Have Attention Heads in BERT Learned Constituency Grammar?

Ziyang Luo
Uppsala University
Ziyang.Luo.9588@student.uu.se

Abstract

With the success of pre-trained language models in recent years, more and more researchers focus on opening the "black box" of these models. Following this interest, we carry out a qualitative and quantitative analysis of constituency grammar in attention heads of BERT and RoBERTa. We employ the syntactic distance method to extract implicit constituency grammar from the attention weights of each head. Our results show that there exist heads that can induce some grammar types much better than baselines, suggesting that some heads act as a proxy for constituency grammar. We also analyze how attention heads' constituency grammar inducing (CGI) ability changes after fine-tuning with two kinds of tasks, including sentence meaning similarity (SMS) tasks and natural language inference (NLI) tasks. Our results suggest that SMS tasks decrease the average CGI ability of upper layers, while NLI tasks increase it. Lastly, we investigate the connections between CGI ability and natural language understanding ability on QQP and MNLI tasks.

1 Introduction

Recently, pre-trained language models have achieved great success in many natural language processing tasks (Devlin et al., 2019; Yang et al., 2019), including sentiment analysis (Liu et al., 2019), question answering (Lan et al., 2020) and constituency parsing (Zhang et al., 2020), to name a few. Though these models have become more and more popular in many NLP tasks, they are still "black boxes". What they have learned, and why and when they perform well remain unknown. To open these "black boxes", researchers have used many methods to analyze the linguistic knowledge that these models encode (Goldberg, 2019; Clark et al., 2019; Hewitt and Manning, 2019; Kim et al., 2020).

Pre-trained language models use self-attention mechanism in each layer to compute the internal representations of each token. In this work, we investigate the hypothesis that some attention heads in pre-trained language models have learned constituency grammar. We use an unsupervised constituency parsing method to extract constituency trees from each attention heads of BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) before and after fine-tuning. This method computes the syntactic distance between every two adjacent words and generates a constituency parsing tree recursively. We analyze the extracted constituency parsing trees to investigate whether specific attention heads induce constituency grammar better than baselines, and which types of constituency grammars they learn best.

In prior work, Kim et al. (2020) show that some layers of pre-trained language models exhibit syntactic structure akin to constituency grammar to some degree. However, they do not analyze how fine-tuning affects models. We first follow their methods to extract constituency grammar from BERT and RoBERTa. Then, we use the same approach to analyze BERT and RoBERTa after fine-tuning. To the best of our knowledge, we are the first to investigate how fine-tuning affects the constituency grammar inducing (CGI) ability of attention heads. We fine-tune them on two types of GLUE natural language understanding (NLU) tasks (Williams et al., 2018; Wang et al., 2018). The first type is the sentence meaning similarity (SMS) task. We fine-tune our models on two datasets, QQP [1] and STS-B (Cer et al., 2017). The second type is the natural language inference (NLI) task. We fine-tune our models on two datasets, MNLI (Williams et al., 2018) and QNLI (Rajpurkar et al., 2016; Wang et al., 2018). Lastly, we investigate the rela-

[1]https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs

tions between CGI ability of attention heads and natural language understanding ability on QQP and MNLI tasks.

The findings of our study are as follows:

1. Attention heads in the higher layers of BERT and the middle layers of RoBERTa have better constituency grammar inducing (CGI) ability. Some heads act as a proxy for some constituency grammar types, but all heads do not appear to fully learn constituency grammar.

2. The sentence meaning similarity task decreases the average CGI ability in the higher layers. The natural language inference task increases it in the higher layers.

3. For QQP and MNLI tasks, attention heads with better CGI ability are more important for BERT. However, this relation is different in RoBERTa.

2 Related Work

Many works have proposed methods to induce constituency grammar and extract constituency trees from the attention heads of the transformer-based model. Mareček and Rosa (2018) aggregate all the attention distributions through the layers and get an attention weight matrix. They extract binary constituency tree and undirected dependency tree from this matrix. Kim et al. (2020) use the attention distribution and internal vector representation to compute Syntactic Distance (Shen et al., 2018) between every two adjacent words to draw constituency trees from raw sentences without any training.

