What have Transformers Learned from Relation Extraction?

Xu Liang*, Peng Shi**, Yasufumi Taniguchi*, and Takahiro Kubo*

*Strategy Technology Center, TIS Inc.
{ryo.sho,taniguchi.yasufumi,kubo.takahiro}@tis.co.jp
**David R. Cheriton School of Computer Science, University of Waterloo
peng.shi@uwaterloo.ca

Abstract

Recently, the transformer-based pre-trained models have made significant progress in the Relation Extraction (RE) task. However, what the transformer models have learned from the RE task fine-tuning is still unclear. To answer this question, analyzing the attention distribution and figuring out what kinds of tokens contribute more to the final prediction can be helpful. In this work, we hypothesize that the fine-tuned transformer will pay more attention to the Shortest Dependency Path (SDP) tokens than non-SDP tokens. To validate this hypothesis, we define a metric, Attention Target Averaged Count (ATAC), to measure the importance of different token types for the RE task. Experiments show that the transformer model does utilize the SDP information during the inference: There is a considerable growth on the ATAC score regarding the SDP type after the fine-tuning. Moreover, a much sharper performance drop is observed after pruning the SDP attention heads compared with pruning those non-SDP attention heads.

1 Introduction

Relation Extraction (RE) is a task to detect the predefined relations between two given entities in a sentence (see example in Figure 1). Recently, transformer-based models such as BERT (Devlin et al., 2019) are applied to RE task and achieve impressive results (Shi and Lin, 2019; Wu and He, 2019; Baldini Soares et al., 2019; Wang et al., 2019; Joshi et al., 2019; Alt et al., 2019).

Meanwhile, the effectiveness of transformer motivates a growing body of research investigating what aspects of language they are able to learn. Clark et al. (2019) analyzed the attention head pattern in the pre-training phase, showing that certain attention heads specialize to specific dependency relations. Goldberg (2019); Hewitt and Manning (2019); Tenney et al. (2019); Liu et al. (2019); Tenney et al. (2019); Jawahar et al. (2019); Htut et al. (2019) showed that representation constructed from BERT can encode rich syntactic phenomena.

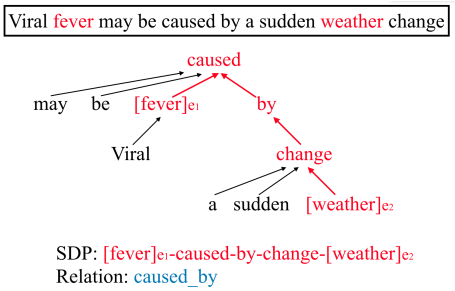

Figure 1: A dependency tree example. Entities are "fever" and "weather". The Shortest Dependency Path (SDP) between two entities is colored in red (edges and tokens).

However, none of them explore the sources of the performance gains in the fine-tuning phase for the RE task. In this work, we try to answer the question by hypothesizing that the Shortest Dependency Path (SDP) information contributes to the effectiveness of the transformer on RE task. To verify this hypothesis, we design the experiments with three steps: We firstly classify the tokens into different types, and then investigate the importance of them based on the attention head pattern changes before and after the fine-tuning phase, measured by our proposed metric, Attention Target Averaged Count (ATAC). We further verify the importance of each token type by the head pruning experiments, by observing the performance drops.

The major contribution of our work is to answer the question: What kind of knowledge do the transformers learn in RE task? We find that transformers automatically learn SDP information and prove that the learned SDP information is a key factor for the performance gains. Our work makes the transformer models more interpretable and connects the existing feature engineering with deep transformer

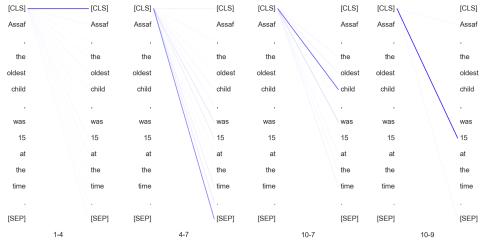

Figure 2: Attention patterns in different heads. From left to right: [CLS] → [CLS], [CLS] → [SEP], [CLS] → Others, [CLS] → SDP. The <layer number>-<head number> denotes a particular attention head.

models. To the best of our knowledge, this is the first study on analyzing transformer models for the RE task from the syntactic perspective.

