Promoting the Knowledge of Source Syntax in Transformer NMT

Thuong-Hai Pham and **Dominik Macháček** and **Ondřej Bojar**
Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
{pham,machacek,bojar}@ufal.mff.cuni.cz

Abstract

The utility of linguistic annotation in neural machine translation has been already established. The experiments were however limited to recurrent sequence-to-sequence architectures and relatively small data settings. We focus on the state-of-the-art Transformer model and use comparably larger corpora. Specifically, we try to promote the knowledge of source-side syntax using multi-task learning either through simple data manipulation techniques or through a dedicated model component. The novel idea is to interpret Transformer self-attention as a dependency parse. While the data manipulation techniques are ineffective in large data settings, the treatment of self-attention as dependencies helps in translation and reveals that Transformer model can very easily grasp this structure.

1 Introduction

Neural machine translation (NMT) has dominated the field of MT and many works are emerging that document that the quality of NMT can be, under some circumstances, further improved by incorporating linguistic information from the source and/or target side.

Experiments so far were however limited to the recurrent sequence-to-sequence architectures (Cho et al., 2014; Bahdanau et al., 2015).

The latest WMT evaluation Bojar et al. (2018) and Popel (2018) show that the novel Transformer architecture (Vaswani et al., 2017) has set the new benchmark and it is thus interesting to see if providing this architecture with linguistic information is equally helpful or if Transformer already models the phenomena unsupervised.

We experiment with German-to-Czech and Czech-to-English translation and focus on source-side dependency annotation using multi-task techniques. We try two ways of forcing the model to consider source syntax: (1) by linearizing the syntactic tree and mixing the translation and parsing training examples, and (2) by adding a secondary objective to interpret one of the attention heads as the syntactic tree.

In Section 2, we survey recent experiments with incorporating linguistic information into NMT, focusing particularly on works which use multi-task learning strategies and on works that consider the syntactic analysis of the sentence. A brief description of the data and common settings of our experiments is provided in Section 3. In Section 4, we explore the simple technique of multi-task by alternating training examples of the individual tasks. The main positive contribution of this work is presented in Section 5, where we interpret the self-attention matrix in the Transformer architecture as the dependency tree of the source sentence. Section 6 discusses the observations and we conclude in Section 7.

2 Related Work

The idea of multi-task training is to benefit from inherent and implicit similarities between two or more machine learning tasks. If the tasks are solved by a joint model with fewer or more parameters shared among the tasks, the model should exploit the commonalities and perform better in one or more tasks. This improvement can come from various sources, including the additional (often different) training data used in the additional tasks or some form of regularization or generalization that the other tasks promote.

In machine translation, multitasking has brought interesting results in multi-lingual MT systems and also in using additional linguistic annotation (Luong et al., 2015; Zoph et al., 2016; Ha et al., 2016; Johnson et al., 2016).

Eriguchi et al. (2017) combined translation and

dependency parsing by sharing the translation encoder hidden states with the buffer hidden states in a shift-reduce parsing model (Dyer et al., 2016). Aiming at the same goal, Aharoni and Goldberg (2017) proposed a very simple method. Instead of modifying the model structure, they represented the target sentence as a linearized lexicalized constituency tree. Subsequently, a sequence-to-sequence (seq2seq) model (Sutskever et al., 2014) was used to translate the source sentence to this linearized tree, i.e. indeed performing the two tasks: producing the string of the target sentence jointly with its syntactic analysis. Le et al. (2017) use the same trick for (target-side) dependency trees, proposing a tree-traversal algorithm to linearize the dependency tree. Unfortunately, their algorithm was limited to projective trees.

In parallel to our work, Kiperwasser and Ballesteros (2018) examined various scheduling strategies for a very simple approach to multi-tasking: representing all the tasks converted to a common format of source and target sequences of symbols from a joint vocabulary and training one sequence-to-sequence system on the mix of training examples from the different tasks. The scheduling strategy specified the proportion of the tasks in training batches in time. Kiperwasser and Ballesteros (2018) report improvements in BLEU score (Papineni et al., 2002) in small-data setting for German-to-English translation for all multi-task setups (translation combined with POS tagging and/or source-side[1] parsing). In the "standard" data size and the opposite translation direction, results are mixed and only one of the scheduling strategies and only the POS secondary task help to improve MT over the baseline.

The papers mentioned so far targeted primarily the quality of MT (as measured by BLEU), not the secondary tasks. Kiperwasser and Ballesteros (2018) note that their system performs reasonably well in both tagging and parsing. Shi et al. (2016) present an in-depth analysis of the syntactic knowledge learned by the recurrent sequence-to-sequence NMT. Tran et al. (2018) are the first to use Transformer and observe that the recurrence is indeed important to model hierarchical structures.

[1] Kiperwasser and Ballesteros (2018) do not explicitly state whether they use the source or the target language treebank as the training data for the parsing task. While both is actually possible, and while even the combination of both could be tried, we assume they used the source-side treebank only.