Additionally, researchers have investigated how fine-tuning affects syntactic knowledge that BERT learns. Kovaleva et al. (2019) use the subset of GLUE tasks (Wang et al., 2018) to fine-tune BERT-base model. They find that fine-tuning does not change the self-attention patterns. They also find that after fine-tuning, the last two layers' attention heads undergo the largest changes. Htut et al. (2019) investigates whether fine-tuning affects the dependency syntax in BERT attentions. They find that fine-tuning does not have great effects on attention heads' dependency syntax inducing ability. Zhao and Bethard (2020) investigate the negation scope linguistic knowledge in BERT and RoBERTa's attention heads before and after fine-tuning. They find that after fine-tuning, the average attention heads are more sensitive to negation.

While there are some prior works analyzing attention heads in BERT, we believe we are the first to analyze the constituency grammar learned by fine-tuned BERT and RoBERTa models.

3 Methods

3.1 Transformer and BERT

Transformer (Vaswani et al., 2017) is a neural network model based on self-attention mechanism. It contains multiple layers and each layer contains multiple attention heads. Each attention head takes a sequence of input vectors $h = [h_1, ..., h_n]$ corresponding to the n tokens. An attention head will transform each vector h_i into query q_i, key k_i, and value v_i vectors. Then it computes the output o_i by a weighted sum of the value vectors.

$$a_{ij} = \frac{\exp(q_i^T k_j)}{\sum_{t=1}^{n} \exp(q_i^T k_t)} \tag{1}$$

$$o_i = \sum_{j=1}^{n} a_{ij} v_j \tag{2}$$

Attention weights distribution of each token can be viewed as the "importance" from other tokens in the sentence to the current token.

BERT is a Transformer-based pre-trained language model. It is pre-trained on BooksCorpus (Zhu et al., 2015) and English Wikipedia with masked language model (MLM) objective and next sentence prediction (NSP) objective. RoBERTa is a modified version of BERT. It removes the NSP pretraining objective and training with much larger mini-batches and learning rates. We use the uncased base size of BERT and base size of RoBERTa which have 12 layers and each layer contains 12 attention heads. Our models are downloaded from Hugging Face's Transformers Library [2] (Wolf et al., 2020).

3.2 Analysis Methods

We aim to analyze constituency grammar in attention heads. We use a method to extract constituency parsing trees from attention distributions. This method operates on the attention weight matrix $W \in (0, 1)^{T\times T}$ for every head at a given layer, where T is the number of tokens in the sentence.

[2]https://huggingface.co/models

Method: Syntactic Distance to Constituency Tree To extract complete valid constituency parsing trees from the attention weights for a given layer and head, we follow the method of Kim et al. (2020) and treat every row of the attention weight matrix as attention distribution of each token in the sentence. As in Kim et al. (2020), we compute the syntactic distance vector $\mathbf{d}= [d_1, d_2, ..., d_{n-1}]$ for a given sentence $w_1, ..., w_n$, where d_i is the syntactic distance between w_i and w_{i+1}. Each d_i is defined as follows:

$$d_i = f(g(w_i), g(w_{i+1})), \tag{3}$$

where $f(\cdot, \cdot)$ and $g(\cdot)$ are a distance measure function and feature extractor function. We use Jensen-Shannon function to measure the distance between each attention distribution. Appendix A gives a brief introduction of this function. $g(w_i)$ is equal to the i^{th} row of the attention matrix W.

To introduce the right-skewness bias for English constituency trees, we follow Kim et al. (2020) by adding a linear bias term to every d_i:

$$\hat{d}_i = d_i + \lambda \cdot Mean(\mathbf{d}) \times \left(1 - \frac{i-1}{m-1}\right), \tag{4}$$

where $m = n - 1$ and λ is set to 1.5.

After computing the syntactic distance, we use the algorithm introduced by Shen et al. (2018) to get the target constituency tree. Appendix B describes this algorithm.

