An Analysis of Encoder Representations in Transformer-Based Machine Translation

Raganato, Alessandro

The Association for Computational Linguistics
2018

Raganato , A & Tiedemann , J 2018 , An Analysis of Encoder Representations in þÿ T r a n s f o r m e r - B a s e d M a c h i n e T r a n s l a t i o n . i n L T a l , G C h r u p a B a & A A l Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP . The Association for Computational Linguistics , Stroudsburg , pp. 287-297 , 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP , Brussels , Belgium , 31/10/2018 . < http://aclweb.org/anthology/W18-5431 >

http://hdl.handle.net/10138/263704

unspecified
acceptedVersion

An Analysis of Encoder Representations in Transformer-Based Machine Translation

Alessandro Raganato and **Jörg Tiedemann**
Department of Digital Humanities
University of Helsinki
{alessandro.raganato,jorg.tiedemann}@helsinki.fi

Abstract

The attention mechanism is a successful technique in modern NLP, especially in tasks like machine translation. The recently proposed network architecture of the *Transformer* is based entirely on attention mechanisms and achieves new state of the art results in neural machine translation, outperforming other sequence-to-sequence models. However, so far not much is known about the internal properties of the model and the representations it learns to achieve that performance. To study this question, we investigate the information that is learned by the attention mechanism in Transformer models with different translation quality. We assess the representations of the encoder by extracting dependency relations based on self-attention weights, we perform four probing tasks to study the amount of syntactic and semantic captured information and we also test attention in a transfer learning scenario. Our analysis sheds light on the relative strengths and weaknesses of the various encoder representations. We observe that specific attention heads mark syntactic dependency relations and we can also confirm that lower layers tend to learn more about syntax while higher layers tend to encode more semantics.

1 Introduction

Machine translation (MT) is one of the prominent tasks in Natural Language Processing, tackled in several ways (Bojar et al., 2017). Neural MT (NMT) has become the de-facto standard with a performance that clearly outperforms the alternative approach of Statistical Machine Translation (Luong et al., 2015b; Bojar et al., 2016; Bentivogli et al., 2016). NMT also improves training procedures due to the end-to-end fashion without tedious feature engineering and complex setups. During recent years, a lot of research has been done on NMT, designing new architectures, starting from the plain sequence-to-sequence model (Sutskever et al., 2014; Cho et al., 2014), to an improved version featuring an attention mechanism (Bahdanau et al., 2015; Luong et al., 2015a), to models that only use attention instead of recurrent layers (Vaswani et al., 2017) and models that apply convolution networks (Gehring et al., 2017a,b). Among the different architectures, the Transformer (Vaswani et al., 2017) has emerged as the dominant NMT paradigm.[1] Relying only on attention mechanisms, the model is fast, highly accurate and has been proven to outperform the widely used recurrent networks with attention and ensembling (Wu et al., 2016) by more than 2 BLEU points. Improved translation quality is typically related to better representation of structural information. While other approaches make use of external information to improve the internal representation of NMT models (Arthur et al., 2016; Niehues and Cho, 2017; Alkhouli and Ney, 2017), the Transformer seems to be able to encode a lot of structural information without explicitly incorporating any structural constraints. However, being a rather new architecture, little is known about what the model exactly learns internally. A better understanding of the internal representations of neural models has become a major challenge in NMT (Koehn and Knowles, 2017).

In this work we investigate the kind of linguistic information that is learned by the encoder. We start by training the Transformer system from English to seven languages, with different training set sizes, resulting in models that are not only trained for different target languages but also with expected differences in translation quality. First, we visually inspect the attention weights of the encoders,

[1] Most submissions for the WMT18 shared task on news (http://matrix.statmt.org/) employ the Transformer architecture.

Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pages 287–297

in order to find linguistic patterns. As the next step, we exploit the attention weights of the network to build a graph and induce tree structures for each sentence, showing whether syntactic dependencies between words have been learned or not in the spirit of Williams et al. (2018) and Liu and Lapata (2018). Additionally, following previous studies on how to analyze the internal representation of neural systems (Adi et al., 2016; Shi et al., 2016; Belinkov et al., 2017a), we probe the encoder weights of the trained models to address different sequence labeling tasks: Part-of-Speech tagging, Chunking, Named Entity Recognition and Semantic tagging. We evaluate the quality of the decoder on a given task to assess how discriminative the encoder representation is for that task. Lastly, in order to check whether the learned information can be transferred across models, we use the encoder weights of a high-resource language pair to initialize a low-resource language pair, inspired by the work of Zoph et al. (2016). We show that, also for the Transformer, the knowledge of an encoder representation can be shared with other models, helping them to achieve better translation quality.

