

Transformer-Based Direct Hidden Markov Model for Machine Translation

Weiye Wang Zijian Yang Yingbo Gao Hermann Ney

Human Language Technology and Pattern Recognition Group

Computer Science Department

RWTH Aachen University

{wwang | zyang | ygao | ney}@cs.rwth-aachen.de

Abstract

The neural hidden Markov model has been proposed as an alternative to attention mechanism in machine translation with recurrent neural networks. However, since the introduction of the transformer models, its performance has been surpassed. This work proposes to introduce the concept of the hidden Markov model to the transformer architecture, which outperforms the transformer baseline. Interestingly, we find that the zero-order model already provides promising performance, giving it an edge compared to a model with first-order dependency, which performs similarly but is significantly slower in training and decoding.

1 Introduction

Recently, significant improvements have been made to neural machine translations (NMT). Regardless of whether a recurrent neural network with long short-term memory (Hochreiter and Schmidhuber, 1997) (LSTM-RNN) (Bahdanau et al., 2015) or a convolutional neural network (CNN) (Gehring et al., 2017) or a self-attentive transformer network (Vaswani et al., 2017) is used, the attention mechanism is always one of the key components that all state-of-the-art NMT systems contain.

Several attempts have been made to explore alternative architectures that do not use an attention mechanism (Wang et al., 2017, 2018; Bahar et al., 2018; Press and Smith, 2018). However, either the performance of those systems is significantly worse than that of the LSTM-RNN-based approaches, or the time and memory complexity is much higher. Since the transformer architecture has upgraded the state-of-the-art to an even higher standard, fewer studies are being carried out in this direction.

Despite the promising translation performance of the transformer architecture, recent studies have found that the quality of the word alignments produced by the multi-head cross-attention weights is

quite poor, and various techniques are proposed to address this problem (Alkhouli et al., 2018; Garg et al., 2019; Zenkel et al., 2020). While these works focus on extracting promising alignment information from the transformer architecture, we aim to improve the translation performance of the baseline model by introducing alignment components while keeping the system monolithic. To this end, the possibilities are studied to apply the transformer architecture to the direct hidden Markov model (HMM), which is not as straightforward as in the case of LSTM-RNN due to the cross-attention through all decoder layers. Experimental results show that the zero-order direct HMM already outperforms the baseline transformer model in terms of TER scores (Snover et al., 2006), while the first-order dependency with higher computational complexity offers no further improvements.

2 Related Work

The attention component is introduced by Bahdanau et al. (2015) in NMT to simulate the alignment between the source and target sentence, which leads to significant improvements compared to the pure sequence-to-sequence model (Sutskever et al., 2014). Wang et al. (2018) present a LSTM-RNN-based HMM that does not employ an attention mechanism. This work aims to build a similar model with the transformer architecture. While they perform comparable to the LSTM-RNN-based attention baseline with a much slower model, our model outperforms the transformer baseline in terms of TER scores.

The derivation of neural models for translation on the basis of the HMM framework is also studied in Yu et al. (2017) and Alkhouli et al. (2018). In Yu et al. (2017), alignment-based neural models are used to model alignment and translation from the target to the source side (inverse direction), and

a language model is included in addition. And Alkhouli et al. (2018) rely on alignments generated by statistical systems that serve as supervision for the training of the neural systems. By contrast, the model proposed in this work does not require any additional language model or alignment information and thus keeps the entire system monolithic.

Several works have been carried out to change attention models to capture more complex dependencies. Cohn et al. (2016) introduce structural biases from word-based alignment concepts such as fertility and Markov conditioning. Arthur et al. (2016) incorporate lexical probabilities to influence attention. These changes are based on the LSTM-RNN-based attention model. Garg et al. (2019) and Zenkel et al. (2020) try to generate translation and high-quality alignment jointly using an end-to-end neural training pipeline. By contrast, our work focuses more on improving the translation quality using the alignment information generated by the self-contained model.