Constituency parsing is a word-level task, but BERT uses byte-pair tokenization (Sennrich et al., 2016). This means that some words are tokenized into subword units. Therefore, we need to convert token-to-token attention matrix to word-to-word attention matrix. We merge the non-matching subword units and compute the means of the attention distributions for the corresponding rows and columns. We use two baselines in our experiments. They are left-branching and right-branching trees.

3.3 Experiments Setup

In our experiments, we use an unsupervised constituency parsing method to induce constituency grammar on WSJ Penn Treebank (PTB, Marcus et al. (1993)) without any training. We use the standard split of the dataset-23 for testing. We use sentence-level F1 (S-F1) score to evaluate our models. In addition, we also report label recall scores for six main phrase categories: SBAR, NP, VP, PP, ADJP, and ADVP.

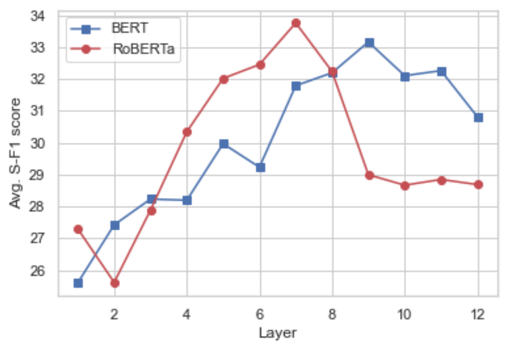

Figure 1: Average constituency parsing S-F1 score of each layer in BERT and RoBERTa.

4 Results and Analysis

4.1 Constituency Grammar in Attention Heads before Fine-tuning

In this part, our goal is to understand how constituency grammar is captured by different attention heads in BERT and RoBERTa before fine-tuning. First, we investigate the common patterns of attention heads' constituency grammar inducing (CGI) ability in BERT and RoBERTa. From Figure 1, we can find that the CGI ability of the higher layers of BERT is better than the lower layers. However, the middle layers of RoBERTa are better than the other layers. In appendix C, two heatmaps of every heads' S-F1 score in BERT and RoBERTa also show such patterns.

Table 1 describes the S-F1 scores of the best attention heads of BERT and RoBERTa. We also choose the best recall for each phrase type. We observe that the S-F1 scores of BERT and RoBERTa are only slightly better than the right-branching baseline. This implies that the attention heads in BERT and RoBERTa do not appear to fully learn constituency grammar. However, they outperform the baselines by a large margin for noun phrase (NP), preposition phrase (PP), adjective phrase (ADJP), and adverb phrase (ADVP). This implies that the attention heads in BERT and RoBERTa only learn a part of constituency grammar.

4.2 Constituency Grammar in Attention Heads after Fine-tuning

In this part, we fine-tune BERT and RoBERTa with four downstream tasks, QQP, STS-B, QNLI, and MNLI. These four tasks can be divided into two types. The first type is the sentence meaning similarity task (SMS), including QQP and STS-B. This

Models	S-F1	SBAR	NP	VP	PP	ADJP	ADVP
Baselines							
Left-branching Trees	8.73	5.46%	11.33%	0.82%	5.02%	2.46%	8.04%
Right-branching Trees	39.46	68.76%	24.89%	71.76%	42.43%	27.65%	38.11%
Pre-trained LMs							
BERT	39.47	67.32%	46.48%	68.82%	57.26%	46.39%	65.03%
BERT-QQP	39.97	67.32%	45.39%	68.79%	50.71%	45.01%	61.54%
BERT-STS-B	39.48	67.32%	44.16%	68.82%	56.68%	48.39%	57.69%
BERT-QNLI	39.74	67.32%	50.96%	68.81%	65.38%	46.08%	63.29%
BERT-MNLI	39.66	67.32%	44.89%	68.75%	62.81%	49.16%	64.69%
RoBERTa	39.60	67.43%	47.92%	69.35%	56.53%	49.00%	66.43%
RoBERTa-QQP	39.41	66.70%	43.02%	69.45%	51.06%	43.16%	60.84%
RoBERTa-STS-B	40.36	66.76%	46.82%	69.50%	54.91%	46.54%	64.34%
RoBERTa-QNLI	43.95	66.76%	52.51%	69.48%	58.30%	48.39%	69.23%
RoBERTa-MNLI	40.41	66.76%	47.97%	69.42%	57.50%	47.77%	68.88%

Table 1: Highest constituency parsing scores of all models. Blue score means that this score is lower after fine-tuning. Red score means that this score is higher after fine-tuning.