Overall, our analysis leads to interesting insights about strengths and weaknesses of the attention weights of the Transformer, giving more empirical evidence about the kind of information the model is learning at each layer:

- We find that each layer has at least one attention head that encodes a significant amount of syntactic dependencies.
- Consistent with previous findings on the sequence-to-sequence paradigm, probing the encoder to four different sequence labeling tasks reveals that lower layers tend to encode more syntactic information, whereas upper layers move towards semantic tasks.
- The information about the length of the input sentence starts to vanish after the third layer.
- The study corroborates that attention can be used to transfer knowledge between high- and low-resource languages.

2 Architecture

The architecture of the Transformer system follows the so called encoder-decoder paradigm, trained in an end-to-end fashion. Without using any recurrent layer, the model takes advantage of the positional

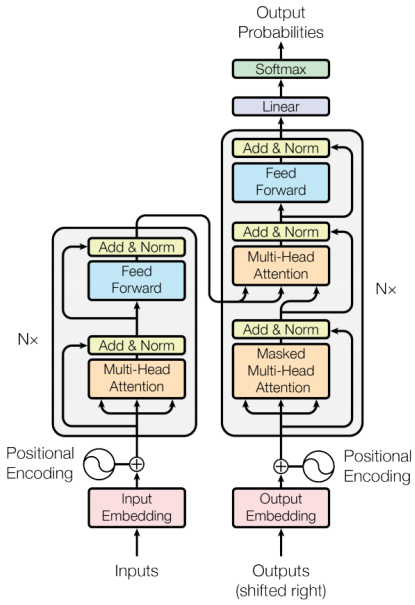


Figure 1: The Transformer architecture (illustration from Vaswani et al. (2017)).

embedding as a mechanism to encode order within a sentence. The encoder, typically stacks 6 identical layers, in which each of them makes use of the so called multi-head attention and of a 2 sublayers feed-forward network, coupled with layer normalization and residual connection (see Figure 1). The multi-head attention mechanism computes attention weights, i.e., a softmax distribution, for each word within a sentence, including the word itself. Specifically:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

where the input consists of queries Q and keys K of dimension d_k, and values V of dimension d_v. The queries, keys and values are linearly projected h times, to allow the model to jointly attend to information from different representation, concatenating the result,

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O$$
$$where\, head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

with parameter matrices $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{model}}$. [2]

[2] As hyper-parameters we used the *base* version from Vaswani et al. (2017).

On top of the multi-head attention there is a feed-forward network that consists of two layers with a ReLU activation in between. Each encoder layer takes as input the output of the previous layer, allowing it to attend to all positions of the previous layer.

The decoder has the same architecture as the encoder, stacking 6 identical layers of multi-head attention with feed-forward networks. However, here there are two multi-head attention sub-layers: i) a decoder self-attention and ii) a encoder-decoder attention. The decoder self-attention attends on the previous predictions made step by step, masked by one position. The second multi-head attention performs an attention between the final encoder representation and the decoder representation.

To summarize, the Transformer model consists of three different attentions: i) the encoder self-attention, in which each position attends to all positions in the previous layer, including the position itself, ii) the encoder-decoder attention, in which each position of the decoder attends to all positions in the last encoder layer, and iii) the decoder self-attention, in which each position attends to all previous positions including the current position.

In this work, we focus on analyzing the structure that is learned by the first type of attention weights of the model, i.e., the encoder self-attention, across different models with different target language and translation quality.

3 Methodology

We aim at analyzing the encoder representation of different models by assessing their quality through several experiments: i) by visualizing the attention weights (Section 5), ii) by inducing tree structure from the encoder weights (Section 6), iii) by probing the encoder as input representation for various prediction tasks (Section 7), and iv) by transferring the knowledge of one encoder to another (Section 8). We start by looking for linguistic patterns through the visualization of the heat-maps of the encoder weights. Next, we use the softmax weights extracted from the multi-head attention to build maximum spanning trees from the input sentences, assessing the quality of the induced tree through dependency parsing. Additionally, we evaluate the ability of the decoder, using a fixed encoder representation as input, on several sequence labeling tasks, measuring how important the input features are for various tasks. As test bed we use four different tasks, ranging from syntax to semantics, i.e, PoS tagging, Chunking, Named Entity Recognition, and Semantic tagging. The assumption is that if a property is well encoded in the input representation then it is easy for the decoder to predict that property. In practice, after training the MT system, we freeze the encoder parameters, and train one decoder layer for each task. The decoder layer is simpler than the original one used for MT; it consists only of one attention head and one feed-forward layer with ReLU activation. Moreover, in order to output the right amount of labels, the decoder also has to learn implicitly the length of the input sentence. Note that our goal is not to beat the state of the art in a given task but rather to analyze the representation of an encoder trained for MT on different tasks referring to different linguistic properties. Finally, to assess whether the knowledge captured within an encoder is general enough to also be used for other models, we test a transfer learning scenario in which we use the encoder representation of a high resource language pair to initialize the encoder of a low resource language pair. Here, we assume that a model is better at encoding abstract linguistic properties if it can share useful information to enhance another weaker model.

