End-to-End Non-Autoregressive Neural Machine Translation with Connectionist Temporal Classification

Jindřich Libovický and **Jindřich Helcl**
Charles University, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Malostranské náměstí 25, 118 00 Prague, Czech Republic
{libovicky, helcl}@ufal.mff.cuni.cz

Abstract

Autoregressive decoding is the only part of sequence-to-sequence models that prevents them from massive parallelization at inference time. Non-autoregressive models enable the decoder to generate all output symbols independently in parallel. We present a novel non-autoregressive architecture based on connectionist temporal classification and evaluate it on the task of neural machine translation. Unlike other non-autoregressive methods which operate in several steps, our model can be trained end-to-end. We conduct experiments on the WMT English-Romanian and English-German datasets. Our models achieve a significant speedup over the autoregressive models, keeping the translation quality comparable to other non-autoregressive models.

1 Introduction

Parallelization is the key ingredient for making deep learning models computationally tractable. While the advantages of parallelization are exploited on many levels during training and inference, autoregressive decoders require sequential execution.

Training and inference algorithms in sequence-to-sequence tasks with recurrent neural networks (RNNs) such as neural machine translation (NMT) have linear time complexity w.r.t. the target sequence length, even when parallelized (Sutskever et al., 2014; Bahdanau et al., 2014).

Recent approaches such as convolutional sequence-to-sequence learning (Gehring et al., 2017) or self-attentive networks a.k.a. the Transformer (Vaswani et al., 2017) replace RNNs with parallelizable components in order to reduce the time complexity of the training. In these models, the decoding is still sequential, because the probability of emitting a symbol is conditioned on the previously decoded symbols.

In non-autoregressive decoders, the inference algorithm can be parallelized because the decoder does not depend on its previous outputs. The apparent advantage of this approach is the near-constant time complexity achieved by the parallelization. On the other hand, the drawback is that the model needs to explicitly determine the target sentence length and reorder the state sequence before it starts generating the output. In the current research contributions on this topic, these parts are trained separately and the inference is done in several steps.

In this paper, we propose an end-to-end non-autoregressive model for NMT using Connectionist Temporal Classification (CTC; Graves et al. 2006). The proposed technique achieves promising results on translation between English-Romanian and English-German on the WMT News task datasets.

The paper is organized as follows. In Section 2, we summarize the related work on non-autoregressive NMT. Section 3 describes the architecture of our proposed model. Section 4 presents details of the conducted experiments. The results are discussed in Section 5. We conclude and present ideas for future work in Section 6.

2 Non-Autoregressive NMT

In this section, we describe two methods for non-autoregressive decoding in NMT. Both of them are based on the Transformer architecture (Vaswani et al., 2017), with the encoder part unchanged.

Gu et al. (2017) use a latent fertility model to copy the sequence of source embeddings which is then used for the target sentence generation. The fertility (i.e. the number of target words for each source word) is estimated using a softmax on the encoder states. In the decoder, the input embeddings are repeated based on their fertility.

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3016–3021
Brussels, Belgium, October 31 - November 4, 2018.

The decoder has the same architecture as the encoder plus the encoder attention. The best results were achieved by sampling fertilities from the model and then rescoring the output sentences using an autoregressive model. The reported inference speed of this method is 2–15 times faster than of a comparable autoregressive model, depending on the number of fertility samples.

Lee et al. (2018) propose an architecture with two decoders. The first decoder generates a candidate translation from a source sentence padded to an estimated target length. The explicit length estimate is done with a softmax over possible sentence lengths (up to a fixed maximum). The output of the first decoder is then fed as an input to the second decoder. The second decoder is used as a denoising auto-encoder and can be applied iteratively. Both decoders have the same architecture as in Gu et al. (2017). They achieved a speedup of 16 times over the autoregressive model with a single denoising iteration. They report the best result in terms of BLEU (Papineni et al., 2002) after 20 iterations with almost no inference speedup compared to their autoregressive baseline.

3 Proposed Architecture

Similar to the previous work (Gu et al., 2017; Lee et al., 2018), our models are based on the Transformer architecture as described by Vaswani et al. (2017), keeping the encoder part unchanged. Figure 1 illustrates our method and highlights the differences from the Transformer model.

In order to generate output words in parallel, we formulate the translation as a sequence labeling problem. Neural architectures used for encoding input in NLP tasks usually generate sequences of hidden states of the same or shorter length as the input sequence. For this reason, we cannot apply the sequence labeling directly over the states because the target sentence might be longer than the source sentence.