2 Model and Dataset

Model We use BERT (Devlin et al., 2019) as the transformer (Vaswani et al., 2017) based model for the experiment. In the fine-tuning experiments, we construct the input sequence in the form of <[CLS], TOKEN, ..., [SEP]>, where [CLS] and [SEP] are two special tokens. To prevent overfitting, we replace the entity mentions in the sentence with masks (Zhang et al., 2017), composed of argument type (subject or object) and entity type (such as location and person), e.g., SUBJ-LOC, denoting that the subject entity is a location. The hidden representation of [CLS] from BERT is used for the final prediction over the predefined relation types. Cross-entropy loss function is used. With the fine-tuned model, we extract the attention map of each self-attention head for further analysis.

Dataset TACRED (Zhang et al., 2017) is a widely used relation extraction dataset, covering 41 relation types (e.g., *per:schools_attended*). Examples are labeled as *no_relation* if no defined relation is held. Our analysis is based on two settings: **full dataset** (full 15,509 test samples) and **positive dataset** (3,325 positive samples where samples tagged with *no_relation* are removed). F_1 is the evaluation metric for the dataset.

Setting We use BERT `base` (12 layers and 144 attention heads in total) for experiments. `<layer number>-<head number>` denotes a particular attention head. We fine-tune the model on TACRED dataset, achieving 70.46% F_1 score on average with three different random seeds (these three models obtain 70.87%, 70.22%, 70.3% on F_1 score, respectively), and we use the model that achieves 70.87% for the analysis.

3 Methodology

Our analysis method has three steps: Token types classification, attention pattern analysis, and head pruning based verification.

Token Type Classification: For each example, we classify the tokens into four token types, namely `SDP` tokens (the tokens in the Shortest Dependency Path, including the two entities), `[CLS]` token, `[SEP]` token, and `Others`. To obtain the `SDP` tokens, we leverage the off-the-shelf dependency parser[1] to acquire the dependency tree and extract the Shortest Dependency Path between two given entities. The tokens fell along the path are classified as `SDP` tokens (see example in Figure 1). `[CLS]` and `[SEP]` are the two special tokens in BERT model. And other tokens automatically fall in `Others` type.

Attention Pattern Analysis: Based on the token type classification, we then observe how the attention patterns of the self-attention heads change. To achieve this, we firstly define the *major attention token type* (MATT) of a head as:

$$\mathrm{MATT}(h, s) = type(\mathrm{argmax}_t \alpha_{h,s,t}), \quad (1)$$

where h, s, t and α denote an attention head, attention source token, attention target token and attention weight, respectively. In this work, the model has totally 12 × 12 = 144 heads in total. When a specific attention head h is applied to a token s, namely the attention source token, the attention weight α from s to another token t, namely the attention target token, can be computed. We define the $type$ of t that has largest attention weight as MATT of the attention head h on the token s. For example, in Figure 2 where the darker color denotes the larger attention weight score, a $MATT(h = \texttt{1-4}, s = \textit{[CLS]}) = \texttt{[CLS]}$ and $MATT(h = \texttt{10-7}, s = \textit{[CLS]}) = \texttt{Other}$, because the token "child" has the largest attention weight α and its type is `Others` in the context.