	#Training sentences
English → Czech	51.391.404
English → German	25.746.259
English → Estonian	1.064.658
English → Finnish	2.986.131
English → Russian	9.140.469
English → Turkish	205.579
English → Chinese	23.861.542

Table 1: Number of training instances used to train each system.

	newstest 2017	newstest 2018
English → Czech	18.11	17.36
English → German	23.37	34.46
English → Estonian	–	13.05
English → Finnish	15.06	10.32
English → Russian	21.30	18.96
English → Turkish	6.93	6.22
English → Chinese	23.10	23.75

Table 2: BLEU score for the newstest2017 and newstest2018 test data.

4 Model setup

We trained Transformer models[3] from English to seven languages, Czech, German, Estonian, Finnish, Russian, Turkish and Chinese, using the parallel data provided by the WMT18 shared task on news translation.[4] The parallel data come from different sources, mainly from Europarl (Koehn, 2005), News Commentary (Tiedemann, 2012) and ParaCrawl.[5]

The data sets are partially noisy, especially ParaCrawl being on its first release, and to filter out potentially incorrect parallel sentences we used a language identifier[6] to tag each source and target sentence, discarding the sentences that do not match across languages (Stymne et al., 2013; Zariņa et al., 2015). As development set we used the provided newsdev data from the shared task, while using the newstest from WMT 2017 and 2018 as test data. A widely used technique to allow an open vocabulary is byte pair encoding (Sennrich et al., 2016), in which the source and target words are split into subword units. However, in this work we prefer to use the full word forms, allowing us to evaluate and compare the internal representation on standard sequence labeling benchmarks tagged with gold labels on the full word forms. Therefore, we use a large vocabulary of 100K words per language. General statistics on the training data are given in Table 1. As can be seen, we ended up having an heterogeneous amount of data, ranging from 200K for Turkish up to 51M for Czech. We trained each model for maximum 20 epochs, taking the best one according to the development set as model to evaluate. The BLEU score[7] of each model is shown in Table 2. Even though the scores seem low for the Transformer architecture for the MT task, we have to note that each model is trained using full word forms in order to facilitate the analysis of the encoder representation (our results are in line with the comparison between subword units and full word forms done by Sennrich et al. (2016)). We do not aim at beating the best system on the test data, as our main point is to analyze different encoder representations across models with different translation quality and target language.

[3] We used the OpenNMT framework (Klein et al., 2017).

[4] The provided data are already preprocessed and freely available at http://data.statmt.org/wmt18/translation-task/preprocessed/.

[5] https://paracrawl.eu/

[6] We used the fasttext language identifier tool (Joulin et al., 2016b,a) from https://fasttext.cc/docs/en/language-identification.html

[7] We used the SACREBLEU script (Post, 2018), with signature BLEU+case.mixed+lang.en-{targetLanguage}+numrefs.1+smooth.exp+test.wmt{17,18}+tok.13a+version.1.3.0

5 Encoder Evaluation: Visualization

One of the most straightforward ways of understanding the weights of a neural network is by visualizing them. In its base setting, the Transformer employs 6 layers with 8 different attention heads for each of them, making complete visualization difficult. Therefore, we focus only on attention weights with high scores that are visually interpretable.

We discovered four different patterns shared across models: paying attention to the word itself, to the previous and next word and to the end of the sentence (Figure 2). We found that, usually on the first layer, i.e., layer 0, more attention heads focus their weights on the word itself, while on the subsequent layers the network moves the attention more on other words, e.g., on the next and previous word, and to the end of the sentence. This suggests that the transformer tries to find long dependencies between words on higher layers whereas it tends to focus on local dependencies in lower layers.

6 Encoder Evaluation: Inducing Tree Structure

The architecture of the Transformer, linking each word with each other with an attention weight, can be seen as a weighted graph in which the words are the nodes and from which tree structure can be extracted. Even though the models are not trained to produce any trees or to a specific syntax task, we used the attention weights in each layer to extract a tree of the input sentences and inspect whether they reflect a dependency tree.

We evaluated the induced trees on the English PUD treebank from the CoNLL 2017 Shared Task (Zeman et al., 2017). The PUD treebank consists of 1000 sentences randomly taken from on-line newswire and Wikipedia. We measure the performance as Unlabeled Attachment Score (UAS) with the official evaluation script[8] from the shared task, using gold segmentation and tokenization. Plus, given that our weights have no knowledge about the root of the sentence, we decided to use the gold root as starting node for the maximum spanning