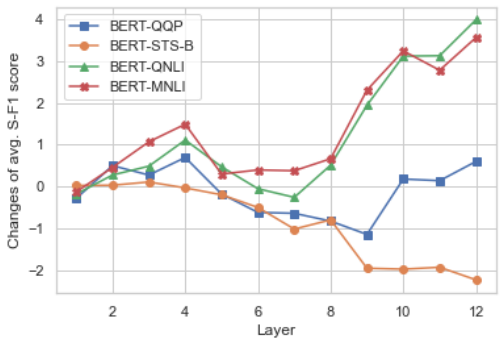

Figure 2: Changes of average S-F1 score of each layer in BERT after fine-tuning.

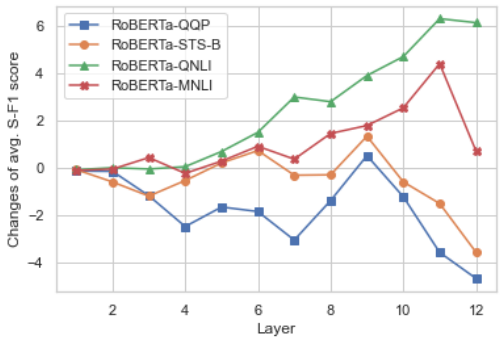

Figure 3: Changes of average S-F1 score of each layer in RoBERTa after fine-tuning.

task requires models to determine whether two sentences have the same meaning. The second type is the natural language inference task (NLI), including QNLI and MNLI. This task requires models to determine whether the first sentence can infer the second sentence. We want to analyze how these two kinds of downstream tasks affect constituency grammar inducing (CGI) ability of attention heads in BERT and RoBERTa.

Figure 2 and Figure 3 show that these four tasks do not have much influence on BERT and RoBERTa for the lower layers. For the higher layers, fine-tuning with NLI tasks can increase the average CGI ability of attention heads in BERT and RoBERTa. However, fine-tuning with SMS tasks harms it.

Table 1 shows that fine-tuning can increase the highest constituency parsing scores of all models except RoBERTa-QQP. However, fine-tuning with SMS tasks decreases the ability of attention heads to induce NP, PP, ADJP, and ADVP. For BERT, NLI tasks can increase the ability of attention heads to induce NP, PP. For RoBERTa, NLI tasks can increase the ability of attention heads to induce NP, VP, PP, and ADVP.

4.3 Constituency Grammar Inducing Ability and Natural Language Understanding Ability

In this part, we analyze the relations between constituency grammar inducing (CGI) ability and natural language understanding (NLU) ability on QQP

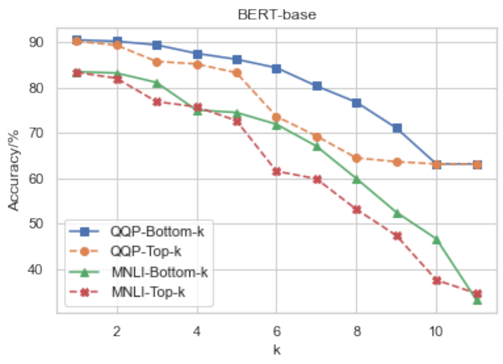

Figure 4: QQP dev and MNLI dev-matched accuracy after masking the top-k/bottom-k attention heads in each layer of BERT-QQP and BERT-MNLI.