Intuitively, the MATT shows the token type that an attention head cares about the most. Based on all the examples in the corpus, we define the Attention Target Averaged Count (ATAC) of a specific MATT p that an attention head h attends to:

$$\mathrm{ATAC}(h, s, p) = \frac{\sum \sigma(\mathrm{MATT}_i(h, s) = p)}{\mathrm{N}}, \quad (2)$$

where N is the total sample size, the σ function equals to 1 when the expres-

[1] spacy: https://spacy.io

Head	ATAC for different MATTs			
	SDP	Others	[CLS]	[SEP]
1-4	0	0	1	0
3-10	0	0	0	1
8-1	0.0036	0.9946	0.0012	0.0012
11-9	1.1582	0.8277	0	0
12-11	1.2009	0.7408	0.0033	0.0039

Table 1: Head examples of ATAC for different MATTs

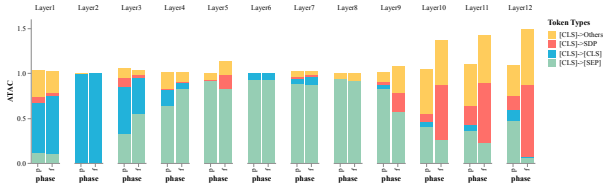

Figure 3: ATAC for different MATTs before and after fine-tuning phase grouped by layers.

sion true otherwise equals to 0, and $p \in \{\texttt{SDP}, \texttt{[CLS]}, \texttt{[SEP]}, \texttt{Others}\}$.

Considering the [CLS] hidden state represents the whole sentence, it will be the only attention source token in this work. Intuitively, $\text{ATAC}(h, s = \textit{[CLS]}, p)$ shows the average count of an attention head attending to the token type p from [CLS]: Higher value implies that this attention head cares more about the token type p when the attention source token is [CLS]. For example, in Table 1, the ATAC (the attention source token is [CLS]) of some attention heads are shown. Head `11-9` attends to 1.1582 `SDP` tokens, and 0.8277 `Others` tokens. We can further aggregate $\text{ATAC}(h, s = \textit{[CLS]}, p)$ over all the attention heads globally or within a layer: $\text{ATAC}(s = \textit{[CLS]}, p) = \sum_h \text{ATAC}(h, s = \textit{[CLS]}, p)$, which shows the attention patterns in higher level. With the ATAC metric, we can further determine the token types on which the attention heads focus more after the fine-tuning phase.

Head Pruning Based Verification: In this step, we quantify the token type importance with the heads pruning experiments. Firstly, we gather top-k (k=10 in this work) heads based on the value of $ATAC(h, s = \textit{[CLS]}, p)$ for a specific token type p. Then we mask out these heads during the model inference phase, and observe the changes of the model performance. Intuitively, if those heads that are attending to important tokens are masked out during the inference, the evaluation metric will drop significantly.

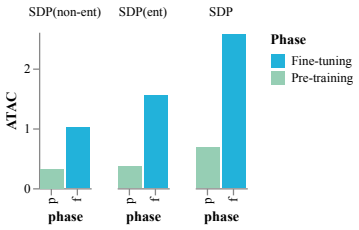

Figure 4: ATAC on `SDP(non-ent)`, `SDP` and `SDP(ent)` before and after fine-tuning phase.

4 Attention Pattern Analysis

In this section, we give a comprehensive analysis for the head attention pattern changes before and after the fine-tuning phase to determine the important token types. Figure 3 demonstrates how ATAC changes before and after fine-tuning phases in layer level. We can observe that the increase of the ATAC regarding `Others` and `[SDP]` reveals mainly in top 4 layers (layer 9, 10, 11, 12). Correspondingly, the decrease of the ATAC regarding `[SEP]` also locates in top 4 layers. The variances of the ATAC in other layers are small.