Dataset	de2cs	cs2en
Train sent. pairs	8.8M	#00-#08: 5.2M
Train tokens (src/tgt)	89M/78M	61M/69M
Test sent. pairs	news 2013: 3k	#09: 10k
Dev sent. pairs	news 2011: 3k	#09: 1k

Table 1: Data used in our experiments. Test and dev data for de2cs originate in WMT newstests, all data for cs2en in indicated blocks of CzEng.

Nadejde et al. (2017) benefit from CCG tags (Steedman, 2000) added to NMT on the source side in the form of word factors and on the target side by interleaving the CCG tags and target words. The additional information proves useful when the CCG tags and words are processed in sync. Tamchyna et al. (2017) report similar success in interleaving words and morphological tags.

3 Data and Common Settings

Experiments in this paper are based on two language pairs: German-to-Czech (de2cs) translation trained on Europarl (Koehn, 2005) and OpenSubtitles2016 (Tiedemann, 2009), after some cleanup preprocessing, character normalization and tokenization. These are the only publicly available parallel data for this language pair. Czech-to-English (cs2en) translation was trained on a subset of CzEng 1.7 (Bojar et al., 2016).[2] The data sizes used for MT training are summarized in Table 1. For training of parsing tasks, we used the same datasets automatically annotated on source sides. For German source we used UDPipe (Straka and Straková, 2017), with the model trained on Universal Dependencies 2.0 (UD, Nivre et al., 2017). For Czech source we used the annotation provided in CzEng release, originally created by Treex (Popel and Žabokrtský, 2010). This annotation is based on Prague Dependency Treebank (PDT, Hajič et al., 2006). For parsing evaluation we used gold test set from UD and PDT, respectively.

We use several automatic evaluation metrics to assess translation quality: BLEU (Papineni et al., 2002), CharacTER (Wang et al., 2016), BEER (Stanojević and Sima'an, 2014), and chrF3 (Popovic, 2015). For experiments in Section 4, the BLEU score is cased, implemented within T2T,[3]

[2] http://ufal.mff.cuni.cz/czeng

[3] https://github.com/tensorflow/tensor2tensor/blob/master/tensor2tensor/utils/bleu_hook.py

in Section 5 with sacrebleu.[4] For dependency parsing task, we use unlabeled attachment score (UAS).

To assess the significance of the improvement over a given baseline, we use MT-ComparEval (Klejch et al., 2015), which implemented the paired bootstrap resampling test (confidence level 0.05 or 0.01; Koehn, 2004).

4 Simple Alternating Multi-Task

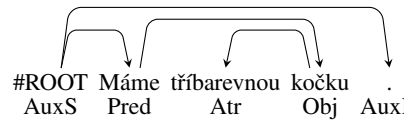
4.1 Approach

For simple multi-task learning, where the input and output of each task are represented as sequences, the basic architecture for MT can be used without any modifications. The identification of the task can be provided by a special token on the source side, which we add as the very last symbol of the sentence. The encoder and decoder of the NMT model are thus shared for all tasks which enables the encoder to learn source language better, but on the other hand it occupies a certain part of the model with task alternation and multiple language models for each task in the decoder.

In our experiments, we mix two tasks: MT and one additional linguistic (see Figure 1) or dummy referential task (see Figure 2). In "DepHeads" task, word forms of nodes' parents in the dependency tree are predicted. We can reconstruct unlabeled dependency tree in a post-processing step.[5] "DepLabels" task is tagging with dependency labels, and "DepHeads+DepLabels" is an interleaved combination of the two.

The training data in multitasking are selected by constant scheduler as in Kiperwasser and Ballesteros (2018), with parameter 0.5, which means the trainer alternates between the tasks, in average, after every training step. As Kiperwasser and Ballesteros (2018) reminds, this is different from Luong et al. (2015) and Zoph and Knight (2016) where the mixing happens at the level of batches and not individual examples.

The experiments here in Section 4 used Google's Tensor2Tensor version 1.2.9,



MT source (cs)		Máme tříbarevnou kočku . #Translate
MT target (en)		We have a three-colored cat .
DepHeads	src	Máme tříbarevnou kočku . #DepHeads
	tgt	#ROOT kočku Máme #ROOT
DepLabels	src	Máme tříbarevnou kočku . #DepLabels
	tgt	#Pred #Atr #Obj #AuxK
DepHeads +DepLabels	src	Máme tříbarevnou kočku . #DHeadsLab
	tgt	#ROOT #Pred kočku #Atr Máme #Obj #ROOT #AuxK

Figure 1: Sample dependency tree, inputs and expected outputs of linguistic secondary tasks.

Source words	We have a three-colored cat .
CountSrcWords	6
EnumSrcWords	W W W W W W
CopySrc	We have a three-colored cat .

Figure 2: Sample inputs and expected outputs of dummy secondary tasks.

`transformer_big_single_gpu` hyperparameter set (hidden size 1024, filter size 4096, 16 self-attention heads, 6 layers) with batch size 1500, 60k warmup steps and 100k shared vocabulary provided by T2T's default SubwordTextEncoder.

