CS447 Literature Review: BERT Explained

Hyeonjae Cho,
hc53@illinois.edu

December 9, 2022

Abstract

This review focuses on BERT and the reasons behind its exceptional performance.

1 Introduction

BERT, one of the Transformer-based natural language processing frameworks, has revolutionized the NLP landscape. The most outperforming models in the GLUE benchmark[1] are BERT variants. Table 1 shows that currently highly ranked cutting-edge models originated from BERT. They all use Transformer architecture as the basis. Therefore, it is crucial to understand the architecture of Transformer and BERT thoroughly before we tackle NLP tasks. This report reviews the concept of the Transformer, BERT, and its variant and explains how BERT learns linguistic knowledge by probing the attention mechanism and its structure.

2 Background

Exploring how deep learning-based machine translation techniques have evolved so far helps us grasp the background of popular NLP algorithms. RNN, which is the most principal model we use for NLP tasks, emerged in 1986. RNN-based models (Yin et al., 2017) defeated all the former ones and took a lead until 2017. It is well-suited for sequential data since the way it processes tokens is similar to human writing. It predicts the next word token based on the previous one. In 2015, people started to combine attention architecture (Bahdanau et al., 2014) with RNN to recover the vanishing gradient problem, which is the intrinsic drawback of the model. RNN only uses the output of the encoder in the first hidden state of its decoder. This made the model performance degrades as the length of the input sequence grows. In 2017, Google released Transformer and started a new era. The Transformer architecture would be summarized in the following section.

[1]GLUE benchmark, which stands for General Language Understanding Evaluation, provides collections of data that evaluate how well a model can handle the given tasks.

Model	MNLI-m	MNLI-mm	QQP	QNLI	SST-2	MRPC	RTE	Score
DeBERTa + CLEVER	92.1	91.8	76.5	96.7	97.6	91.1	93.2	91.1
StructBERT + CLEVER	91.7	91.5	75.6	97.4	97.7	91.9	92.5	91.0
RoBERTa	90.8	90.2	74.3	95.4	96.7	89.8	88.2	88.1

Table 1: GLUE Leaderboard Score

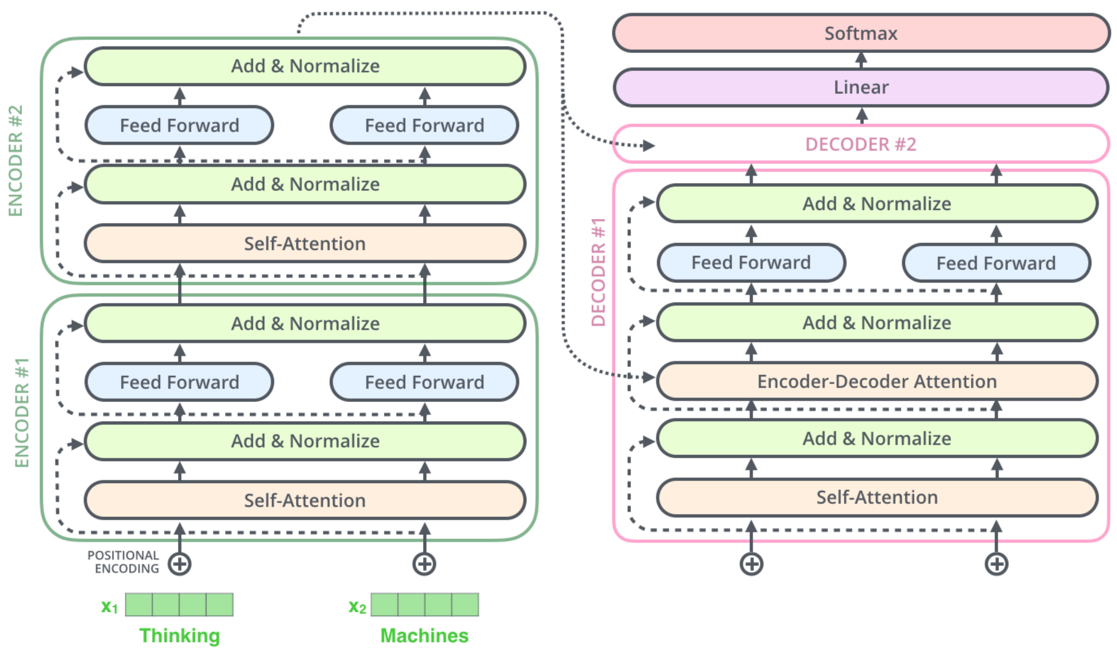

Figure 1: Transformer (from `https://jalammar.github.io/illustrated-transformer/`)