Attend to `SDP` In Figure 3, when the layers are close to the output layer, such as layer 10, 11 and 12, it is clear that the attention heads attend more to `[SDP]` tokens after fine-tuning phase. Based on the ATAC aggregated within a layer, $ATAC(h, s = \textit{[CLS]}, p = \textit{[SDP]})$ has the largest increase after the fine-tuning, when compared with other token types, such as `Others`, which remains unchanged or slightly increases. This pattern reveals that the fine-tuned model captures the representative syntactic features from the raw sentences. For example, in Figure 2, the subject entity token and object entity token are "Assaf" and "15", respectively and the relation is *per:age* between these two entities. Before the fine-tuning phase, the head `10-7` attends to token "child". But after the fine-tuning phase, the head `10-7` attends to token "15".

In order to further investigate the pattern changes of `SDP` tokens, we categorize `SDP` tokens into `SDP(ent)` (entity tokens in SDP) tokens and `SDP(non-ent)` (non-entity tokens in SDP) tokens, represented as $ATAC(s = \textit{[CLS]}, p = \textit{[SDP(non-ent)]})$ and $ATAC(s = \textit{[CLS]}, p = \textit{[SDP(ent)]})$. In Figure 4, as we mask the both entities as entity types to prevent overfitting, it is not surprising that the `SDP(ent)` tokens attract more attention in the fine-tuning phase. But we can also observe that $ATAC(s = \textit{[CLS]}, p = \textit{[SDP(non-ent)]})$ increases with a large margin, in-

Rank	ATAC-based heads						Gradient-based heads
	SDP	SDP(ent)	SDP(non-ent)	Others	[CLS]	[SEP]	
1	10-9	11-9	12-6	8-1	1-4	1-12	11-9
2	12-11	12-3	11-3	1-8	1-6	3-6	11-5
3	11-9	10-4	11-10	1-5	1-7	3-7	10-9
4	12-3	10-2	11-6	4-9	1-9	3-10	11-2
5	10-2	12-11	9-1	10-4	2-1	3-11	9-1
6	5-2	5-2	12-7	1-10	2-2	4-5	10-4
7	10-4	10-9	10-10	9-3	2-3	4-6	10-8
8	11-5	11-5	10-7	9-6	2-4	4-8	12-11
9	11-8	11-8	10-3	11-9	2-5	4-10	3-6
10	12-12	12-12	10-6	12-3	2-6	4-11	12-12

Table 2: Top-10 important heads based on ATAC-based method and gradient-based method

Methods		F1 after pruning	
		Positive Dataset	Full Dataset
Fine-tuned(no pruning)		80.77	70.87
ATAC-based	**SDP**	**-16.02**	**-9.58**
	SDP(ent)	-14.60	-8.49
	SDP(non-ent)	-5.49	-3.12
	Others	-4.77	-1.82
	[CLS]	-1.06	-0.47
	[SEP]	-0.06	-0.50
Gradient-based		-10.11	-5.10

Table 3: F_1 scores (%) after pruning top-10 heads based on different methods. Positive means 3,325 test data with real relation. Full data means 15,509 test data, containing non-relation.

dicating that the model can automatically detect the SDP structure in the raw sentence, to encode distinctive features for classification.

Attend to `[Others]` When the layers are very close to the output layer, such as layer 12, the attention heads attend more to `[Others]`. However the change is less significant than the one of `[SDP]` because the `[Others]` tokens are less informative than the `[SDP]` tokens, attracting less attention overall.

Attend to `[CLS]` and `[SEP]` An obvious phenomenon in Figure 3 is that the heads in the bottom three layers (layer 1, 2, 3) pay most of the attention on the `[CLS]` token. Also, similar to the results from Clark et al. (2019), we also find that a significant number of heads attend to `[SEP]` tokens in the middle layers (from layer 4 to layer 9 in Figure 3). The absolute value changes of $ATAC(s = \textit{[CLS]}, p = \textit{[CLS]})$ are subtle when we compare fine-tuned models with the pretrained one in bottom layers. Similarly for $ATAC(s = \textit{[CLS]}, p = \textit{[SEP]})$ in middle layers. When the layers are closer to the top layer, these two values start to decrease in both stages because the top layers are more important for extracting task specific features and attending to `[CLS]` can not contribute to producing useful information.