3 Direct HMM

The goal of machine translation is to find the target language sentence $e_1^I = e_1, e_2, \dots, e_I$ that is the translation of a particular source language sentence $f_1^J = f_1, f_2, \dots, f_J$ with the maximum likelihood ($\arg \max_{I, e_1^I} \{\Pr(e_1^I | f_1^J)\}$). In the direct HMM, an alignment from target to source ($i \rightarrow j = b_i$) is introduced into the translation probability:

$$\Pr(e_1^I | f_1^J) = \sum_{b_1^I} \Pr(e_1^I, b_1^I | f_1^J) \quad (1)$$

$$= \sum_{b_1^I} \prod_{i=1}^I \Pr(b_i, e_i | b_0^{i-1}, e_0^{i-1}, f_1^J) \quad (2)$$

$$= \sum_{b_1^I} \prod_{i=1}^I \underbrace{\Pr(e_i | b_0^{i-1}, e_0^{i-1}, f_1^J)}_{\text{lexicon probability}} \cdot \underbrace{\Pr(b_i | b_0^{i-1}, e_0^{i-1}, f_1^J)}_{\text{alignment probability}} \quad (3)$$

The term ‘‘direct’’ refers to the modeling of $p(e|f)$ instead of $p(f|e)$ as in the conventional HMM (Vogel et al., 1996). In Wang et al. (2018), two LSTM-RNN based neural networks are used to model the lexicon and the alignment probability separately. In this work they are modeled with a single transformer-based network.

4 Direct HMM in Transformer

This section describes in detail how we modify the transformer model so that both the alignment and

the lexicon probability can be generated. While the lexicon model in the direct HMM has a zero-order dependency on the current alignment position b_i :

$$\Pr(e_i | b_0^i, e_0^{i-1}, f_1^J) := p(e_i | b_i, e_0^{i-1}, f_1^J) \quad (4)$$

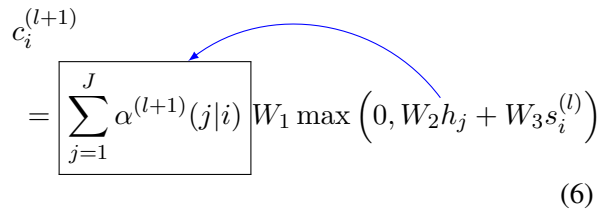
we implement zero- and first-order dependencies for the alignment model.

4.1 Zero-order Architecture

In the zero-order architecture, the alignment model is defined as follows:

$$\Pr(b_i | b_0^{i-1}, e_0^{i-1}, f_1^J) := p(b_i | e_0^{i-1}, f_1^J) \quad (5)$$

To obtain the alignment probability we change the order of the weighted sum and the activation function at each decoder layer in the transformer:

$$c_i^{(l+1)} = \sum_{j=1}^J \alpha^{(l+1)}(j|i) W_1 \max\left(0, W_2 h_j + W_3 s_i^{(l)}\right) \quad (6)$$


l : index of the decoder layer $\in \{1, 2, \dots, L\}$

c_i : context vector, input to the next layer

h_j : source hidden state (key and value)

s_i : target hidden state (query)

W_n : weight matrices

$\alpha(j|i)$: $\text{softmax}(A[s_i, h_j])$ cross-attention weights

The arrow indicates that the weighted sum with the cross-attention is moved outside of the ReLU activation function. Before the ReLU function is employed, the target hidden state s_{i-1} is projected and added to the projected source hidden state h_j in order to include information from the target side to the context vector, which can also be considered as a substitution for the residual layer in the standard transformer architecture. As the outputs of the last decoder layer (and the entire network) we have a lexicon probability:

$$p(e_i | j, e_0^{i-1}, f_1^J) = \text{softmax}\left(W_4 \cdot \max\left(0, W_5 \cdot h_j + W_6 \cdot s_i^{(L)}\right)\right) \quad (7)$$

and an alignment probability:

$$p(j | e_0^{i-1}, f_1^J) = \alpha^{(L)}(j|i) \quad (8)$$

The output probability for the current word is:

$$p(e_i | e_0^{i-1}, f_1^J) = \sum_{j=1}^J p(j | e_0^{i-1}, f_1^J) \cdot p(e_i | j, e_0^{i-1}, f_1^J) \quad (9)$$