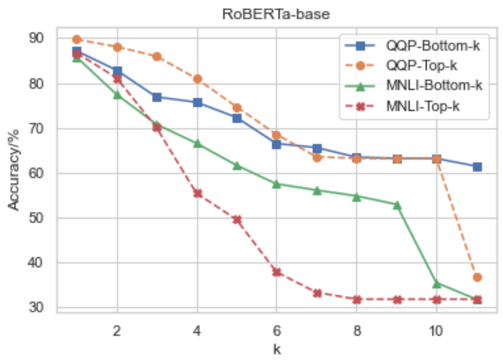

Figure 5: QQP dev and MNLI dev-matched accuracy after masking the top-k/bottom-k attention heads in each layer of RoBERTa-QQP and RoBERTa-MNLI.

and MNLI tasks. We use the performance of BERT and RoBERTa to evaluate their NLU ability. We report the scores on the validation, rather than test data, so the results are different from the original BERT paper.

First, we sort all attention heads in each layer based on their S-F1 scores before fine-tuning. Then we use the method in Michel et al. (2019) to mask the top-k/bottom-k ($k = 1, ..., 11$) attention heads in each layer and compute the accuracy on two downstream tasks, QQP and MNLI.

Figure 4 shows that downstream tasks accuracy scores decrease quicker when we have masked the top-k attention heads in BERT. Especially for the QQP task, after masking the bottom-7 attention heads in all layers, accuracy is still higher than 80%, which is more than 10% higher than masking the top-7 attention heads.

Figure 5 shows that masking RoBERTa has different results from BERT. For the QQP task, when k is smaller or equal to 6, masking the bottom-k attention heads in all layers decreases faster. For the MNLI task, when k is 1 or 2, masking the bottom-k heads decreases also faster. When k is larger than 6 in the QQP task and 2 in the MNLI task, masking the top-k heads decreases faster.

For BERT, the results show that attention heads with better CGI ability are more important for a model to gain NLU ability on these two tasks. For RoBERTa, the connections between CGI ability and NLU ability are not as strong as BERT. For the MNLI task, we still can find that better CGI ability is more important for NLU ability. However, better heads are not so important for QQP task.

5 Discussion

The experiments detailed in the previous sections point out that the attention heads in BERT and RoBERTa does not fully learn much constituency grammar knowledge. Even after fine-tuning with downstream tasks, the best constituency parsing score does not change much. Our results are similar to Htut et al. (2019). They also point out that the attention heads do not fully learn much dependency syntax. Fine-tuning does not affect these results. This raises an interesting question: do attention heads not contain syntax (constituency or dependency) information? If this is true, where does BERT encode this information? Also, is syntax information not important for BERT to understand language? Our simple experiment in §4.3 shows that the attention heads with better constituency grammar inducing ability are not important for RoBERTa on QQP task. Glavaš and Vulic (2020) also point out that leveraging explicit formalized syntactic structures provides zero to negligible impact on NLU tasks. The relations between syntax and BERT's NLU ability still need to be further analyzed.

6 Conclusion

In this work, we investigate whether the attention heads in BERT and RoBERTa have learned constituency grammar before and after fine-tuning. We use a method to extract constituency parsing trees without any training, and observe that the upper layers of BERT and the middle layers of RoBERTa show better constituency grammar ability. Certain attention heads better induce specific phrase types, but none of the heads show strong constituency grammar inducing (CGI) ability. Furthermore, we observe that fine-tuning with SMS tasks decreases the average CGI ability of upper layers, but NLI tasks can increase it. Lastly, we mask some heads based on their parsing S-F1 scores. We show that attention heads with better CGI ability are more important for BERT on QQP and MNLI tasks. For RoBERTa, better heads are not so important on QQP task.

One of the directions for future research would be to further study the relations between downstream tasks and the CGI ability in attention heads and to explain why different tasks have different effects.

Acknowledgments

This project grew out of a master course project for the Fall 2020 Uppsala University 5LN714, *Language Technology: Research and Development*. We would like to thank Sara Stymne and Ali Basirat for some great suggestions and the anonymous reviewers for their excellent feedback.