4.2 Training Cost of the Multi-Task

Adding training examples of the secondary task is bound to affect the training throughput and speed.[6] The hope is that this extra training cost is worth the gains obtained in the main task. We examine it empirically by comparing the training speed of the baseline run (no multi-task) and several versions of "dummy" multi-task setups as illustrated in Figure 2. In "CountSrcWords", the system is expected to count the source words and emit the result as one token holding the decimal number. "EnumSrcWords" is similar but the expected output is much easier for the architecture to grasp: the count should be expressed by an appropriate number of copies of the same special output token. In "CopySrc", the system should simply learn to copy the source, which should be very easy for an attentive architecture.

The task identification is clearly marked on input with a special token. To measure its impact on MT quality, we provide an experimental run "MT TaskID", where only one MT task with task iden-

[4] https://github.com/awslabs/sockeye/tree/master/contrib/sacrebleu

[5] If one word form appears multiple times in a sentence, we attach the edge to the nearest option. We propose this approach mostly for annotation schemes, in which content words (in contrast to function words) appear as inner nodes of dependency trees. Since content words are usually not repeated in sentences, there is a low chance they will be mismatched.

[6] We adopt the terminology of Popel and Bojar (2018).

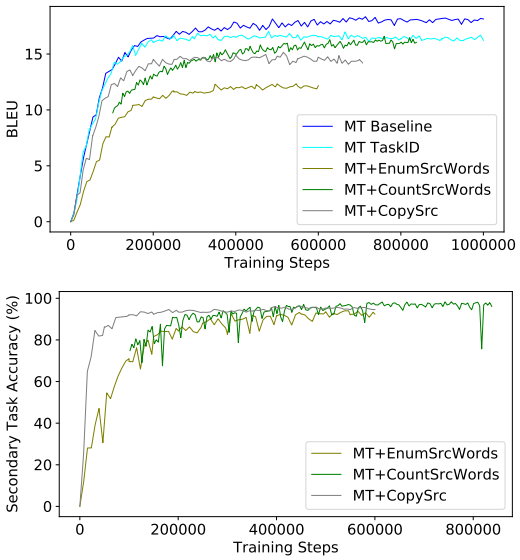


Figure 3: Learning curves of the de2cs baseline and dummy secondary tasks over training steps. MT BLEU on top, percentage of correct answers for secondary task on bottom.

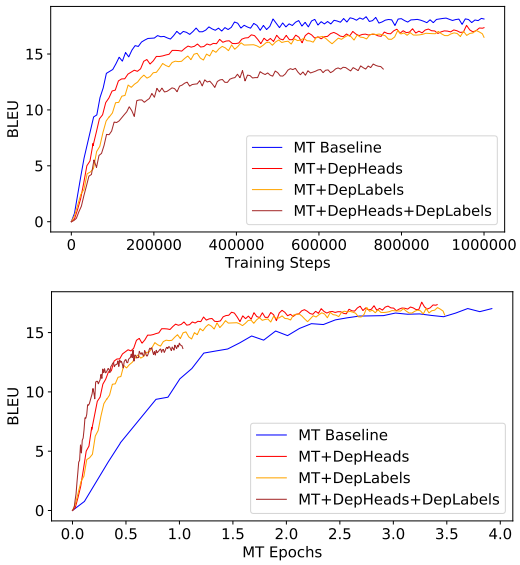


Figure 4: Learning MT BLEU curves of the de2cs baseline and linguistic secondary tasks over training steps (top) and over MT epochs (bottom).

tification token is provided.

Figure 3 summarizes the resulting learning curves on the development set. As we supposed, the secondary dummy task was easy to learn but it hurts MT performance. Enumerating and counting full words are very similar tasks in difficulty, the model learned them almost in same time, but enumerating worsens MT quality much more. It probably employs bigger part of decoder. A surprising result is that the task identification token on baseline MT data decreases overall MT performance in the long run.

4.3 Results of Simple Alternating Multi-Task

As Table 2 and Figure 4 (top) indicate, none of simple alternating multi-tasking method with linguistic secondary task outperformed baseline MT on any of our language pairs after the same amount of time. (In our conditions, training steps and training time are easily convertible; 600k training steps correspond to approximately 40 hours.)

However, if we measure the performance on MT training data throughput, we see the multi-tasking runs achieved the same level as the baseline with less training data. We conclude that the cost for sharing encoder and decoder between two tasks is higher than benefits from additional linguistic resources, but in particularly small data settings multi-tasking may be desirable.

Table 3 shows comparison between linguistic and dummy secondary tasks. "DepHeads" and "DepLabels" outperformed "CountSrcWords" and all other dummy tasks, so we conclude the model gains from semi-supervised syntactic input.

Table 4 shows the performance in parsing. As the referential parser, we use UDPipe for German, the one which supervised our model. Our system gains a similar UAS performance. It should be noted that we used the supervision by UDPipe in a non-standard way. Our system (and the referential parser) take raw word tokens on input, while UDPipe is designed to segment multi-word tokens, such as the German *zum*, into syntactic words, as *zu dem*, each of which are single nodes in tree. For Czech, we report the score of winner in CoNLL Shared Task 2007 (Nivre et al., 2007), the latest available evaluation on same data. We expect that the state of the art is higher nowadays. The limitation of our model may be the shared decoder and potentially inaccurate automatically annotated training data.