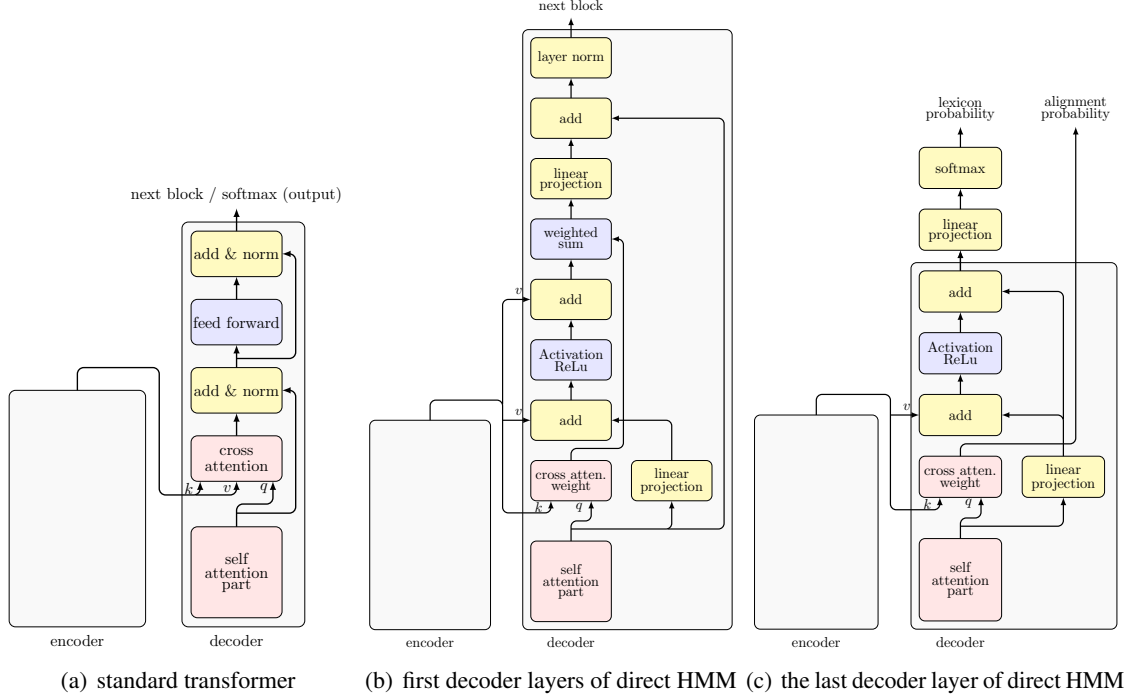


Figure 1: Visualized comparison between the direct HMM and the standard transformer architecture.

And the sentence probability is then:

$$p(e_1^I | f_1^J) = \prod_{i=1}^I p(e_i | e_0^{i-1}, f_1^J) \quad (10)$$

Due to the redefinition of the context vector, layer normalization, residual connection and linear projection are also modified accordingly. Detailed changes to the architecture are shown in Figure 1. Note that all modifications are made to decoder layers while encoder layers remain unchanged.

4.2 First-order Architecture

In the first-order architecture, the alignment model is defined as follows:

$$\Pr(b_i | b_0^{i-1}, e_0^{i-1}, f_1^J) := p(b_i | b_{i-1}, e_0^{i-1}, f_1^J) \quad (11)$$

The lexicon probability remains the same as in the zero-order model (Equation 4). To consider the dependency on the previous source position ($j' = b_{i-1}$), we change the cross-attention weights: $\alpha^{(L)}(j|i, j')$

$$= \text{softmax} \left(A \left[s_i^{(L-1)}, W \cdot [h_j^{(L)}, h_{j'}^{(L)}] \right] \right) \quad (12)$$

where $[h_j^{(L)}, h_{j'}^{(L)}]$ denotes the concatenation of the source hidden states at positions j and j' .

Changing the architecture from the zero-order model to the first-order model is straightforward,

but the main challenge is in the training process. Due to the first-order dependency, the complexity of the brute-force search (forward path) becomes exponential (confirm Equation 3). To address this problem, we apply a dynamic programming algorithm to find the probability of the entire sentence:

$$Q(i, j) = \sum_{j'} p(e_i, j | j', f_1^J, e_0^{i-1}) \cdot Q(i-1, j') \quad (13)$$

$$p(e_1^I | f_1^J) = Q(I) = \sum_j Q(I, j) \quad (14)$$

where Q denotes the recursive function. For given sentence pairs (F_r, E_r) , the training criterion is then the maximization of the log-likelihood function $\arg \max_{\theta} \sum_r \log p(E_r | F_r, \theta)$.

