Syntax-Based Attention Masking for Neural Machine Translation

Colin McDonald and David Chiang
University of Notre Dame
Dept. of Computer Science and Engineering
{cmcdona8,dchiang}@nd.edu

Abstract

We present a simple method for extending transformers to source-side trees. We define a number of masks that limit self-attention based on relationships among tree nodes, and we allow each attention head to learn which mask or masks to use. On translation from English to various low-resource languages, and translation in both directions between English and German, our method always improves over simple linearization of the source-side parse tree and almost always improves over a sequence-to-sequence baseline, by up to +2.1% BLEU.

1 Introduction

The transformer model for machine translation (Vaswani et al., 2017) was originally defined as a mapping from sequences to sequences. More recent work has explored extensions of transformers to other structures: a tree transformer would be able to make use of syntactic information, and a graph transformer would be able to make use of semantic graphs or knowledge graphs.

There have been a number of proposals for transformers on trees, including phrase-structure trees and dependency trees for natural languages, and abstract syntax trees for programming languages. One common strategy is to *linearize* a tree into a sequence (Ahmad et al., 2020; Currey and Heafield, 2019). Another strategy is to recognize that transformers are fundamentally defined not on sequences but on bags; all information about sequential order is contained in the positional encodings, so all that is needed to construct a tree transformer is to define new positional encodings on trees (Shiv and Quirk, 2019; Omote et al., 2019).

In this paper, we present a third approach, which is to enhance the encoder's self-attention mechanism with *attention masks* (Shen et al., 2018), which restrict the possible positions an attention head can attend to. We extend this idea in two new ways. First, our attention masks are based on relationships among tree positions (for example, "is an ancestor of" or "is a descendant of") rather than sequence positions ("is left of" or "is right of"). Second, instead of pre-assigning different masks to each attention head, we allow each attention head to learn separately which mask or masks to use.

We experiment on machine translation of several low-resource language pairs (Section 3). Compared to linearization without masks, our method always improves accuracy, by up to +1.7 BLEU (all BLEU reported as percen t). Compared with a sequence-to-sequence baseline, our method improves accuracy by up to +2.1 BLEU. On tasks where linearization hurts, our method is usually, but not always, able to turn the loss into a gain.

2 Methods

Like several previous approaches, we use linearized syntax trees. But whereas the usual linearization traverses a node both before and after its descendants, we use a preorder traversal of the tree. In other words, our linearization does not have closing brackets. Our linearization does not have enough information to reconstruct the original tree; this information is contained in the attention masks, which we describe next.

Shen et al. (2018) introduce the idea of using *masks* in a string transformer to allow attention heads to attend only to the left or only to the right. We apply this idea to tree transformers, with two modifications. First, instead of masking out the left or right context, we use masks based on the structure of the tree. Second, instead of allocating a fixed number of heads to each mask, we let the model learn which mask(s) to use for each attention head.

Given a query $Q \in \mathbb{R}^{n \times d_k}$, key $K \in \mathbb{R}^{n \times d_k}$, and value $V \in \mathbb{R}^{n \times d_v}$ (where n is the number of input tokens and $d_k = d_v$ is d_{model} divided by the number of attention heads), scaled dot-product attention is normally computed as

$$\alpha = \text{softmax}\frac{QK^T}{\sqrt{d_k}}$$
$$\text{Att}(Q, K, V) = \alpha V$$

where $\alpha \in \mathbb{R}^{n\times n}$ is the matrix of attention weights, and the softmax is performed per row. We modify the definition of α to

$$\alpha = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} - \exp \sum_m s^m M^m\right)$$

where, for each m, the matrix $M^m \in \{0, 1\}^{n\times n}$ is a fixed mask and s^m is its corresponding strength, which is learnable. If $[M^m]_{ij} = 1$ and s^m is large, then the attention at position i is prevented from attending to position j. If $[M^m]_{ij} = 0$ or s^m is very negative, then position i is free to attend to position j. With multiple attention heads, each head has its own strength parameters.

The strength parameters are initialized to zero and learned by backpropagation with the rest of the model. In this way, each attention head can learn separately which mask or masks to use.