[8] conll17_ud_eval.py (version 1.1)

		en → cs	en → de	en → et	en → fi	en → ru	en → tr	en → zh
Layer 0	attention head 0	15.06	10.67	8.79	**31.63**	**17.13**	10.99	13.00
	attention head 1	9.94	**32.90**	8.68	12.58	12.02	10.74	15.76
	attention head 2	15.84	10.62	9.60	10.12	12.08	13.69	15.50
	attention head 3	10.62	15.39	**31.38**	8.31	11.08	9.78	22.79
	attention head 4	17.25	18.12	7.76	25.10	11.75	13.20	10.28
	attention head 5	16.71	14.47	24.24	13.63	12.39	27.55	17.19
	attention head 6	**30.26**	26.28	11.76	10.43	11.55	9.90	**33.26**
	attention head 7	15.17	15.31	9.61	9.51	12.13	**31.81**	9.69
Layer 1	attention head 0	10.95	11.73	11.04	11.47	**36.05**	**26.20**	20.33
	attention head 1	10.91	10.65	27.58	10.88	12.66	11.23	10.72
	attention head 2	10.72	10.87	25.80	27.32	25.64	14.46	**35.77**
	attention head 3	12.21	15.06	15.06	20.90	10.45	14.04	9.62
	attention head 4	**35.08**	13.17	11.14	11.01	18.44	15.83	14.17
	attention head 5	29.04	10.69	10.85	12.51	33.23	27.41	10.84
	attention head 6	15.22	**35.94**	13.55	**35.30**	10.27	11.03	11.59
	attention head 7	22.64	35.89	**35.07**	10.10	13.59	11.82	24.09
Layer 2	attention head 0	**35.46**	12.33	7.40	9.01	35.07	20.53	11.02
	attention head 1	10.29	22.62	32.80	10.98	7.63	10.03	11.55
	attention head 2	19.74	9.02	**33.16**	9.00	20.92	9.52	29.40
	attention head 3	16.23	15.82	13.04	13.98	22.27	14.05	10.71
	attention head 4	23.23	11.07	12.58	**29.43**	35.53	10.85	12.98
	attention head 5	16.78	**33.76**	13.80	14.53	**36.08**	**22.56**	**35.80**
	attention head 6	10.17	22.15	10.23	11.30	12.54	19.38	15.16
	attention head 7	32.01	14.97	13.76	18.36	8.84	11.79	22.12
Layer 3	attention head 0	8.28	9.97	11.05	13.89	**35.03**	18.55	13.80
	attention head 1	**35.20**	24.76	7.99	13.72	20.64	**21.53**	13.03
	attention head 2	10.67	10.54	**22.62**	15.14	9.43	17.03	14.78
	attention head 3	31.13	17.36	12.14	**27.24**	9.27	15.67	11.20
	attention head 4	23.89	**35.59**	8.59	12.18	10.36	13.05	14.89
	attention head 5	14.94	10.12	12.37	7.78	12.62	7.18	19.80
	attention head 6	16.02	13.54	13.38	8.70	10.79	8.80	**38.87**
	attention head 7	13.44	11.81	13.02	14.96	29.10	17.83	9.02
Layer 4	attention head 0	14.45	**27.88**	20.86	11.63	12.84	**25.40**	13.34
	attention head 1	10.37	14.37	17.80	**24.00**	10.72	21.11	22.87
	attention head 2	15.06	10.69	11.82	9.52	13.20	11.36	25.25
	attention head 3	13.47	13.47	14.01	10.92	17.11	12.88	12.29
	attention head 4	**29.66**	17.31	19.45	11.82	10.87	11.76	13.55
	attention head 5	28.07	18.14	**32.87**	22.50	13.76	11.06	**35.40**
	attention head 6	13.35	11.27	9.95	15.49	**27.68**	25.13	11.56
	attention head 7	10.84	25.03	14.93	17.32	13.86	14.00	17.52
Layer 5	attention head 0	**36.02**	29.80	17.37	17.49	**35.56**	16.91	16.75
	attention head 1	28.02	27.23	16.68	28.25	13.04	**28.23**	17.71
	attention head 2	20.20	11.14	19.02	33.38	18.49	7.98	13.45
	attention head 3	11.86	8.30	22.45	14.71	19.17	15.76	19.16
	attention head 4	31.71	19.62	**33.68**	**31.87**	26.42	13.61	27.50
	attention head 5	13.55	15.20	30.73	17.35	11.98	23.13	26.70
	attention head 6	26.02	**35.32**	14.83	24.99	9.77	16.99	**29.73**
	attention head 7	18.63	10.33	15.71	11.01	12.59	25.67	14.79

Table 3: UAS F1-score of the induced trees produced by the attention weights on the English PUD treebank from CoNLL 2017.