In previous work on the neural HMM, the forward-backward algorithm is implemented to calculate the posterior probability as the golden truth to guide the training of the lexicon and the alignment models (referred to as “manual differentiation”). But actually it is not necessary. As long as the forward path is implemented according to a recursive function of dynamic programming, as shown in Equation 13, the frameworks can handle the backward path automatically (referred to as “automatic differentiation”). Intuitively, the recursive equation is nothing more than a sum of products that should be easy to work with the au-

automatic differentiation toolkit. Theoretically, the mathematical proof for this is presented in [Eisner \(2016\)](#). And practically, our experimental results of the automatic differentiation and the manual differentiation are the same as long as label smoothing ([Szegedy et al., 2016](#)) is not applied.

Without an explicitly implemented forward-backward algorithm, applying label smoothing is not straightforward as it should be applied to the words while the automatic differentiation is performed after the forward path has been done for the entire sentence. To solve this problem, we apply label smoothing to the lexicon probability $p(e_i|j, e_0^{i-1}, f_1^J)$ at each step of the forward path. Although in this case the type of label smoothing is different for the automatic and manual differentiation, experimental results are quite similar ($< 0.1\%$ differences). The automatic differentiation has an advantage in terms of memory and time complexity and is therefore used for all subsequent experiments.

5 Experiments

5.1 Translation Performance

In order to test the performance of the direct HMM, we carry out experiments on the WMT 2019¹ German→English (de-en), WMT 2019 Chinese→English (zh-en) and WMT 2018² English→Turkish (en-tr) tasks. These three tasks represent different amounts of training data, from hundreds of thousands to tens of millions. Detailed data statistics are shown in Appendix A.

The proposed approaches are completely implemented in fairseq ([Ott et al., 2019](#)). The standard transformer base model ([Vaswani et al., 2017](#)) implemented in the fairseq framework is used as our baseline and we follow the standard setup for hyperparameters. Translation performance is measured by case-insensitive BLEU ([Papineni et al., 2002](#)) and TER ([Snover et al., 2006](#)) scores with SACRE-BLEU toolkit ([Post, 2018](#)). The results are shown in Table 1.

The results show that the direct HMMs achieve comparable performance to the transformer baselines in terms of BLEU scores and outperform the baseline systems in terms of TER scores. The TER metric is known to favor shorter hypotheses, but from the length ratio results we can conclude that the improvements are not due to it. In addition, it

¹<http://www.statmt.org/wmt19/>

²<http://www.statmt.org/wmt18/>

BLEU [%]	de-en	zh-en	en-tr
transformer base	38.7	31.5	17.4
zero-order HMM	38.5	31.5	17.6
first-order HMM	38.7	31.3	17.7
TER [%]	de-en	zh-en	en-tr
transformer base	48.2	56.6	71.9
zero-order HMM	47.7	55.7	71.4
first-order HMM	47.9	55.4	71.2
length ratio [%]	de-en	zh-en	en-tr
transformer base	97.3	94.1	99.7
zero-order HMM	97.7	94.0	99.7
first-order HMM	98.0	93.9	99.5

Table 1: Experimental results on the WMT news translation tasks.

can be seen that the first-order dependency could not provide further improvements over the zero-order model. To find the possible reasons for this, we try to extract alignment heat maps with regard to the dependencies between the current position j and the predecessor position j' .

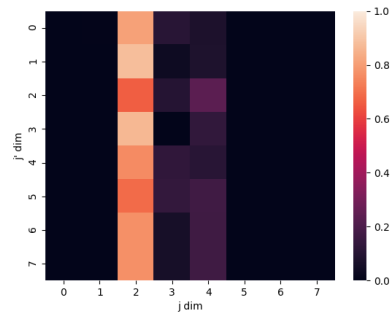


Figure 2: Alignment heat map for $p(j|j', e_0^{i-1}, f_1^J)$ with fixed target position i . The heat map is extracted when the training is almost converging.

As shown in Figure 2, the target position j with the maximum probability is often the same for different predecessor positions j' , which indicates that the training of the model tends to “forget” the explicit first-order dependency. We checked a lot of heat maps and this happens quite often, in fact, for short sentences it almost always happens. This essentially explains why the first-order model fails to make improvements. To benefit from the first-order dependency, constraints or other techniques might be used during training.