3 Transformers

The transformer (Vaswani et al., 2017) pulls RNN or CNN in a whole model architecture and solely depends on the self-attention mechanism. With the mechanism, models become even superior in dealing with long input sequences such as paragraphs or complex sentences. The Transformer also allows parallelization since it does not need to process the sequence in order, which reduces the training time cost to unprecedented rates. Considering these two strengths, several state-of-the-art models adopt Transformer architecture, subtly different from the original structure to optimize performance for the particular task. The paper elaborates on the structure of the encoder and decoder that compose the architecture and self-attention mechanism.

The Transformer model is innovative in that it takes the whole input sequence all at once. The architecture substitutes RNN with a positional embedding vector which encodes the positional information of each token. Additionally, it introduces the self-attention mechanism, which alleviates long-term dependency problems by feeding the weight vectors that contain relevance information between tokens in the given sequence. With these two techniques, the architecture carries both contextual and positional information.

Like RNN, the vector produced by the encoder of the Transformer is utilized in the decoder layers. However, as described in Figure 1, each of the hidden states uses the output vector unlike RNN and this enables all layers of the decoder to hold context data of the original sequence. In other words, the decoder takes both the decoder output in the previous step and the final hidden state of the encoder for each time step. This leads the Transformer to capture the relationships between even farther-located words by considering the non-consecutive tokens as well.

4 BERT

BERT (Devlin et al., 2019) stands for Bidirectional Encoder Representations from Transformers. As the name implies, the model does not take the input sequence in a single direction, either left-to-right or right-to-left. This language representation model is the first-ever bidirectional one. Due to the bidirectionality, BERT captures the context of a word better than the traditional models in that it can scan the neighbor words wrapping around it. This solved the limitation of unidirectional models that have fewer options for choosing pre-trained model architectures. In addition, it is pre-trained on absolutely large-scale data and can be fine-tuned by simply adding an output layer. The model shows predominant performance in numerous kinds of tasks and takes the lead in the NLP domain.

4.1 Model Architecture

According to the paper, BERT architecture shows little difference from the original Transformer architecture. Its framework consists of two steps: pre-training and fine-tuning and both steps share the same architecture except the output layer. In the pre-training step, the model trains on the unlabeled data. Then the model is initialized with pre-trained parameters in the fine-tuning step and those values are further adjusted to fit labeled data.

4.2 Pre-training Step

In the pre-training step, two main techniques are involved: 1) Masked Language Model (MLM) and 2) Next Sentence Prediction (NSP). For the masked language model, BERT randomly selects one word to replace with the [MASK] token for a certain percentage of the time. Then, as shown in Figure 2, the model has to predict the original token based on all the other words in the sequence, which can also be regarded as the context of input. Hence the MLM technique can produce a deep bi-directional pre-trained model.