To enable the labeler to generate sentences that are longer than the source sentence, we project the encoder output states $\mathbf{h}$ into a k-times longer sequence $\mathbf{s}$, such that:

$$s_{ci+b} = \left(W_{\text{spl}} h_c + b_{\text{spl}}\right)_{bd:(b+1)d} \qquad (1)$$

for $b = 0 \dots k-1$, and $c = 0 \dots T_x$ where d is the Transformer model dimension, T_x is the length of the source sentence, and $W_{\text{spl}} \in \mathbb{R}^{d \times kd}$ and $b_{\text{spl}} \in \mathbb{R}^{kd}$ are trainable projection parameters. In other words, after a linear projection, each state is sliced to k vectors, creating a sequence of length kT_x.

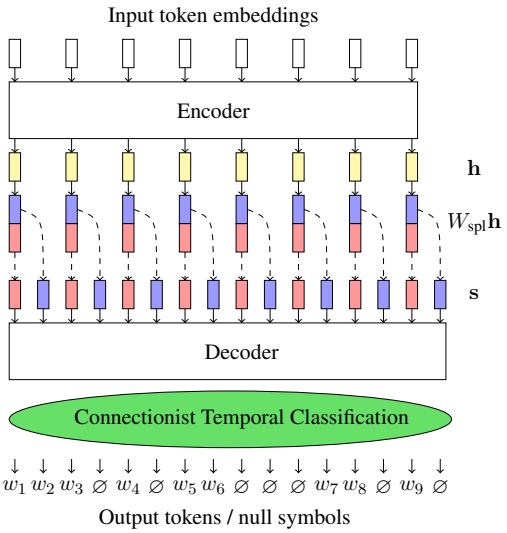

Figure 1: Scheme of the proposed architecture. The part between the encoder and the decoder is expressed by Equation 1.

In the next step, we process the sequence $\mathbf{s}$ with a decoder. Unlike the Transformer architecture, our decoder does not use the temporal mask in the self-attention step.

Finally, the decoder states are labeled either with an output token or a null symbol. The number of combinations of the possible positions of the null symbols in the output sequence given reference sequence length T_y is $\binom{kT_x}{T_y}$. Because there is no prior alignment between the input and output symbols, we consider all output sequences that yield the correct output in the loss function. Because summing the exponential number of combinations directly is not tractable, we we use the CTC loss (Graves et al., 2006) which employs dynamic programming to compute the negative log-likelihood of the output sequence, summed over all the combinations.

The loss can be computed using a linear algorithm similar to training Hidden Markov Models (Rabiner, 1989). The algorithm computes and stores partial log-probabilities sums for all prefixes and suffixes of the output symbol sequence using dynamic programming. The table of precomputed log-probablities allows us to compute the probability of being a part of a correct output sequence by combining the log-probabilities of its prefix and suffix.

An appealing property of training using the

CTC loss is that the models support left-to-right beam search decoding by recombining prefixes that yield the same output. Unlike the greedy decoding this can no longer be done in parallel. However, the linear computation is in theory still faster than autoregressive decoding.

4 Experiments

We experiment with three variants of this architecture. All of them have the same total number of layers. First, the *deep encoder* uses a stack of self-attentive layers only. We apply the state splitting and the labeler on the output of the last encoder layer. In contrast to Figure 1, this variant omits the decoder part. Second, the *encoder-decoder* consists of two stacks of self-attentive layers – encoder and decoder. The outputs of the encoder are transformed using Equation 1 and processed by the decoder. In each layer, the decoder part attends to the encoder output. Third, we extend the encoder-decoder variant with positional encoding (Vaswani et al., 2017). The positional encoding vectors are added to the decoder input s.

In all the experiments, we used the same hyper-parameters. We set the model dimension to 512 and the feed-forward layer dimension to 4096. We use multi-head attention with 16 heads. In the deep encoder setup, we use 12 layers in the encoder, in the encoder-decoder setup, we use 6 layers for the encoder and 6 layers for the decoder. We set the split factor k to 3, so the encoder states are projected to vectors of 1536 units.

We conduct our experiments on English-Romanian and English-German translation. These language pairs were selected by the authors of the previous work because the training datasets for these language pairs are of considerably different sizes. We follow these choices in order to present comparable results.

For English-Romanian experiments, we used the WMT16 (Bojar et al., 2016) news dataset. The training data consists of 613k sentence pairs, validation 2k and test 2k. We used a shared vocabulary of 38k wordpieces (Wu et al., 2016; Johnson et al., 2017).