It remains to define the masks M^m. A mask can be defined for any imaginable string or tree relationship. Because the model can always choose not to use a mask, we can add as many masks as we want. We use the following set:

self position i is equal to position j

parent position i is the parent of position j

child position i is a child of position j

left-sib position i is a left sibling of position j

right-sib position i is a right sibling of position j

anc position i is an ancestor (but not a parent) of position j

desc position i is a descendent (but not a child) of position j

left-other position i has none of the above relationships with position j, but is left of position j

right-other position i has none of the above relationships with position j, but is right of position j

Task	Lines			Avg. source	
	train	dev	test	words	nodes
En-Vi	131k	1,553	1,268	22.9	36.4
En-De	100k*	3,000	3,003	28.5	45.5
De-En	100k*	3,000	3,003	29.6	34.6
En-Tu	59k	1,114	544	28.7	39.1
En-Ha	45k	914	497	26.5	39.3
En-Ur	11k	1,271	652	22.5	30.7

Table 1: Dataset statistics. Nodes: average number of *interior* nodes. *The original German–English dataset had 4.5M lines, but we only trained on subsets of up to 100k lines.

Although none of the above masks overlap, there would be no problem with defining masks that do.

Please see Figure 1 for an example. In (a) is an English tree; (b) shows the same tree after applying byte pair encoding (BPE) subword segmentation (see Section 3 below); and (c) shows the relationships of all the nodes with the second NP (the one dominating *my father*).

3 Experiments

3.1 Data

We tested on the following datasets:

en-vi English to Vietnamese, from the IWSLT 2015 shared task.[1] To test for dependence of our method on training data size, we also used random subsets of 20k and 50k.

de-en, en-de German↔English, from the WMT 2016 news translation task.[2] For training, we used random subsets of 20k, 50k, and 100k. We used news-test2013 for validation and news-test2014 for testing.

en-tu, en-ha, en-ur English to Turkish, Hausa, and Urdu, from the DARPA LORELEI program.

Some statistics of the datasets are shown in Table 1. This table lists the average number of source words and source *interior* nodes, from which the average number of tokens in the **linearized** and **mask** systems can be derived.

We tokenized using the Moses tokenizer, then divided words into subwords using BPE (Sennrich

[1] https://nlp.stanford.edu/projects/nmt/
[2] https://www.statmt.org/wmt16/translation-task.html

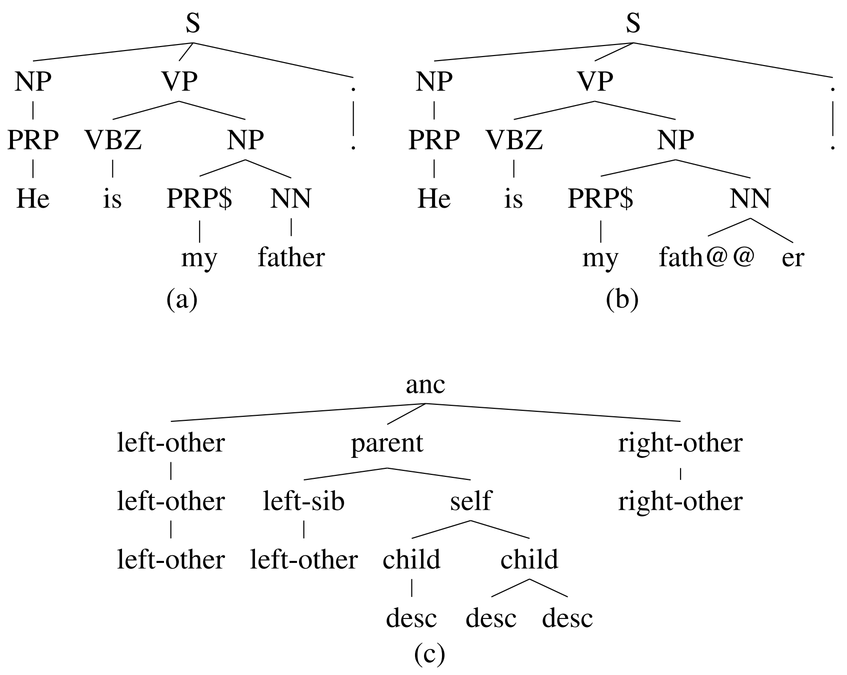

Figure 1: (a) Example tree. (b) With BPE. (c) Relationships of all nodes to the second NP (dominating *my father*).

et al., 2016). For en-vi, en-tu, en-ha, and en-ur, we used 8k joint BPE operations, and for en-de and de-en, we used 32k operations.

To parse English or German sentences, we used the Berkeley Neural Parser (Kitaev and Klein, 2018; Kitaev et al., 2019) with the included `benepar_en2` model for English and `benepar_de` for German. The parser reads in untokenized strings and writes out tokenized trees; we used the parser's tokenization, but applied BPE to the leaves, as shown in Figure 1b.