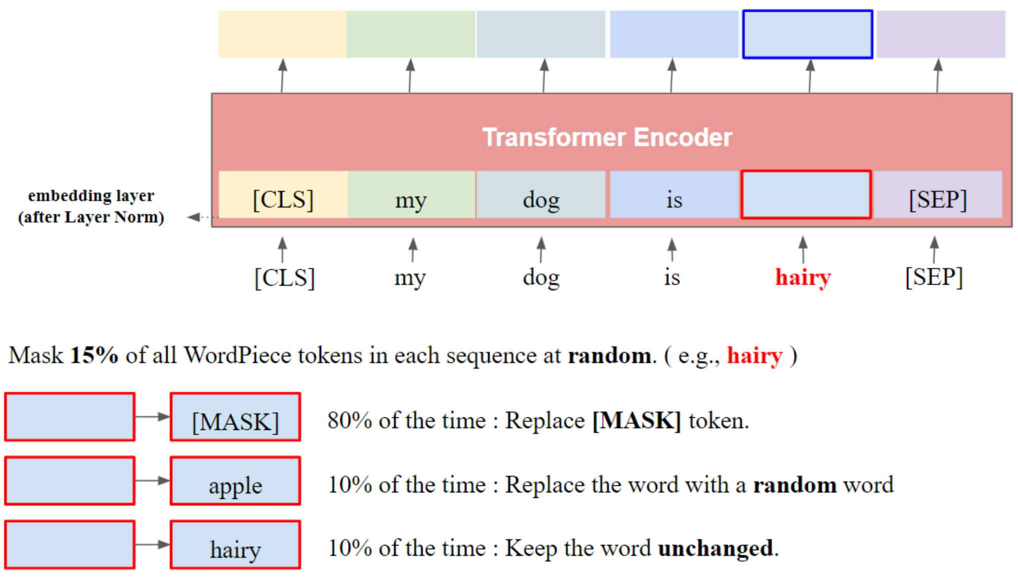
Transformer Encoder
embedding layer (after Layer Norm)
[CLS]
my
dog
is
[SEP]
[CLS]
my
dog
is
hairy
[SEP]
Mask 15% of all WordPiece tokens in each sequence at random. (e.g., hairy)
[MASK]
80% of the time : Replace [MASK] token.
apple
10% of the time : Replace the word with a random word
hairy
10% of the time : Keep the word unchanged.

Figure 2: MLM Example (from https://hryang06.github.io/nlp/BERT/)

For the Next Sentence Prediction, the model trains on the pairs of sentences and comprehends the relationships between those. In the 50% NSP learning time, collections of successive sentences are given as labeled 'IsNext' and for the rest of the time, the model learns sets of sequences labeled 'NotNext'. Consequently, BERT can handle context-heavy tasks competently. It shows an impressive performance in Question-Answering and Natural Language Inference tasks.

4.3 Fine-tuning Step

In the fine-tuning step, almost every parameter is the same except the learning rate, the number of epochs, and batch size. In this step, the hyperparameter values are fine-tuned for the downstream tasks. The literature provides a proper range for each parameter as references. BERT requires a minimal number of parameters to train additionally in fine-tuning step, which enables us to optimize them at a low time cost.

5 An Analysis of BERT's Attention

As well known, numerous models showing outstanding performance are based on BERT but why and how is not clear. The original paper (Devlin et al., 2019) also does not explicitly state the technical reason why BERT is best. The literature (Clark et al., 2019) presents methodologies to examine the attention map of BERT. The paper[2] helps us understand what kind of linguistic concepts the attention heads capture. Complementary to the study, the paper offers visualizations that enable us to get the picture of attention head behaviors. Since the attention mechanism is the main component in the Transformer and BERT, the literature plays a pivotal role in understanding two fundamental models.

5.1 Shallow Analysis of Attention

The paper discovered common patterns among multiple attention heads and categorized them into three types. First, in the earlier layers of BERT, some attention heads intensely attend to the next or previous tokens. Those heads also exist in other layers, but more are in the early stage. These heads are known as the heads attending to relative positions.

Second, the author found that the attention weights having high value in a [SEP] token do not carry significant importance. As visualized in Figure 3, when observing a particular head that treats its syntactic dependency, other tokens heavily attend to [SEP] token. To discover the reason behind this, the author used a gradient-based measure scoring feature importance by replacing tokens attended to and seeing the performance difference. As a result, changing the degree of attention to [SEP] token did not affect BERT output.

Finally, the paper calculated the average entropy on each attention distribution. In lower layers, some heads show high entropy values, which implies that those heads express broad attention. We can regard these heads as bag-of-vector representations. In the last layer, attention heads to [CLS] token also showed high entropy. The author interpreted this situation as the head collecting the whole input sequence for the NSP objective in the pre-training step.

Additionally, the literature clustered attention heads behaviors described above and the result is Figure 4. The figure also shows that the attention heads in the same layer share similar patterns.