Here the results of the RNN-based direct HMM are not included as one of the baselines, as the performance of the RNN-based approaches is significantly surpassed by the transformer-based ap-

proaches. We believe this work will outperform the system proposed in (Wang et al., 2018), but that is mainly due to the transformer architecture rather than refinements we made.

Compared to the baseline transformer model, the direct HMM only has about 2% more free parameters. While the first-order model has a clear disadvantage in terms of training and decoding speed compared to the baseline system due to the inevitable loop over the target position i , the decoding speed of the zero-order model is only slightly slower than that of the transformer baseline. Details of time usage are given in Appendix B.

5.2 Alignment Quality

In addition to improvements in the TER scores, we believe that the direct HMM also provides better alignment quality than the standard cross-attention. To verify this assumption, we compute the alignment error rate (AER) (Och and Ney, 2000) on the RWTH German-English Golden Alignments corpus (Vilar et al., 2006), which provides 505 manually word-aligned sentence pairs extracted from the Europarl corpus. We take the argmax of the alignment probability output of our model as an estimated alignment. In addition, as with the conventional HMM, the argmax of the posterior probability can also be used as an estimated alignment, which explicitly includes the lexicon information and should lead to a better quality. As baselines, we take the argmax of the average of the attention heads in the fifth and sixth decoder layers, since Garg et al. (2019) claim that the cross-attention weights in the fifth layer produce more accurate alignment information than the last layer. All models are trained in both directions to get bidirectional alignments. These bidirectional alignments are then merged using the grow diagonal heuristic (Koehn et al., 2005).

model	alignment from	AER
transformer	fifth layer	39.1
	sixth layer	55.7
direct HMM	alignment prob.	31.8
	posterior prob.	27.4

Table 2: Experimental results on the German-English alignment task in AER [%].

From the results shown in Table 2, we can observe that the alignment generated by the direct HMM has a significantly better quality than that

extracted directly from the transformer attention weights. The posterior probability that contains the lexicon information indeed provides better alignments, which can be seen as a further advantage of the direct HMM, since it cannot be calculated in the standard transformer architecture without an explicit alignment probability. In terms of AER performance, our model stands behind GIZA++ (Och and Ney, 2003) as well as the approaches proposed in Garg et al. (2019) and Zenkel et al. (2020). Note, however, that our zero-order model does not include the future target word information in estimating alignments, and we do not use additional loss for alignment training, since the original goal of this work is to improve translation quality by applying HMM factorization.

In addition to the AER results, Appendix C shows heat maps extracted for the alignment probability from direct HMM compared to those extracted for cross-attention weights from the standard transformer model.

6 Conclusion

This work exhibits the use of the transformer architecture in a direct HMM for machine translation, which significantly improves TER scores. In addition, we show that the proposed system tends to “refuse” to learn first-order dependency during training. The zero-order model achieves a good compromise between performance and decoding speed, which is much faster than previous work on the direct HMM. In order to benefit from the predecessor alignment information, further techniques should be carried out. Another future work would be to combine the attention mechanism with the alignment information to further improve performance.

Acknowledgments

This work has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No 694537, project “SEQCLAS”). The work reflects only the authors’ views and the European Research Council Executive Agency (ERCEA) is not responsible for any use that may be made of the information it contains.