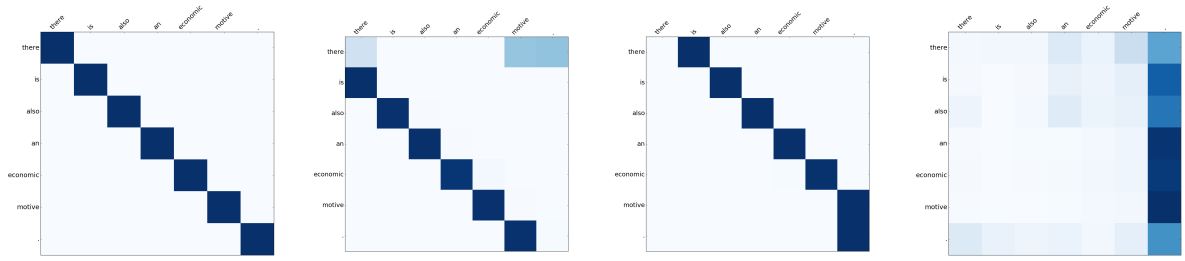
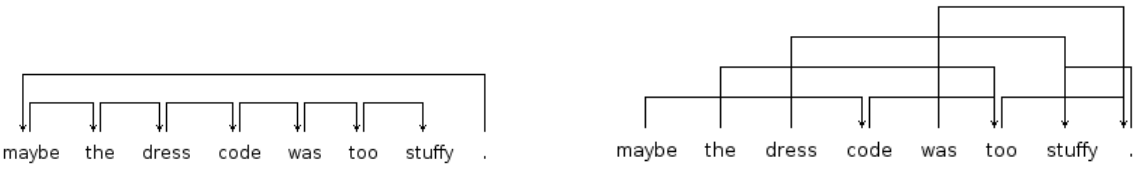


Figure 2: Four examples of the discovered patterns through visualization for the sentence: ”there is also an economic motive .”.



Sample tree from the attention head 1, layer 3

Sample tree from the attention head 4, layer 3

Figure 3: Sample trees induced by the attention weights from the English-Czech model.

tree algorithm. Specifically, we run the Chu-Liu-Edmonds algorithm (Chu, 1965; Edmonds, 1967) for each attention head of each layer of the models to extract the maximum spanning trees. Table 3 shows the F1-score of the induced structures. For comparison purposes, in this dataset, a state of the art supervised parser (Dozat et al., 2017) reaches 88.22 UAS F1-score and our random baseline, i.e., induced trees with random weights and gold root, achieves 10.1 UAS F1-score on average.[9] Given our findings in Section 5, we also computed a left- and right- branching baseline (with golden root), obtaining 10.39 and 35.08 UAS F1-score respectively.

Although our models are not trained to produce trees, the best dependency trees induced on each layer are far better than the random baseline, suggesting that the models are learning some syntactic relationships. However, the best scores do not achieve results much beyond the right branching baseline, showing that it is difficult to encode more complex and longer dependencies.

Overall, for all language pairs we notice the same performance trend across layers. Comparing our low resource language pair, English-Turkish, to the other high resource languages, we can see that the models trained with larger dataset are able to induce better syntactic relationships, while among high resource languages all models are in the same ballpark, without any specific correlation with BLEU score, suggesting that it becomes more difficult to induce better dependency relations at a certain point. Figure 3 shows some examples of induced dependency trees. Interestingly enough, we can see that the trees with higher scores follow the patterns found in Section 5, in which each word is linked to the next one, so encoding most compounds and multi-word expressions. From visualizing other trees, even if they do not belong to the best attention head, we can see that they try to capture longer dependencies, as for *dress* and *stuffy* in the example in Figure 3.

7 Encoder Evaluation: Probing Sequence Labeling Tasks

We evaluated the encoder representation through four different sequence labeling tasks: Part-of-Speech (PoS) tagging, Chunking, Named Entity Recognition (NER) and Semantic tagging (SEM). In this test bed we used the trained weights of the encoder, keeping them fixed, training only one de-

[9] Even though not comparable in this setting, unsupervised systems developed to build dependency trees achieve on an English dataset UAS F1-score ranging from 27.9 to 51.4 when using the output of a PoS tagger system (Alonso et al., 2017).

