–Ta, En–Th, Jn–Jk and Th–En.