5 Promoting Dependency Interpretation of Self-Attention

In this section, we propose a different but similarly simple technique to promote explicit knowledge of source syntax in the model. Our inspiration comes from the neural model for dependency parsing by Dozat and Manning (2016). The model produces a matrix $S(u, v)$ expressing the probabil-

Model	de2cs dev				de2cs test				cs2en dev				cs2en test			
MT Baseline	**17.90‡**	**60.73**	**52.30**	**47.06**	**19.74‡**	**58.60**	**53.08**	**48.62**	**44.92**	**42.11**	**63.32**	**65.34**	**44.20**	**41.68**	**62.70**	**63.88**
MT+DepLabels	*16.52*	62.67	*50.86*	45.18	*17.87*	*59.65*	*51.67*	*47.01*	*41.98*	*43.28*	*61.80*	*63.63*	*41.94*	42.54	*61.63*	*61.96*
MT+DepHeads	16.36	*62.55*	50.76	*45.21*	17.51	62.15	51.29	46.52	40.72	43.75	61.85	62.78	41.10	*42.30*	61.41	61.68
MT+DepHeads+ DepLabels	13.62	70.25	48.52	43.06	15.45	67.14	49.69	44.79	39.57	45.63	60.50	61.30	40.25	43.63	60.97	61.05

Table 2: Automatic scores for MT with multi-task by simple alternation. All experiments are after 600k of training steps. Scores: BLEU, CharacTER, BEER, and chrF3. Best in bold, second-best slanted. Statistical significance marked as † ($p < 0.05$) and ‡ ($p < 0.01$) when compared to the second-best.

Model	dev	test
MT Baseline	**17.90**	**19.74**
MT TaskID	16.53	18.20
MT+DepLabels	16.52	17.87
MT+DepDheads	16.36	17.62
MT+CountSrcWords	15.70	17.51
MT+CopySrc	14.73	16.07
MT+DepHeads+DepLabels	13.62	15.45
MT+EnumSrcWords	12.16	14.04

Table 3: Comparison of BLEU scores at 600k training steps for linguistic and dummy secondary tasks with simple alternating approach.

Model	de2cs UAS	de2cs label acc	cs2en UAS	cs2en label acc
referential parser	**62.87**	*73.62*	**86.28**	83.38
MT+DepLabels	–	**75.40**	–	**85.01**
MT+DepHeads	*62.15*	–	*80.35*	–
MT+DepHeads+ DepLabels	54.98	68.44	80.01	*83.99*

Table 4: Test set scores for parsing source language (German and Czech, resp.) by simple alternation. "label acc" is the accuracy of tagging words with their dependency labels. Best in bold, second-best slanted.

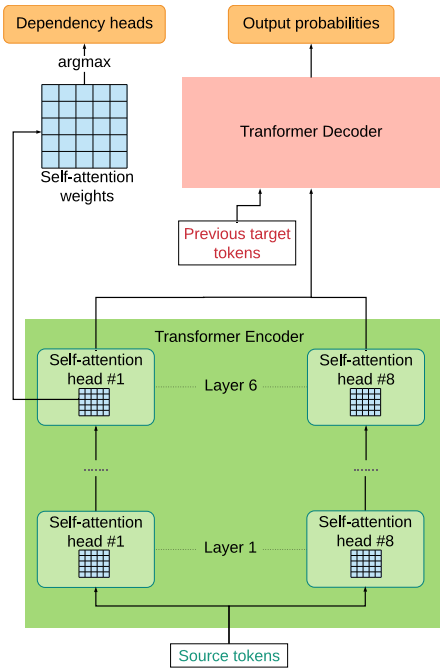


Figure 5: Joint dependency parsing and translation model ("DepParse").

ity that the word u is the head of v. The construction of this matrix is very similar to the matrix of self-attention weights α in the Transformer model. From this similarity, we speculate that the self-attentive architecture of Transformer NMT has the capacity to learn dependency parsing and we only need to promote a little the particular linguistic dependencies captured in a treebank.

5.1 Model Architecture

Figure 5 illustrates our joint model ("DepParse"). The translation part is kept unchanged. The only difference is that we reinterpret one of the self-attention heads in the Transformer encoder as if it was the dependency matrix $S(u, v)$. The training objective is combined and maximizes both the translation quality in terms of cross-entropy of the candidate translation and the unlabeled attachment score (UAS) of the proposed head against the (automatic) golden parse. The particular choice of the head which will serve as the dependency parser is arbitrary. Put differently, we constrain the Transformer model to use one of its heads to follow the given syntactic structure of the sentence.

It would be also possible to use e.g. the deep-syntactic parse of the sentence (the tectogrammatical layer as defined e.g. for the Prague Dependency Treebank, Hajič et al., 2006); we leave that for future work.

5.2 Experiment Setup

Experiments in this section were carried out with T2T version 1.5.6 at the *word level*, i.e. without using subword units. We decided for this simplification for an easier alignment between the translation and parsing tasks.