3.2 Evaluation

We compare against two baselines: **Sequence** is a standard sequence-to-sequence model, run on words only. **Linearized** is a standard sequence-to-sequence model, run on linearized trees. A leaf node w is linearized as w. An interior node X is linearized as $\boxed{(X}$ followed by the linearization of its children followed by $\boxed{)}$. Against these baselines, we compare our model, **Mask**, which uses a pre-order traversal of the tree together with the masks described above in Section 2.

All systems are implemented on top of Witwicky,[3] an open-source implementation of the transformer. We use all default settings; in particular, layer normalization is performed after residual connections (Nguyen and Salazar, 2019).

[3]https://github.com/tnq177/witwicky

We score detokenized system outputs using case-sensitive BLEU against raw references (except on en-vi, where we use tokenized outputs and references), using bootstrap resampling (Koehn, 2004; Zhang et al., 2004) for significance testing.

3.3 Results

The results are shown in Table 2. Relative to the **linearized** baseline, our method (**mask**) always improves, by up to +1.7 BLEU for English–Turkish. The difference is statistically significant ($p < 0.05$) except for English–Urdu.

Relative to the **sequence** baseline, the story is more complex. Whenever **linearized** helps over **sequence**, our method helps more, up to a total of +2.1 BLEU for German↔English (50k). But when **linearized** hurts, our method sometimes helps overall (all tasks with 20k lines of training) and sometimes doesn't (e.g., English–Urdu, with only 11k lines of training). A simple possible explanation is that additional tokens make training more difficult on the very smallest datasets, and the effect is stronger for **linearized**, which has twice as many extra tokens.

4 Analysis

4.1 Which masks get used

Figure 3 shows a heatmap of mask strengths for the English–German task (100k lines), and Figure 2

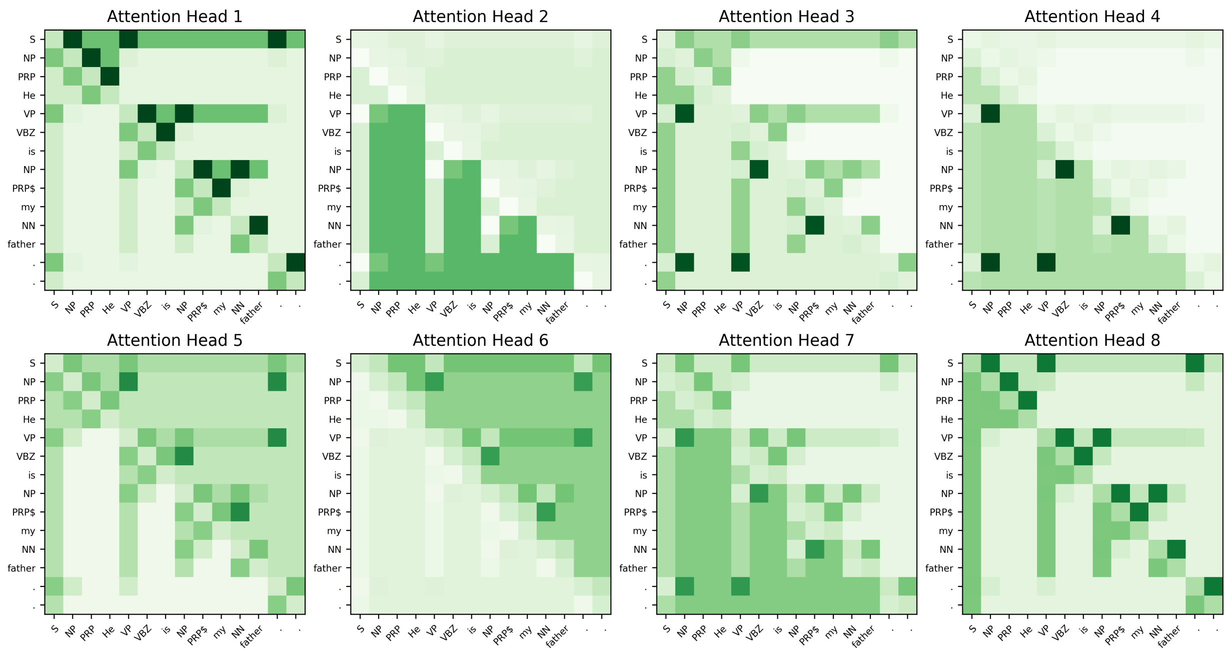

Figure 2: English–German attention masks. The cell in row i, column j shows the strength of the attention mask for node i attending to node j. Light is the strongest (least attention) and dark is the weakest (most attention); see Figure 3 for scale.