References

- Tamer Alkhouli, Gabriel Bretschner, and Hermann Ney. 2018. [On the alignment problem in multi-head attention-based neural machine translation](#). In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 177–185, Brussels, Belgium. Association for Computational Linguistics.
- Philip Arthur, Graham Neubig, and Satoshi Nakamura. 2016. [Incorporating discrete translation lexicons into neural machine translation](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1557–1567, Austin, Texas. Association for Computational Linguistics.
- Parnia Bahar, Christopher Brix, and Hermann Ney. 2018. [Towards two-dimensional sequence to sequence model in neural machine translation](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3009–3015, Brussels, Belgium. Association for Computational Linguistics.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. [Neural machine translation by jointly learning to align and translate](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Trevor Cohn, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. 2016. [Incorporating structural alignment biases into an attentional neural translation model](#). In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 876–885, San Diego, California. Association for Computational Linguistics.
- Jason Eisner. 2016. [Inside-outside and forward-backward algorithms are just backprop \(tutorial paper\)](#). In *Proceedings of the Workshop on Structured Prediction for NLP*, pages 1–17, Austin, TX. Association for Computational Linguistics.
- Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. [Jointly learning to align and translate with transformer models](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4453–4462, Hong Kong, China. 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*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252. PMLR.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. [Long short-term memory](#). *Neural Computation*, 9(8):1735–1780.
- Philipp Koehn, Amittai Axelrod, Alexandra Birch, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. [Edinburgh system description for the 2005 IWSLT speech translation evaluation](#). In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, pages 68–75, Pittsburgh, PA, USA.
- Ruixuan Luo, Jingjing Xu, Yi Zhang, Xuancheng Ren, and Xu Sun. 2019. [Pkuseg: A toolkit for multi-domain chinese word segmentation](#). *CoRR*, abs/1906.11455.
- Franz Josef Och and Hermann Ney. 2000. [Improved statistical alignment models](#). In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong. Association for Computational Linguistics.
- Franz Josef Och and Hermann Ney. 2003. [A systematic comparison of various statistical alignment models](#). *Computational Linguistics*, 29(1):19–51.
- Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. [fairseq: A fast, extensible toolkit for sequence modeling](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. [Bleu: a method for automatic evaluation of machine translation](#). In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.
- Matt Post. 2018. [A call for clarity in reporting BLEU scores](#). In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.
- Ofir Press and Noah A. Smith. 2018. [You may not need attention](#). *CoRR*, abs/1810.13409.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. [Neural machine translation of rare words with subword units](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.
- Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. [A study of translation edit rate with targeted human annotation](#). In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231, Cambridge, MA, USA.

- Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. [Sequence to sequence learning with neural networks](#). In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.
- C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. 2016. [Rethinking the inception architecture for computer vision](#). In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, Los Alamitos, CA, USA. IEEE Computer Society.
- 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 *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.
- David Vilar, Maja Popović, and Hermann Ney. 2006. [AER: Do we need to “improve” our alignments?](#) In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, pages 205–212, Kyoto, Japan.
- Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. [HMM-based word alignment in statistical translation](#). In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.
- Weiyue Wang, Tamer Alkhouli, Derui Zhu, and Hermann Ney. 2017. [Hybrid neural network alignment and lexicon model in direct HMM for statistical machine translation](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 125–131, Vancouver, Canada. Association for Computational Linguistics.
- 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, Melbourne, Australia. Association for Computational Linguistics.
- Lei Yu, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Tomáš Kociský. 2017. [The neural noisy channel](#). In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.
- Thomas Zenkel, Joern Wuebker, and John DeNero. 2020. [End-to-end neural word alignment outperforms GIZA++](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1605–1617, Online. Association for Computational Linguistics.

A Data Statistics

WMT 2019 German→English	train		valid		test	
	source	target	source	target	source	target
# sentence pairs	5.9M		2169		2000	
# original vocabulary	2.1M	932k	12.2k	10.7k	10.8k	9.5k
# vocabulary after BPE	45.1k	33.2k	10.5k	8.2k	9.3k	7.3k
# running words	137M	144M	38.2k	40.8k	31.1k	34.4k
# running BPE sub-words	160M	157M	54.8k	53.1k	44.7k	43.4k
WMT 2019 Chinese→English	train		valid		test	
	source	target	source	target	source	target
# sentence pairs	26.0M		2002		2000	
# vocabulary	1.3M	651k	9.2k	8.7k	9.5k	8.5k
# vocabulary after BPE	47.0k	32.2k	9.2k	9.2k	9.3k	8.8k
# running words	555M	606M	53.7k	59.8k	62.7k	82.2k
# running BPE sub-words	588M	658M	58.7k	65.1k	69.2k	87.2k
WMT 2018 English→Turkish	train		valid		test	
	source	target	source	target	source	target
# sentence pairs	208k		3007		3000	
# vocabulary	70.6k	160k	8.7k	15.1k	9.4k	16.4k
# vocabulary after BPE	7280	7324	4944	5437	5093	5592
# running words	5.16M	4.61M	68.3k	55.0k	70.5k	56.8k
# running BPE sub-words	6.72M	7.45M	98.0k	101k	101k	107k

For the German→English task, joint byte pair encoding (BPE) (Sennrich et al., 2016) with 32k merge operations is used. The `newstest2015` dataset is used as the validation set and `newstest2019` as the test set.