		en → cs	en → de	en → et	en → fi	en → ru	en → tr	en → zh
POS	layer 0	91.13 / 7.70	91.06 / 8.20	84.49 / 18.20	86.88 / 25.00	89.47 / 6.00	**68.47** / 52.10	90.81 / 12.20
	layer 1	92.79 / **2.90**	93.12 / 4.60	**87.11** / 18.40	**87.58** / **12.40**	90.67 / 10.60	67.53 / 47.00	**92.60** / **7.90**
	layer 2	**93.20** / 5.40	**93.18** / **4.50**	84.99 / **14.70**	86.41 / 15.20	91.86 / **3.90**	68.13 / **45.40**	91.68 / 13.30
	layer 3	92.24 / 9.50	92.31 / 8.60	84.51 / 16.60	85.16 / 18.70	91.46 / 6.00	66.50 / 53.20	89.52 / 19.00
	layer 4	91.66 / 10.80	90.85 / 13.70	82.65 / 23.70	83.46 / 24.40	**91.98** / 12.00	65.66 / 53.90	86.47 / 22.10
	layer 5	87.14 / 19.10	87.83 / 24.10	82.11 / 23.60	80.41 / 33.30	89.47 / 16.30	62.80 / 54.80	82.95 / 31.30
CHUNK	layer 0	90.28 / 4.37	89.78 / 9.49	86.98 / 13.47	87.75 / **8.90**	88.12 / 6.61	**72.64** / **31.21**	90.37 / 5.42
	layer 1	92.98 / **4.32**	92.91 / 3.58	**88.00** / **11.78**	**88.92** / 10.19	91.16 / **4.03**	71.59 / 40.81	92.76 / **6.71**
	layer 2	**93.56** / 6.56	**93.92** / **3.53**	**88.00** / 12.28	88.65 / 13.22	91.60 / 5.82	70.25 / 37.38	**93.40** / 11.18
	layer 3	93.46 / 12.33	**93.92** / 10.14	87.56 / 14.36	87.41 / 19.93	**92.78** / 5.91	69.20 / 46.17	90.83 / 16.90
	layer 4	92.68 / 14.66	92.83 / 12.77	85.80 / 22.81	86.60 / 20.13	92.73 / 12.72	68.54 / 51.04	89.30 / 19.09
	layer 5	90.87 / 14.46	89.92 / 16.60	85.34 / 19.88	84.04 / 27.14	90.95 / 15.11	65.01 / 53.33	82.82 / 31.71
NER	layer 0	91.18 / 23.75	92.71 / 12.02	87.21 / 33.03	89.38 / 29.53	91.29 / 14.58	86.49 / 39.47	91.72 / **11.05**
	layer 1	93.29 / 9.80	93.36 / 7.27	88.65 / **15.99**	90.14 / **20.77**	92.22 / 10.07	85.66 / 38.14	92.93 / 11.13
	layer 2	**93.83** / **7.11**	94.13 / **11.13**	87.46 / 37.30	90.20 / 26.47	**93.20** / **8.12**	86.52 / 43.05	**93.72** / 12.35
	layer 3	93.23 / 16.53	**94.32** / 14.85	**88.95** / 33.31	**90.22** / 26.57	93.14 / 9.42	86.82 / **37.68**	93.07 / 18.32
	layer 4	93.72 / 11.81	93.93 / 12.51	88.57 / 40.55	89.14 / 34.28	92.02 / 12.65	**87.21** / 53.99	91.93 / 26.95
	layer 5	92.62 / 21.63	94.11 / 17.35	87.64 / 30.13	89.40 / 31.49	92.33 / 13.98	86.06 / 44.25	92.35 / 30.08
SEM	layer 0	83.99 / 13.56	84.05 / 13.35	81.87 / 14.73	81.99 / 14.69	83.36 / 14.07	79.04 / **16.87**	84.08 / 13.63
	layer 1	84.84 / 12.48	85.27 / 12.16	82.25 / **14.11**	82.70 / 13.97	84.12 / 13.26	78.80 / 17.10	84.93 / 11.88
	layer 2	85.17 / 11.95	85.11 / 12.16	82.28 / 14.25	82.76 / 14.85	84.09 / 13.03	78.26 / 18.09	85.40 / 11.74
	layer 3	85.34 / 12.02	84.77 / 11.45	82.17 / 14.41	82.82 / 14.00	**85.21** / 12.32	**79.22** / 17.28	84.79 / 11.91
	layer 4	85.29 / **11.38**	**85.91** / **9.93**	**82.44** / 14.50	**83.19** / **13.77**	84.26 / 12.50	78.36 / 19.26	85.38 / 11.42
	layer 5	**86.27** / 11.68	85.71 / 10.78	82.27 / 14.55	82.96 / 13.84	84.56 / **11.79**	78.67 / 18.78	**85.98** / **10.62**

Table 4: Results in terms of precision for each test set (↑, on the left side of each cell), together with the error rate on the sentence length (↓, on the right side of each cell).

	#labels	#training sentences	#testing sentences	average sent. length
PoS	17	12543	1000	21.2
Chunk	22	8042	2012	23.5
NER	9	14987	3684	12.7
SEM	80	62739	4351	6.4

Table 5: Statistics of the evaluation benchmarks used for the probing task.

coder layer using one attention head and one feed-forward layer. We then assess the quality of the encoder representation across stacked layers.

Evaluation Benchmarks. We used a standard benchmark for each task: the Universal Dependencies English Web Treebank v2.0 (Zeman et al., 2017) for PoS tagging, the CoNLL2000 Chunking shared task (Tjong Kim Sang and Buchholz, 2000), the CoNLL2003 NER shared task (Tjong Kim Sang and De Meulder, 2003), and the annotated data from the Parallel Meaning Bank (PMB) for Semantic tagging (Abzianidze et al., 2017). Each benchmark provides its own training, development and test data, except chunking in which we use 10% of the training corpus as validation, and the PMB in which we used the silver portion for training and the gold portion for test and dev (following the 80-20 split).[10] Table 5 reports general statistics on each benchmark, regarding the granularity of each task, the number of training and testing instances, and the average length of the test sentences.