[2]Codes are available in https://github.com/clarkkev/attention-analysis/

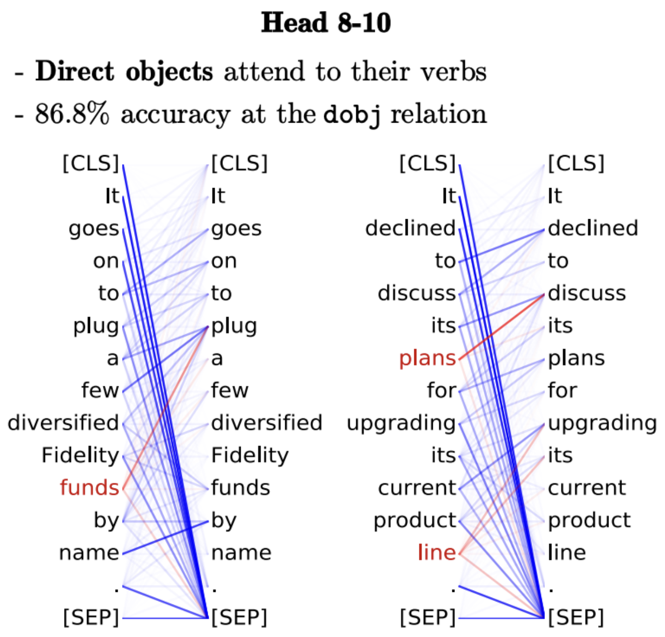

Figure 3: The Head Attending to Direct Object (from Clark et al. (2019))

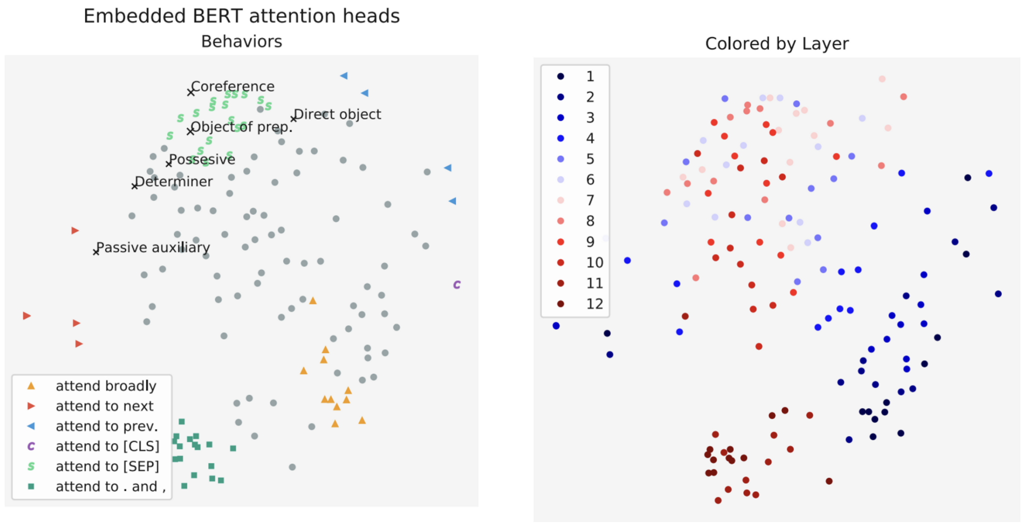

Figure 4: Attention Cluster (from Clark et al. (2019))

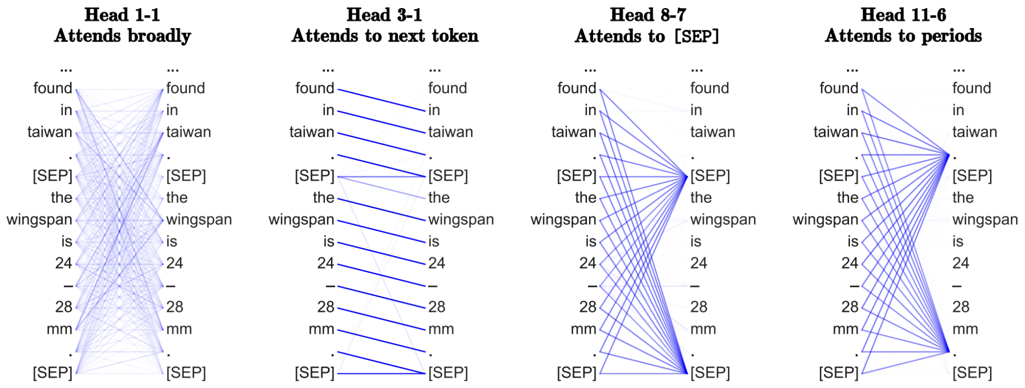

Figure 5: Specialized Attentions (from Clark et al. (2019))

5.2 Detailed Patterns of Individual Attention

To conduct a drill-down analysis, the paper evaluated attention heads using a labeled dataset and tested performance of individual attention heads. The author monitored how well they deal with dependency syntax and coreference resolution.