5 Head Pruning Based Verification

Until now, our analysis shows that in the fine-tuned model, the attention weight of `[SDP]` boosts, indicating the importance of `[SDP]` token type. In this section, we further quantify it with the head pruning experiments.

Table 2 (Columns 2-7) shows the ranked top-10 heads based on the value of $ATAC(h, s = \textit{[CLS]}, p)$ ($p \in \{$`SDP`, `SDP(ent)`, `SDP(non-ent)`, `Others`, `[CLS]`, `[SEP]`$\}$). After pruning these heads regarding different token types, Table 3 shows the quantitative results. If we mask the top-10 heads that are attending to `SDP` tokens, the F_1 score on **positive dataset** drops 16% and the one on **full dataset** drops 9.58% in absolute value. Both scores drop far larger than pruning the heads regarding other token types. The results prove that the SDP information learned during the fine-tuning phase makes the greatest contribution than other token types. The results of `SDP(ent)` and the `SDP(non-ent)` show that the `SDP(ent)` is the major contributions in `SDP` token type. And the `SDP(non-ent)` makes more contributions than other token types such as `Others`, `[CLS]` and `[SEP]`.

We further compare our ATAC-based head pruning method with the gradient-based method (Michel et al., 2019). From Table 2 (Column 2 and Column 8), we can observe that 6 attention heads overlap (`10-9`, `12-11`, `11-9`, `10-4`, `11-5`, `12-12`) in the ranked top-10 heads lists obtained from the ATAC-based method (`SDP`) method (based on the value of $ATAC(h, s = \textit{[CLS]}, p = \textit{[SDP]})$) and the gradient-based method. After the head pruning, we observe that ATAC-based (`SDP`) method leads to larger performance drop than the gradient-based method (14.60% v.s. 10.11% on **positive dataset** and 9.58% v.s. 5.10% on **full dataset**), indicating that the ATAC-based (`SDP`) is a better measurement in determining the head importance for RE task than the gradient-based method.

6 Conclusion and Future Work

In this work, we propose a new metric, Attention Target Averaged Count (ATAC), for tracking attention patterns. With this metric, we prove our hypothesis that the transformers have learned shortest dependency path knowledge in the fine-tuning phase for relation extraction (RE) task, contributing to the performance gains, which is further supported by the ATAC-based head pruning experiments. Based on the conclusion, exploring how to encode SDP information more effectively for transformers will be an interesting direction.

References

Christoph Alt, Marc Hübner, and Leonhard Hennig. 2019. Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. In *Proceedings of ACL*, pages 1388–1398. Association for Computational Linguistics.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of ACL*, pages 2895–2905. Association for Computational Linguistics.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP*, pages 276–286. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186. Association for Computational Linguistics.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of NAACL-HLT*, pages 4129–4138. Association for Computational Linguistics.

Phu Mon Htut, Jason Phang, Shikha Bordia, and Samuel R. Bowman. 2019. Do attention heads in bert track syntactic dependencies? *arXiv preprint arXiv:1911.12246*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of ACL*, pages 3651–3657. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2019. Spanbert: Improving pre-training by representing and predicting spans. *arXiv preprint arXiv:1907.10529*.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of NAACL-HLT*, pages 1073–1094. Association for Computational Linguistics.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *NeurIPS*, pages 14014–14024.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *CoRR*, abs/1904.05255.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *ICLR*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*, pages 5998–6008.

Haoyu Wang, Ming Tan, Mo Yu, Shiyu Chang, Dakuo Wang, Kun Xu, Xiaoxiao Guo, and Saloni Potdar. 2019. Extracting multiple-relations in one-pass with pre-trained transformers. In *Proceedings of ACL*, pages 1371–1377. Association for Computational Linguistics.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of CIKM*, pages 2361–2364.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of EMNLP*, pages 35–45, Copenhagen, Denmark. Association for Computational Linguistics.