Evaluation Results. Table 4 reports the performance for each task and stacked layers, together with the error rate for sentence length prediction. For each language pair, we can see that the syntax information, i.e., the PoS task, is encoded mostly in the first 3 layers, corroborating the results in Section 6, while moving towards more semantic tasks, as NER and SEM we can see that in general the decoder needs more encoder layers to achieve better results. Another interesting finding is provided by the length mismatch between the output of the models and the gold labels. Clearly the models encode the information about the sentence length in the first three layers, and then the information starts to vanish with an increase of the error rate. The only exception is given by the SEM task, but as can be seen from the statistics in Table 5, the

[10] We used the sem-0.1.0 version.

	newstest 2017	newstest 2018
English → Turkish	6.93	6.22
English TL1 → Turkish	8.72	7.93
English TL2 → Turkish	7.82	6.91

Table 6: BLEU score for the newstest2017 and newstest2018 test data for the transfer learning experiment.

average sentence length is very short and so it is easier to predict. Overall, comparing the performance reached on these probing tasks with the BLEU score of each model, we can see again that the high resource language pairs achieve better results compared to our low resource language pair. Moreover, we notice that in general higher BLEU score correspond to higher probing results, confirming the trend that encoding linguistic proprieties within the encoder representation go on par with better translation quality (Niehues and Cho, 2017; Kiperwasser and Ballesteros, 2018).

8 Encoder Evaluation: Transfer learning

To assess whether the knowledge encoded in the attention units can help other models in a low resource scenario, we additionally carried out an evaluation of the encoder representation in a transfer learning task. Similar to Zoph et al. (2016), we used the encoder weights from one high resource language, i.e., English-German, to train a Transformer system for our low resource language pair, English-Turkish. We provide two experiments: i) initializing and fine tuning the encoder weights (**TL1**), ii) initializing and keeping the encoder weights fixed (**TL2**). Table 6 shows the BLEU scores of the systems evaluated with and without transferring the encoder parameters. Both transfer learning settings are helpful to the decoder to reach a better translation quality, with almost 2 BLEU point more on the best scenario. Starting with a better encoder representation, taken from a high resource language pair, and then fine tuning the parameters on the low resource language achieves the best result, matching and corroborating previous findings on recurrent networks (Zoph et al., 2016).

9 Related Work

The problem of interpreting and understanding neural networks is attracting more and more interest and work, with so many models and new architectures being published continuously each year. One of the first techniques to examine a neural network involves the analysis of activation patterns of the hidden layers (Elman, 1991; Giles et al., 1992). Nowadays, given its popularity, recurrent neural networks are the most evaluated networks, mainly investigated on the structures and linguistic properties they are encoding (Linzen et al., 2016; Enguehard et al., 2017; Kuncoro et al., 2017; Gulordava et al., 2018).

Traditionally, a common way to inspect neural networks is by visualizing the hidden representation trained for a specific task (Ding et al., 2017; Strobelt et al., 2018a,b), and to evaluate them by assessing the properties through downstream tasks (Chung et al., 2014; Greff et al., 2017).

Other recent studies look for hidden linguistic units that provide information on how the network works (Karpathy et al., 2015; Qian et al., 2016; Kádár et al., 2017), while another line of analysis probes the representation learned by a neural network as input to a classifier of another task (Shi et al., 2016; Adi et al., 2016; Belinkov et al., 2017a; Tran et al., 2018).

The most closely related work is by Belinkov et al. (2017b), in which they investigate the representation learned by the encoder of a sequence-to-sequence NMT system across different languages. Unlike them, we studied a neural network without any recurrent layers, which allows us to induce a tree representation from the input sentence, probing the encoder representation towards more downstream tasks, and showing that the attention weights can also be used to transfer knowledge to low-resource languages.

10 Conclusion

In this paper we investigated the kind of information that is captured by the encoder representation of a Transformer model trained for the task of Machine Translation. We analyzed and compared experimentally different models across several languages, including the visualization of weights, building tree structure from each sentence, probing the representation to four different sequence-labeling tasks and by transferring the encoder knowledge to a low resource language. Unlike most previous studies, where the analysis is made only on RNNs, we examined an architecture based on attention only. Our experimental evaluation sheds lights on interesting findings about dependency relations and syntactic and semantic

behavior across layers. In future work, we plan to extend the analysis with probing tasks to evaluate other linguistic properties (Conneau et al., 2018) as well as to a recent evaluation dataset (Sennrich, 2017), tackling also the attention weights between the encoder and the decoder.

Acknowledgments

The work in this paper is supported by the Academy of Finland through project 314062 from the ICT 2023 call on Computation, Machine Learning and Artificial Intelligence. We would also like to acknowledge NVIDIA and their GPU grant.

References

Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. 2017. The parallel meaning bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 242–247, Valencia, Spain. Association for Computational Linguistics.

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207*.

Tamer Alkhouli and Hermann Ney. 2017. Biasing attention-based recurrent neural networks using external alignment information. In *Proceedings of the Second Conference on Machine Translation*, pages 108–117.

Héctor Martínez Alonso, Željko Agić, Barbara Plank, and Anders Søgaard. 2017. Parsing universal dependencies without training. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 230–240.