The Transformer hyper-parameter set `transformer_base` (Popel and Bojar, 2018) was used for all model variants with hidden size

	BLEU		UAS	
	Dev	**Test**	**Dev**	**Test**
TransformerBase	37.28	36.66	–	–
Parse from layer 0	36.95	36.60	81.39	82.85
Parse from layer 1	**38.51**	**38.01**	90.17	90.78
Parse from layer 2	38.50	37.87	91.31	91.18
Parse from layer 3	38.37	37.67	91.43	91.43
Parse from layer 4	37.86	37.60	**91.65**	**91.56**
Parse from layer 5	37.63	37.67	91.44	91.46

Table 5: DepParse's results in translation (BLEU) and parsing (UAS) on automatically annotated (cs2en). All test BLEU gains, except for layer 0, are statistically significant with $p < 0.01$ when compared to TransformerBase.

512, filter size 2048, 8 self-attention heads and 6 layers in each of the encoder and decoder. From now on, we refer the Transformer model with this hyper-parameter set as "TransformerBase", our baseline. We also experimented with the choice of the encoder layer, which we use for parsing.

In addition to the standard preprocessing for MT, we inserted a special "ROOT" word to the beginning of every sentence, so that the selected self-attention head would be able to represent a dependency tree correctly.

5.3 Layer Choice

Firstly, we experiment with selection of one of the six encoder layers from which we take the self-attention head that will serve as the dependency parse. Table 5 presents the results for both translation and parsing.

It is apparent that layer 0 (the first layer) is a too early stage for both tasks. The self-attention mechanism has only access to input word embeddings, and their relations are very likely to be useful semantically rather than syntactically. On the other hand, layers 1 and 2 perform well in parsing, and they are the best layers for translation quality. A possible explanation is that they already have sufficient information for a reasonably precise parse and do not consume the encoder's capacity for translation. Further layers perform generally better and better in parsing (because they are more informed) and maintain a solid performance in translation, but the translation quality is slowly decreasing. For the following, we select layer 1 to demand syntactic information from.

5.4 Performance in Translation

Table 6 compares the performance of the baseline Transformer, the simple alternating setup from Section 4 (DepHeads src) and the multi-task setup from this section. All these runs use T2T version 1.5.6 and use words, not subword units. This also explains the decrease in BLEU compared to Table 2. The DepParse model significantly outperforms the baseline (38.01 vs. 36.66 and 14.27 vs. 13.96).

Model	**de2cs**	**cs2en**
TransformerBase	*13.96*	*36.66*
Alternating multi-tasking (Section 4)	12.85	36.47
DepParse (Section 5)	**14.27**†	**38.01**‡

Table 6: BLEU scores on test set for translation task (T2T 1.5.6, word level). Statistical significance marked as † ($p < 0.05$) and ‡ ($p < 0.01$) when compared to TransformerBase.

Model	**de2cs**	**cs2en**
Referential parsers	62.87	86.28
DepParse	76.48	82.53

Table 7: UAS on gold annotated test sets for parsing task.

5.5 Performance in Parsing

In addition to the automatically annotated dev and test set, we also evaluated our model on the gold evaluation sets from UD 2.0 for German and from PDT 2.5 for Czech. The referential parsers were defined in Table 4. Table 7 shows that our model achieved good results in comparison to the baseline model on those datasets, even though ours was trained using synthetic data.

5.6 Diagonal Parse

For contrast, we conduct an experiment with a simpler sentence structure, which we call the "diagonal parse". In the diagonal parse, the dependency head of a token is simply the previous token, as illustrated in Figure 6.

Our model for the joint diagonal parsing and translation ("DiagonalParse") is identical to the "DepParse" model, which has been described in Section 5.1. We only use diagonal matrices during training, instead of the dependency matrices. The main goal of this setup is to examine the benefits of the additional syntactic information for machine translation.

Table 8 documents that the "DiagonalParse" model is very effective. The diagonal parsing precision is, as expected, very high, ranging from 99.95% to 99.99% on the test set. This joint model

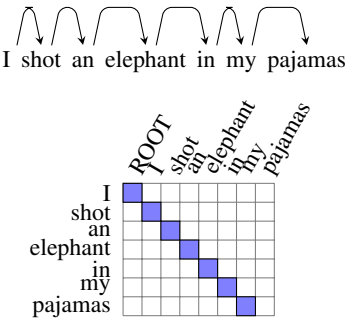


Figure 6: Dummy dependencies with diagonal matrix (the columns represent the heads, the rows are dependents).

	BLEU		Precision	
	Dev	**Test**	**Dev**	**Test**
TransformerBase	37.28	36.66	–	–
Parse from layer 0	38.68	**38.14**	99.97	99.96
Parse from layer 1	**39.11**	38.06	**99.99**	**99.99**
Parse from layer 2	37.85	37.85	99.98	99.98
Parse from layer 3	37.93	37.70	99.97	99.98
Parse from layer 4	37.68	37.47	99.98	99.96
Parse from layer 5	37.53	37.54	99.96	99.95

Table 8: DiagonalParse's results in translation (BLEU) and diagonal parsing (precision) on cs2en. All test BLEU improvements are statistically significant with $p < 0.01$ when compared to the TransformerBase.

also outperformed the baseline in translation task with all its variants (BLEU scores vary from 37.47 to 38.14, compared to 36.66).