The Chinese data are segmented using the `pkuseg` toolkit³ (Luo et al., 2019). The vocabulary size and number of running words are calculated after segmentation. Separate BPE with 32k merge operations is used for Chinese and English data. The `newsdev2017` dataset is used as the validation set and `newstest2019` as the test set.

For the English→Turkish task, separate BPE with 8k merge operations is used. The `newstest2017` dataset is used as the validation set and `newstest2018` as the test set.

³<https://github.com/lancopku/pkuseg-python>

B Training and Decoding Speed

Training and decoding are performed on one NVIDIA GeForce RTX 1080 Ti with 11 GB of GPU memory. Table 3 shows the training and decoding speed on the WMT 2019 German→English dataset. Compared to the baseline system, the disadvantages of the zero-order HMM on training speed are mainly due to the limited GPU memory. Since the largest tensor of the proposed model has a dimension of batch size \times length of the source sentence \times length of the target sentence \times vocabulary size (in the standard transformer the dimension of “length of the source sentence” is not required), the batch size must be reduced to fit in the GPU memory. Although gradient accumulation can be used to guarantee performance, the reduced batch size still linearly slows the training speed. The influence on the decoding speed is rather small. By introducing the first-order dependency, however, a `for` loop over every target position is inevitable, so that the training and decoding speeds are greatly slowed down. This is also reported by the previous work.

model	# parameters	training		decoding	
		tokens/sec	time	tokens/sec	time
transformer baseline	84.2M	10.2k	5d	108.2	6.9min
zero-order HMM	86.1M	2.2k	20d	84.0	8.9min
first-order HMM	88.0M	0.4k	54d	31.7	23.5min

Table 3: Comparison of training and decoding speed.

C Heat Maps of Attention Weights and Alignments

Figure 3 demonstrates the heat maps of some sentence pairs that are randomly selected from the German→English training data after the training has almost converged. Note that here the x and y axes indicate the source and target positions (j and i), which differs from Figure 2, where they indicate the current and previous source positions (j and j'). We can observe that the alignment paths are much more focused than the attention weights. Since our main goal is to propose an alternative technique to improve translation performance rather than alignment quality, alignment error rates are not calculated in this work.

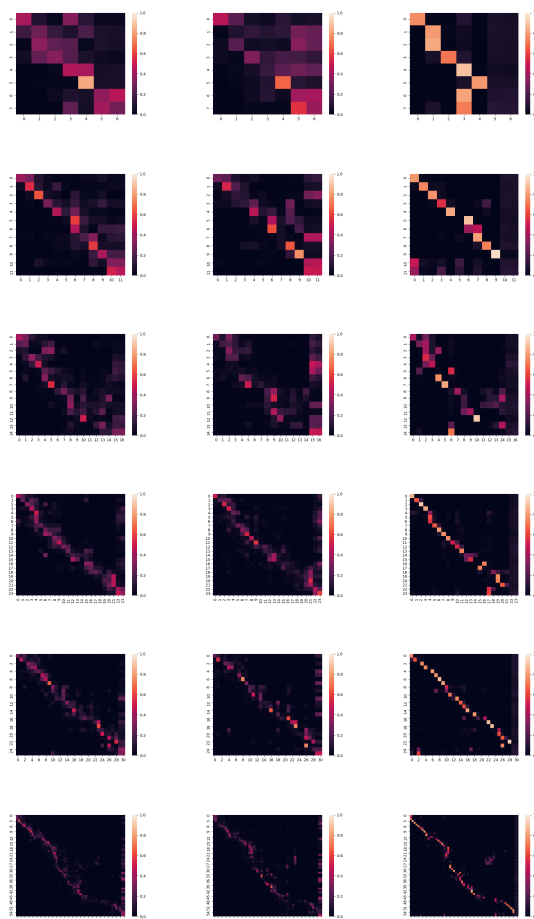


Figure 3: Heat maps of attention weights and alignments. The source sentence goes from left to right and the target sentence goes from top to bottom. The first column shows the attention weight heat maps (average of the multi-head cross-attention) for the 4th decoder layer. The second column shows the attention weight heat maps (average of the multi-head cross-attention) for the 6th (last) decoder layer. The third column shows the alignment heat maps taken from the proposed direct HMM.