The English-German dataset consists of 4.6M training sentence pairs from WMT competitions. As a validation set, we used the test set from WMT13 (Bojar et al., 2013), which contains 3k sentence pairs. To enable comparison to other non-autoregressive approaches, we evaluate our models on the test sets from WMT14 (Bojar et al., 2014) with 3k sentence pairs and WMT15 (Bojar et al., 2015) with 2.1k sentence pairs. As in the previous case, we used shared vocabulary for both languages which contained 41k wordpieces.

The experiments were conducted using Neural Monkey[1] (Helcl and Libovický, 2017). We evaluate the models using BLEU score (Papineni et al., 2002) as implemented in SacreBLEU,[2] originally a part of the Sockeye toolkit (Hieber et al., 2017).

5 Results

Quantitative results are tabulated in Table 1. In general, our models achieve a similar performance to other non-autoregressive models. In case of English-German, our results in both directions are comparable on the WMT 14 test set and slightly better on the WMT 15 test set. This might be given by the fact that our autoregressive baseline performs better for this language pair than for English-Romanian.

The encoder-decoder setup outperforms the deep encoder setup. Including positional encoding seems beneficial when translating into German. Weight averaging from the 5 models with the highest validation score during the training improves the performance consistently.

We performed a manual evaluation on 100 randomly sampled sentences from the English-German test sets in both directions. The results of the analysis are summarized in Table 2.

Non-autoregressive translations of sentences that had errors in the autoregressive translation were often incomprehensible. In general, less than a quarter of the sentences was completely correct and over two thirds (one half in the de→en direction) were comprehensible. The most frequent errors include omitting verbs at the end of German sentences and corruption of named entities and infrequent words that are represented by more wordpieces. Most of these errors can be attributed to insufficient language-modeling capabilities of the model. The results suggest that integrating an external language model into an efficient beam search implementation could boost the translation quality while preserving the speedup over the auto-regressive models.

We also evaluated the translations using sentence-level BLEU score (Chen and Cherry,

[1] https://github.com/ufal/neuralmonkey
[2] https://github.com/mjpost/sacreBLEU

	WMT 16		WMT 14		WMT 15	
	en-ro	ro-en	en-de	de-en	en-de	de-en
autoregressive $b = 1$	31.93	31.55	22.71	26.39	23.40	26.49
autoregressive $b = 4$	32.40	32.06	23.45	27.02	24.12	27.05
Gu et al. (2017) greedy	27.29	29.06	17.69	21.47	—	—
Gu et al. (2017) NPD w/ 100 samples	29.79	31.44	19.17	23.20	—	—
Lee et al. (2018) 1 iteration	24.45	23.73	—	—	12.65	14.48
Lee et al. (2018) best result	29.49	30.41	—	—	19.13	21.69
our autoregressive $b = 1$	21.19	29.64	22.94	28.58	25.12	28.89
deep encoder	17.33	22.85	12.21	12.53	13.14	18.34
+ weight averaging	18.47	**24.68**	14.65	16.72	16.74	18.47
+ beam search	18.70	25.28	15.19	17.58	17.59	18.70
encoder-decoder	18.51	22.37	13.29	17.98	16.01	19.55
+ weight averaging	**19.54**	24.67	16.56	**18.64**	19.46	**21.74**
+ beam search	19.81	25.21	17.09	18.80	20.59	22.55
encoder-decoder w/ pos. encoding	18.13	22.75	12.51	11.35	15.35	19.30
+ weight averaging	19.31	24.21	**17.37**	18.07	**20.30**	19.64
+ beam search	19.93	24.71	17.68	19.80	20.67	20.43

Table 1: Quantitative results in terms of BLEU score of the proposed methods compared to other non-autoregressive models. Note that our method uses only a single pass through the network and should be compared with greedy decoding by Gu et al. (2017) and 1 model iteration by Lee et al. (2018).

	en → de		de → en	
	AR	NAR	AR	NAR
Correct	65	23	67	13
Comprehensible	93	71	92	51
Too short	1	16	0	36
Missing verb	4	35	0	8
Corrupt. named entity	1	27	8	21
Corrupt. other words	1	20	0	46

Table 2: Results of manual evaluation of the autoregressive (AR) and non-autoregressive (NAR) models (in percents).

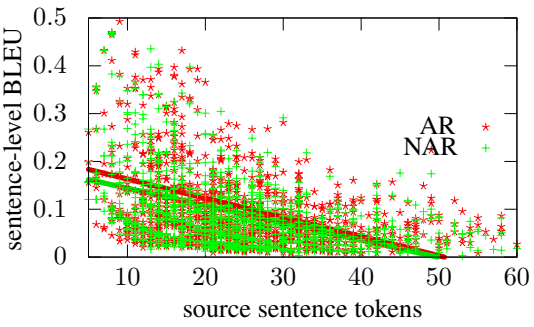

Figure 2: Comparison of the sentence-level BLEU of our English-to-German autoregresssive (AR) and non-autoregressive (NAR) models given the length of the source sentence.