Moreover, these results form an observable pattern, in which the best result comes from the model with parsing with the head on layer 0. Parsing from deeper layers still helps to improve translation over baseline, but the BLEU scores decrease. We believe that a possible explanation for this pattern is that the diagonal matrix represents the relation between the preceding token and the current token. This simple sentence structure can serve as an additional positional information to the absolute positional embeddings. Therefore, the sooner the model is forced to recognize this positional information (via training the parsing task), the better it can learn to translate. Another possible explanation is the regularization effect of the diagonal parse.

5.7 Training Speed

While having achieved the results discussed above, the training costs for our multi-task models are comparable to the baseline Transformer. The training time (including internal evaluation every 1000 steps) on a single GPU NVIDIA GTX 1080 Ti needed to reach 250k steps for Transformer-Base was about 1 day and 4 hours while our joint models needed only 10%-13% more time to train on both tasks.

5.8 Self-Attention Patterns in the Encoder

Figure 7 presents the behavior of self-attention mechanism in each layer of our models for the first 100 sentences in the test set. The bin $[0.0, 0.1)$ was excluded from the picture for clarity because most of the self-attention weights fall into this trivial bin. As can be seen from the figure, the layers in which the model was trained to parse display a very sharp attention, i.e. for each head, each word attends to only one or two words from the previous layer. This behavior is apparent in all our multi-task models except the "Parse from layer 0". As mentioned in Section 5.4, this model performed badly on both tasks. While the causality is unclear, we at least see that the sharpness in attention is related to the better performance.

Figure 8 documents another interesting observation (as above, the bin$[0.0, 0.1)$ was excluded). One could perhaps expect that the particular head trained to predict dependencies will have a sharp attention but interestingly, the same sharpness is observed in all heads of the given layer. A possible reason may be due to the vector concatenation and layer normalization after each multi-head attention layer in the Transformer.

6 Discussion

Kiperwasser and Ballesteros (2018) suggest another representation of "DepHeads", which doesn't suffer on unknown words and repeated words. They represent head as an offset from the node's position represented as decimal number, positive to the right, negative to the left. We showed Transformer can easily learn to count words. This representation should be considered in future work.

We let aside a question of vocabulary design for multi-tasking. In T2T's SubwordTextEncoder (STE), the vocabulary is designed unsupervisedly from a training data sample, so that frequent words are represented as single subwords and rare words as sequences of characters. We assume that advantaging different sides of particular tasks on STE input can lead to better quality. This could be combined with different parameters for the con-

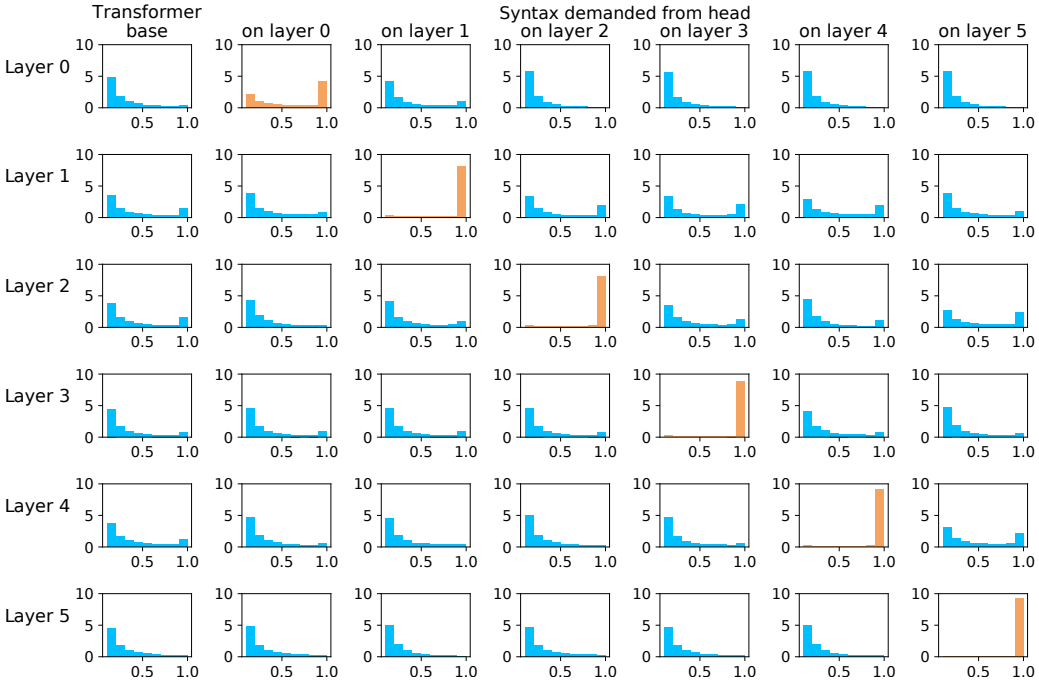


Figure 7: Histogram of normalized self-attention weights in the encoder.