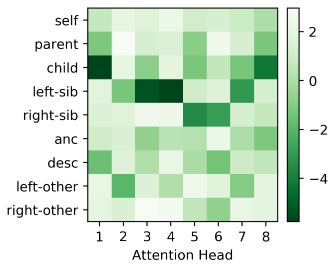

Figure 3: English–German mask strengths. Light is the strongest (least attention) and dark is the weakest (most attention).

Dataset	Min	Max	Range	ΔBLEU
en-vi (full)	−7.52	3.32	10.84	−0.24
en-de (100k)	−5.77	2.96	8.73	1.49
de-en (100k)	−6.29	2.59	8.88	1.41
en-tu	−6.88	3.03	9.91	1.89
en-ha	−4.49	2.41	6.90	1.16
en-ur	−0.32	0.23	0.55	−1.32

Figure 4: Minimum, maximum, and range of mask strengths. ΔBLEU = Change in test BLEU relative to Sequence baseline.

displays the resulting sum of the masks for each attention head for the parse tree of the sentence, "He is my father." There's a strong left-right asymmetry, with heads 2–4 and 7 attending to the left and heads 1 and 5–6 attending to the right. There's also a strong preference to attend to nodes that are nearby in the tree, with strongest weights on the child, left-sib, and right-sib relations.

4.2 Usefulness of masks

Figure 4 shows the minimum, maximum, and range of the mask strengths learned for various tasks. Generally, a mask's range correlates with its usefulness to the model. In particular, on Urdu–English, where we saw the syntax-based models perform the worst, we also see the masks being used the least and distinguished the least. English-Vietnamese is clearly an exception to this, however, with the highest maximum and widest range, but a small (insignificant) loss in BLEU.

5 Conclusion

In this paper, we've shown that syntax can be both helpful and easy to incorporate into low-resource neural machine translation. We introduced learnable attention masks for the transformer that allow each attention head to focus more narrowly on certain node relationships in the syntax tree, improving translation across a variety of low-resource datasets by up to +2.1 BLEU.

English–Vietnamese (en-vi)

	lines		
	20k	50k	131k
Sequence	19.44	**27.23**	**31.99**
Linearized	19.20	25.92	31.06
Mask	**21.41**	26.34	**31.75**

English–German (en-de)

	lines		
	20k	50k	100k
Sequence	**2.77**	10.83	15.45
Linearized	2.18	11.78	16.42
Mask	**2.88**	**12.95**	**16.94**

German–English (de-en)

	lines		
	20k	50k	100k
Sequence	4.19	13.37	18.64
Linearized	3.57	13.81	19.54
Mask	**4.73**	**15.45**	**20.05**

English to Other Languages

	target language / lines		
	tu	ha	ur
	59k	45k	11k
Sequence	22.30	23.46	**12.98**
Linearized	22.47	23.16	11.52
Mask	**24.19**	**24.62**	11.66

Table 2: Experiment results. In each column, the best score and any scores not significantly different from the best ($p \geq 0.05$) are printed in boldface.

6 Acknowledgements

We would like to thank Toan Nguyen for providing a base implementation of the transformer and answering questions about it, and the anonymous reviewers for their helpful comments.

References

Wasi Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. 2020. A transformer-based approach for source code summarization. In *Proc. ACL*, pages 4998–5007.

Anna Currey and Kenneth Heafield. 2019. Incorporating source syntax into transformer-based neural machine translation. In *Proc. Conference on Machine Translation*, pages 24–33.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy. Association for Computational Linguistics.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Toan Q. Nguyen and Julian Salazar. 2019. Transformers without tears: Improving the normalization of self-attention. In *Proc. Workshop on Spoken Language Translation*.

Yutaro Omote, Akihiro Tamura, and Takashi Ninomiya. 2019. Dependency-based relative positional encoding for transformer NMT. In *Proc. RANLP*, pages 854–861.

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.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. DiSAN: Directional self-attention network for RNN/CNN-free language understanding. In *Proc. AAAI*.

Vighnesh Shiv and Chris Quirk. 2019. Novel positional encodings to enable tree-based transformers. In *Advances in Neural Information Processing Systems*, volume 32, pages 12081–12091. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

Ying Zhang, Stephan Vogel, and Alex Waibel. 2004. Interpreting BLEU/NIST scores: How much improvement do we need to have a better system? In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal. European Language Resources Association (ELRA).