2014) and measure the Pearson correlation with the length of the source sentence and the number of null symbols generated in the output. With a growing sentence length, the scores degrade more in the non-autoregressive model ($r = -0.42$) than in its autoregressive counterpart ($r = -0.39$). The relation between sentence-level BLEU and the source length is plotted in Figure 2. The sentence-level score is mildly correlated with the number of null symbols in the non-autoregressive output ($r = 0.15$). This suggests that increasing the splitting factor k in Equation 1 might improve the model performance. However, it also reduces the efficiency in terms of GPU memory usage.

Figure 3 shows the comparison of the decoding time by autoregressive and non-autoregressive models. The average times of decoding a single sentence are shown in Table 3. We suspect that the small difference between CPU and GPU times in the non-autoregressive setup is caused by the CPU-only implementation of the CTC decoder in TensorFlow (Abadi et al., 2015).

6 Conclusions

In this work, we presented a novel method for training a non-autoregressive model end-to-end using connectionist temporal classification. We evaluated the proposed method on neural machine

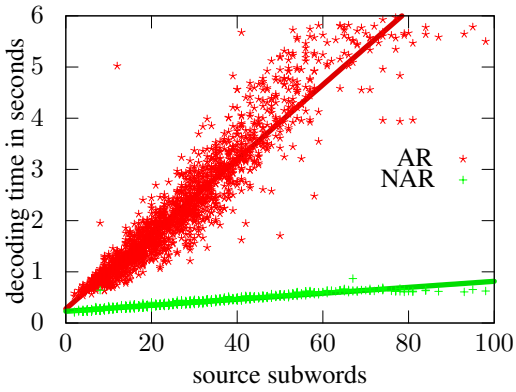

Figure 3: Comparison of CPU decoding time by our autoregressive (AR) and non-autoregressive (NAR) models based on the source sentence length.

	CPU	GPU
AR, $b = 1$	2247 ms	1200 ms
NAR	386 ms	350 ms

Table 3: Average per sentence decoding time for en-de translation.

translation in two language pairs and compared the results to the previous work.

In general, the results match the translation quality of equivalent variants of the models presented in the previous work. The BLEU score is usually around 80–90% of the score of the autoregressive baselines. We measured a 4-times speedup compared to our autoregressive baseline, which is a smaller gain than reported by the authors of the previous work. We suspect this might be due to a larger overhead with data loading and processing in Neural Monkey compared to Tensor2Tensor (Vaswani et al., 2018) used by others.

As a future work, we can try to improve the performance of the model by iterative denoising as done by Lee et al. (2018) while keeping the non-autoregressive nature of the decoder.

Another direction of improving the model might be efficient implementation of beam search which can contain rescoring using an external language model as often done in speech recognition (Graves et al., 2013). The non-autoregressive model would play a role a of the translation model in the traditional statistical MT problem decomposition.

Acknowledgments

This research has been funded by the Czech Science Foundation grant no. P103/12/G084, Charles University grant no. 976518 and SVV project no. 260 453.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation (WMT16). In *Proceedings of the First Conference on Machine Translation (WMT). Volume 2: Shared Task Papers*, volume 2, pages 131–198, Stroudsburg, PA, USA. Association for Computational Linguistics, Association for Computational Linguistics.

Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia

Specia, and Marco Turchi. 2015. Findings of the 2015 workshop on statistical machine translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisbon, Portugal. Association for Computational Linguistics.

Boxing Chen and Colin Cherry. 2014. A systematic comparison of smoothing techniques for sentence-level bleu. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 362–367, Baltimore, Maryland, USA. Association for Computational Linguistics.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 1243–1252.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM.

Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. 2013. Hybrid speech recognition with deep bidirectional lstm. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pages 273–278. IEEE.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2017. Non-autoregressive neural machine translation. *CoRR*, abs/1711.02281.

Jindřich Helcl and Jindřich Libovický. 2017. Neural Monkey: An open-source tool for sequence learning. *The Prague Bulletin of Mathematical Linguistics*, (107):5–17.

Felix Hieber, Tobias Domhan, Michael Denkowski, David Vilar, Artem Sokolov, Ann Clifton, and Matt Post. 2017. Sockeye: A Toolkit for Neural Machine Translation. *arXiv preprint arXiv:1712.05690*.

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

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. *CoRR*, abs/1802.06901.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. 2018. Tensor2tensor for neural machine translation. *CoRR*, abs/1803.07416.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6000–6010. Curran Associates, Inc.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.