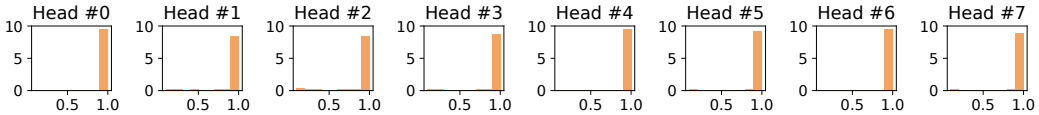


Figure 8: Histogram of self-attention weights in the encoder's layer 4 when parsing from layer 4.

stant task scheduler.

Multiple multi-task experiments (Niehues and Cho, 2017, Zaremoodi and Haffari, 2018, etc.) mention notable gains on small data scenarios. As documented by Koehn and Knowles (2017), under certain training data size, NMT is actually much worse than conventional phrase-based MT. It is unclear if the gains from NMT multi-tasking are obtained also after this critical corpus size, or if they are limited to the data sizes where NMT is ineffective.

One limitation of our setup was that our model was trained on automatic parses. Hence, it would be interesting to fine-tune our model with gold-annotated trees, which could lead to a better parsing performance. We leave this for future work.

7 Conclusion

We proposed two techniques of promoting the knowledge of source syntax in the Transformer model of NMT by multi-tasking and evaluated them at reasonably large data sizes.

The simple data manipulation technique, alternating translation and linearized parsing, is impractical. Learning to translate and parse improves over comparable multi-task setups with uninformative ("dummy") secondary tasks, but overall it performs worse than single-task translation model. In low-resource conditions, the gain from the multi-tasking may be useful.

The other technique, re-interpreting one of the self-attention heads in the Transformer model as the dependency analysis of the sentence, is surprisingly effective. At little or no cost in training time, Transformer learns to translate and parse at the same time. The parse accuracy is reasonable and the translation is significantly better than the baseline. Curiously, very similar gains can be obtained by predicting a "diagonal parse", i.e. linguistically uninformed linear tree. The full explanation of this behavior is yet to be sought for.

References

Roee Aharoni and Yoav Goldberg. 2017. Towards string-to-tree neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Van-*

couver, Canada, July 30 - August 4, Volume 2: Short Papers, pages 132–140.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

Ondřej Bojar, Ondřej Dušek, Tom Kocmi, Jindřich Libovický, Michal Novák, Martin Popel, Roman Sudarikov, and Dušan Variš. 2016. CzEng 1.6: Enlarged Czech-English Parallel Corpus with Processing Tools Dockered. In *Text, Speech, and Dialogue: 19th International Conference, TSD 2016*, number 9924 in Lecture Notes in Computer Science, pages 231–238, Cham / Heidelberg / New York / Dordrecht / London. Masaryk University, Springer International Publishing.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (wmt18). In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 272–307, Belgium, Brussels. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2016. Deep biaffine attention for neural dependency parsing. *CoRR*, abs/1611.01734.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *HLT-NAACL*, pages 199–209. The Association for Computational Linguistics.

Akiko Eriguchi, Yoshimasa Tsuruoka, and Kyunghyun Cho. 2017. Learning to parse and translate improves neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pages 72–78.

Thanh-Le Ha, Jan Niehues, and Alex Waibel. 2016. Toward Multilingual Neural Machine Translation with Universal Encoder and Decoder. In *Proceedings of the International Workshop on Spoken Language Translation*, IWSLT'16, Seattle, USA.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, Zdeněk Žabokrtský, and Magda Ševčíková Razímová. 2006. Prague Dependency Treebank 2.0. LDC2006T01, ISBN: 1-58563-370-4.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's multilingual neural machine translation system: Enabling zero-shot translation. *CoRR*, abs/1611.04558.

Eliyahu Kiperwasser and Miguel Ballesteros. 2018. Scheduled Multi-Task Learning: From Syntax to Translation. *Transactions of the Association for Computational Linguistics*, 6:225–240.

Ondřej Klejch, Eleftherios Avramidis, Aljoscha Burchardt, and Martin Popel. 2015. Mt-compareval: Graphical evaluation interface for machine translation development. *The Prague Bulletin of Mathematical Linguistics*, 104(1):63–74.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*, volume 4, pages 388–395.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand. AAMT, AAMT.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

An Nguyen Le, Ander Martinez, Akifumi Yoshimoto, and Yuji Matsumoto. 2017. Improving sequence to sequence neural machine translation by utilizing syntactic dependency information. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017 - Volume 1: Long Papers*, pages 21–29.

Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2015. Multi-task sequence to sequence learning. *CoRR*, abs/1511.06114.

Maria Nadejde, Siva Reddy, Rico Sennrich, Tomasz Dwojak, Marcin Junczys-Dowmunt, Philipp Koehn, and Alexandra Birch. 2017. Predicting target language ccg supertags improves neural machine translation. In *Proceedings of the Second Conference on Machine Translation, Volume 1: Research Paper*, pages 68–79, Copenhagen, Denmark. Association for Computational Linguistics.