References

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. 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: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. 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 the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Goran Glavaš and Ivan Vulic. 2020. Is supervised syntactic parsing beneficial for language understanding? an empirical investigation.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Taeuk Kim, Jihun Choi, Daniel Edmiston, and Sang goo Lee. 2020. Are pre-trained language models aware of phrases? simple but strong baselines for grammar induction.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

David Mareček and Rudolf Rosa. 2018. Extracting syntactic trees from transformer encoder self-attentions. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 347–349, Brussels, Belgium. Association for Computational Linguistics.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 14014–14024. Curran Associates, Inc.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Yikang Shen, Zhouhan Lin, Athul Paul Jacob, Alessandro Sordoni, Aaron Courville, and Yoshua Bengio. 2018. Straight to the tree: Constituency parsing with neural syntactic distance. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1180, Melbourne, Australia. Association for Computational Linguistics.

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 I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 5753–5763. Curran Associates, Inc.

Yu Zhang, Houquan Zhou, and Zhenghua Li. 2020. Fast and accurate neural crf constituency parsing. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 4046–4053. International Joint Conferences on Artificial Intelligence Organization. Main track.

Yiyun Zhao and Steven Bethard. 2020. How does BERT's attention change when you fine-tune? an analysis methodology and a case study in negation scope. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4729–4747, Online. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books.

A Jensen-Shannon Distance Measure Function

Jensen-Shannon function measures the distance between two distributions. Suppose that we have two distributions P and Q, the Jensen-Shannon Distance is defined as

$$JSD(P||Q) = \left(\frac{D_{KL}(P||M) + D_{KL}(Q||M)}{2}\right)^{\frac{1}{2}}, \tag{5}$$

where $M = (P + Q)/2$ and $D_{KL}(A||B) = \sum_w A(w)\log(A(w)/B(w))$.

B Syntactic Distances to Constituency Trees Algorithm

Algorithm 1 Syntactic Distances to Constituency Trees Algorithm (Shen et al., 2018)

```
 1: $S = [w_1, w_2, ..., w_n]$ : a sentence with n words.
 2: $\mathbf{d} = [d_1, d_2, ..., d_{n-1}]$ : a sequence of distances between every two adjacent words.
 3: function TREE(S, d)
 4:     if d is empty then
 5:         node ← Leaf($S[0]$)
 6:     else
 7:         $i \leftarrow \arg\max_i(\mathbf{d})$
 8:         lchild ← TREE($S_{\leq i}, \mathbf{d}_{<i}$)
 9:         rchild ← TREE($S_{>i}, \mathbf{d}_{>i}$)
10:         node ← Node(lchild, rchild)
11:     end ifreturn node
12: end function
```

C BERT and RoBERTa Heatmaps

In this section, we present two heatmaps of S-F1 score of each heads in BERT and RoBERTa. Row represents layer and column represents head.

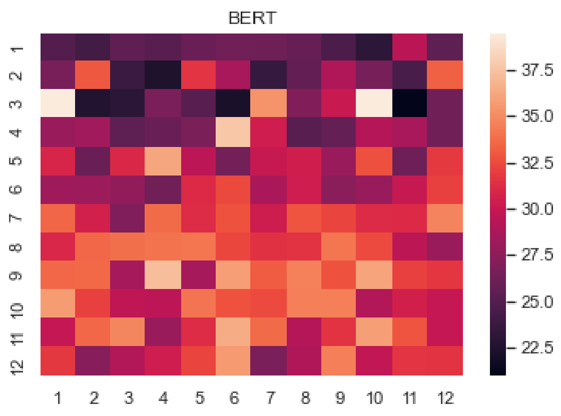

Figure 6: S-F1 score of each heads in BERT.

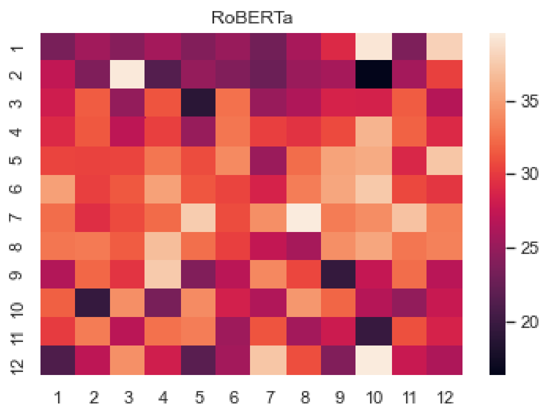

Figure 7: S-F1 score of each heads in RoBERTa.