For dependency syntax, the author evaluated by observing two directions: from head to dependent and vice versa. Consequently, no single attention head fits broadly well with all syntax structures. On the other hand, there are specific attention heads that are syntax-aware. Figure 5 depicts those heads.

There is also a coreference resolution evaluation in the paper. The task is more complex than syntax parsing because coreference links are generally longer than syntactic ones. BERT shows decent performance - 10 points more in accuracy score than the string-matching baseline.

To summarize, BERT captures some English hierarchical syntax structures with attention heads. Considering the model is pre-trained by unlabeled data, the knowledge BERT learns plays a prominent role in its exceptional performance. However, its sensitivity to a linguistic structure is just a derivative of self-supervised training. Therefore, the inducement for such high performance is not a single specialized architecture component, but the result made by combinations of multiple attention heads. This is the rationale why numerous studies are encouraged to specify the exact way how BERT learns some linguistic knowledge.

6 A Primer in BERTology

Compared to the paper in Section 5, the paper (Rogers et al., 2020) focuses on investigating internal vector representations rather than attention mechanisms. The literature reviewed more than 150 studies and made an effort to outline all types of knowledge BERT learns. The paper clarifies the limitations of BERT and proposes future steps addressing its constraints. The report motivated subsequent studies on BERT and emerging variants.

6.1 Knowledge BERT Learns

Some studies proved that BERT understands syntactic, semantic, and world knowledge. For syntactic knowledge, word representations contain syntactic information such as Parts of Speech and syntactic structures. Also, it captures subject-predicate agreement while learning MLM objectives. However, this knowledge is incomplete, so it is hard to argue that the performance does not solely depend on its syntactic knowledge.

Compared to the previous knowledge, it is hard to analyze the semantic knowledge. A few studies have been conducted by probing classifiers and concluded that BERT takes some information about semantic roles. On the other hand, it does not represent numeric values well. In addition, the performance dramatically changes when some tokens are replaced by words that are in coreference relations. This implies that BERT does not perform nicely in named entity replacement tasks.

Based on the prior studies, the paper contends that world knowledge does not play a pivotal role in BERT reasoning. Still, it can capture the affordances and properties of objects. However, it cannot figure out the pragmatic inference and abstract properties of things.

6.2 Limitations and Future Steps

At the moment, the most promising models based on transformer architecture are charged by taking a considerable number of parameters. GPT-3 has about 175B and GPT-4 is expected to have 100 trillion parameters. To handle computational complexity and enhance reproducibility, some studies suggested pruning the parameters. They discovered the fact that the number of parameters does not guarantee to achieve higher performance. Furthermore, we still need to proceed with examinations on how BERT works and improve the model by enabling reasoning.

7 ALBERT

Generally, we can guarantee better performance by increasing the size of parameters when pre-training natural language representations. On the other hand, as described in Section 6.2, it is not always the answer to expand the model size because of the hardware memory limitations and high time cost. ALBERT (Lan et al., 2020), ‘A Lite BERT’ presents two parameter reduction techniques while preserving the performance level. The framework[3] enhances the scalability of BERT, resulting in various state-of-the-art models currently.

7.1 Factorization of Embedding Matrix

Commonly it is known that pre-trained natural language model performance increases as the model size grows. The model size is proportional to the number of layers and hidden units. In a traditional model such as BERT, the input embedding size (E) is the same as the hidden size (H). In other words, if H rises, then E also increases. The author contends that E does not need to be as long as H since E is more independent from context representation than H. Therefore, the paper proposes factorization of the embedding matrix into two small matrices. Unlike BERT, the VxH matrix consists of VxE and ExH matrices. As Figure 6 shows, the performance does not count on the size of E.