Jan Niehues and Eunah Cho. 2017. Exploiting linguistic resources for neural machine translation using multi-task learning. In *WMT*.

Joakim Nivre, Željko Agić, Lars Ahrenberg, Lene Antonsen, Maria Jesus Aranzabe, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Eckhard Bick, Cristina Bosco, Gosse Bouma, Sam Bowman, Aljoscha Burchardt, Marie Candito, Gauthier Caron, Gülşen Cebiroğlu Eryiğit, Giuseppe G. A. Celano, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Silvie Cinková, Çağrı Çöltekin, Miriam Connor, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Marhaba Eli, Ali Elkahky, Tomaž Erjavec, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta Gonzáles Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà Mỹ, Kim Harris, Dag Haug, Barbora Hladká, Jaroslava Hlaváčová, Petter Hohle, Radu Ion, Elena Irimia, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşıkara, Hiroshi Kanayama, Jenna Kanerva, Tolga Kayadelen, Václava Kettnerová, Jesse Kirchner, Natalia Kotsyba, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Tatiana Lando, Phuong Lê H`ông, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Gustavo Mendonça, Anna Missilä, Verginica Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Shunsuke Mori, Bohdan Moskalevskyi, Kadri Muischnek, Nina Mustafina, Kaili Müürisep, Pinkey Nainwani, Anna Nedoluzhko, Luong Nguy˜ên Thị, Huy`ên Nguy˜ên Thị Minh, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Petya Osenova, Lilja Ovrelid, Elena Pascual, Marco Passarotti, Cenel-Augusto Perez, Guy Perrier, Slav Petrov, Jussi Piitulainen, Emily Pitler, Barbara Plank, Martin Popel, Lauma Pretkalniņa, Prokopis Prokopidis, Tiina Puolakainen, Sampo Pyysalo, Alexandre Rademaker, Livy Real, Siva Reddy, Georg Rehm, Larissa Rinaldi, Laura Rituma, Rudolf Rosa, Davide Rovati, Shadi Saleh, Manuela Sanguinetti, Baiba Saulīte, Yanin Sawanakunanon, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Lena Shakurova, Mo Shen, Atsuko Shimada, Muh Shohibussirri, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Antonio Stella, Jana Strnadová, Alane Suhr, Umut Sulubacak, Zsolt Szántó, Dima Taji, Takaaki Tanaka, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Gertjan van Noord, Viktor Varga, Veronika Vincze, Jonathan North Washington, Zhuoran Yu, Zdeněk Žabokrtský, Daniel Zeman, and Hanzhi Zhu. 2017. Universal dependencies 2.0. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan T. McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The conll 2007 shared task on dependency parsing. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*, pages 915–932.

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 on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Martin Popel. 2018. CUNI Transformer Neural MT System for WMT18. In *Proceedings of the Third Conference on Machine Translation*, pages 486–491, Belgium, Brussels. Association for Computational Linguistics.

Martin Popel and Ondej Bojar. 2018. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43 – 70.

Martin Popel and Zdeněk Žabokrtský. 2010. TectoMT: Modular NLP framework. In *Lecture Notes in Artificial Intelligence, Proceedings of the 7th International Conference on Advances in Natural Language Processing (IceTAL 2010)*, volume 6233 of *Lecture Notes in Computer Science*, pages 293–304, Berlin / Heidelberg. Iceland Centre for Language Technology (ICLT), Springer.

Maja Popovic. 2015. chrf: character n-gram f-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation, WMT@EMNLP 2015, 17-18 September 2015, Lisbon, Portugal*, pages 392–395.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural MT learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1526–1534. The Association for Computational Linguistics.

Miloš Stanojević and Khalil Sima'an. 2014. Fitting sentence level translation evaluation with many dense features. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 202–206, Doha, Qatar. Association for Computational Linguistics.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, MA, USA.

Milan Straka and Jana Straková. 2017. Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112.

Aleš Tamchyna, Marion Weller-Di Marco, and Alexander Fraser. 2017. Modeling target-side inflection in neural machine translation. In *Proceedings of the Second Conference on Machine Translation, Volume 1: Research Paper*, pages 32–42, Copenhagen, Denmark. Association for Computational Linguistics.

Jörg Tiedemann. 2009. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In *Proc. of RANLP*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria.

Ke M. Tran, Arianna Bisazza, and Christof Monz. 2018. The importance of being recurrent for modeling hierarchical structure. *CoRR*, abs/1803.03585.

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 6000–6010.

Weiyue Wang, Jan-Thorsten Peter, Hendrik Rosendahl, and Hermann Ney. 2016. Character: Translation edit rate on character level. In *Proceedings of the First Conference on Machine Translation*, pages 505–510, Berlin, Germany. Association for Computational Linguistics.

Poorya Zaremoodi and Gholamreza Haffari. 2018. Neural machine translation for bilingually scarce scenarios: A deep multi-task learning approach.

Barret Zoph and Kevin Knight. 2016. Multi-Source Neural Translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 30–34, San Diego, California. Association for Computational Linguistics.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.