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.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR Workshop*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017a. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 861–872.

Yonatan Belinkov, Lluís Màrquez, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2017b. Evaluating layers of representation in neural machine translation on part-of-speech and semantic tagging tasks. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1–10.

Luisa Bentivogli, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. 2016. Neural versus phrase-based machine translation quality: a case study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 257–267, Austin, Texas. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, et al. 2017. Findings of the 2017 conference on machine translation (wmt17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214.

Ondrej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation. In *ACL 2016 First Conference on Machine Translation (WMT16)*, pages 131–198. The 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.

Yoeng-Jin Chu. 1965. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14:1396–1400.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136. Association for Computational Linguistics.

Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), volume 1, pages 1150–1159.

Timothy Dozat, Peng Qi, and Christopher D Manning. 2017. Stanford's graph-based neural dependency parser at the conll 2017 shared task. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards, B*, 71:233–240.

Jeffrey L Elman. 1991. Distributed representations, simple recurrent networks, and grammatical structure. *Machine learning*, 7(2-3):195–225.

Émile Enguehard, Yoav Goldberg, and Tal Linzen. 2017. Exploring the syntactic abilities of rnns with multi-task learning. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 3–14.

Jonas Gehring, Michael Auli, David Grangier, and Yann Dauphin. 2017a. A convolutional encoder model for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 123–135.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017b. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*.

C Lee Giles, Clifford B Miller, Dong Chen, Guo-Zheng Sun, Hsing-Hen Chen, and Yee-Chun Lee. 1992. Extracting and learning an unknown grammar with recurrent neural networks. In *Advances in neural information processing systems*, pages 317–324.

Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. 2017. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 1195–1205.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016a. Fasttext.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016b. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Akos Kádár, Grzegorz Chrupała, and Afra Alishahi. 2017. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics*, 43(4):761–780.

Andrej Karpathy, Justin Johnson, and Li Fei-Fei. 2015. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*.

Eliyahu Kiperwasser and Miguel Ballesteros. 2018. Scheduled multi-task learning: From syntax to translation. *Transactions of the Association for Computational Linguistics*, 6:225–240.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proc. ACL*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86.

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.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, Graham Neubig, and Noah A. Smith. 2017. What do recurrent neural network grammars learn about syntax? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1249–1258, Valencia, Spain. Association for Computational Linguistics.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Yang Liu and Mirella Lapata. 2018. Learning structured text representations. *Transactions of the Association for Computational Linguistics*. To appear.

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

Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. 2015b. Addressing the rare word problem in neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 11–19.

Jan Niehues and Eunah Cho. 2017. Exploiting linguistic resources for neural machine translation using multi-task learning. In *Proceedings of the Second Conference on Machine Translation*, pages 80–89.

Matt Post. 2018. A call for clarity in reporting bleu scores. *arXiv preprint arXiv:1804.08771*.

Peng Qian, Xipeng Qiu, and Xuanjing Huang. 2016. Analyzing linguistic knowledge in sequential model of sentence. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 826–835.

Rico Sennrich. 2017. How grammatical is character-level neural machine translation? assessing mt quality with contrastive translation pairs. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 376–382.

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)*, volume 1, pages 1715–1725.

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*, pages 1526–1534.

Hendrik Strobelt, Sebastian Gehrmann, Michael Behrisch, Adam Perer, Hanspeter Pfister, and Alexander M Rush. 2018a. Seq2seq-vis: A visual debugging tool for sequence-to-sequence models. *arXiv preprint arXiv:1804.09299*.

Hendrik Strobelt, Sebastian Gehrmann, Hanspeter Pfister, and Alexander M Rush. 2018b. Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE transactions on visualization and computer graphics*, 24(1):667–676.

Sara Stymne, Christian Hardmeier, Jörg Tiedemann, and Joakim Nivre. 2013. Tunable distortion limits and corpus cleaning for smt. In *WMT 2013; 8-9 August; Sofia, Bulgaria*, pages 225–231. Association for Computational Linguistics.

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

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Erik F Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7*, pages 127–132. Association for Computational Linguistics.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 142–147. Association for Computational Linguistics.

Ke Tran, Arianna Bisazza, and Christof Monz. 2018. The importance of being recurrent for modeling hierarchical structure. *arXiv preprint arXiv:1803.03585*.

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*, pages 5998–6008.

Adina Williams, Andrew Drozdov, and Samuel R Bowman. 2018. Do latent tree learning models identify meaningful structure in sentences? *Transactions of the Association of Computational Linguistics*, 6:253–267.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Ieva Zariņa, Pēteris Ņikiforovs, and Raivis Skadiņš. 2015. Word alignment based parallel corpora evaluation and cleaning using machine learning techniques. In *Proceedings of the 18th Annual Conference of the European Association for Machine Translation*.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajic, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, et al. 2017. Conll 2017 shared task: multilingual parsing from raw text to universal dependencies. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19.

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.