7.2 Cross-layer Parameter Sharing

ALBERT presents sharing parameters along particular collections of layers. The literature experimented with three cases: 1) complete cross-layer parameter sharing, 2) only attention parameters shared, and 3) only feed-forward network parameters shared. The baseline was a non-shared model.

[3]Codes and pre-trained models are provided in https://github.com/google-research/albert/

Model	*E*	Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base not-shared	64	87M	89.9/82.9	80.1/77.8	82.9	91.5	66.7	81.3
	128	89M	89.9/82.8	80.3/77.3	83.7	91.5	67.9	81.7
	256	93M	90.2/83.2	80.3/77.4	84.1	91.9	67.3	81.8
	768	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base all-shared	64	10M	88.7/81.4	77.5/74.8	80.8	89.4	63.5	79.0
	128	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1
	256	16M	88.8/81.5	79.1/76.3	81.5	90.3	63.4	79.6
	768	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8

Figure 6: The effect of varying embedding size (from Lan et al. (2020))

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
ALBERT base *E*=768	all-shared	31M	88.6/81.5	79.2/76.6	82.0	90.6	63.3	79.8
	shared-attention	83M	89.9/82.7	80.0/77.2	84.0	91.4	67.7	81.6
	shared-FFN	57M	89.2/82.1	78.2/75.4	81.5	90.8	62.6	79.5
	not-shared	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3
ALBERT base *E*=128	all-shared	12M	89.3/82.3	80.0/77.1	82.0	90.3	64.0	80.1
	shared-attention	64M	89.9/82.8	80.7/77.9	83.4	91.9	67.6	81.7
	shared-FFN	38M	88.9/81.6	78.6/75.6	82.3	91.7	64.4	80.2
	not-shared	89M	89.9/82.8	80.3/77.3	83.2	91.5	67.9	81.6

Figure 7: The result of different parameter-sharing strategies (from Lan et al. (2020))

As the result shown in Figure 7, only the second option showed little difference. The author did not clarify why the third option was not a good solution. From my perspective, the fact that attention heads in the same layer exhibit similar behaviors as stated in Section 5.1, implies that they are acceptable to share parameter values. On the other hand, feed-forward networks function differently from attention layers, which would be inappropriate to be set with the same values.

7.3 Sentence Order Prediction

Originally, BERT is pre-trained by executing NSP objectives. However, ALBERT substitutes NSP with Sentence Order Prediction. All the details are the same apart from data creation in the pre-training step. NSP is more related to topic prediction since the negative samples are created by selecting random sentences. SOP generates negative examples by flipping positive ones. Thus, it is more likely to train the association relationship between two sentences than NSP. In Figure 8, SOP achieves better than NSP when each algorithm serves the other's task.

	Intrinsic Tasks			Downstream Tasks					
SP tasks	MLM	NSP	SOP	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg
None	54.9	52.4	53.3	88.6/81.5	78.1/75.3	81.5	89.9	61.7	79.0
NSP	54.5	90.5	52.0	88.4/81.5	77.2/74.6	81.6	**91.1**	62.3	79.2
SOP	54.0	78.9	86.5	**89.3/82.3**	**80.0/77.1**	**82.0**	90.3	**64.0**	**80.1**

Figure 8: SOP performance on sentence prediction tasks (from Lan et al. (2020))

8 Discussion & Conclusion

Summary Through Section 2 to 4, I tried to illustrate the concept of BERT and background as much as possible. Section 2 demonstrates the former models and the motivation of the Transformer architecture. In Section 3, the foundation of BERT is presented and BERT adopted only the encoder part of the Transformer. Section 4 shows how BERT performs and the techniques it utilizes in the pre-training step. In Section 5, I scrutinized the attention mechanism of BERT with several illustrations. The section also includes information about linguistic knowledge the attention heads capture. The paper in Section 6 is broader than in Section 5, which investigated all the previous studies and rephrased them into several key points. Finally, Section 7 is one of the BERT variants and usually collaborated with other techniques in the current GLUE leaderboard. The literature shows a way to reduce the effort needed in training BERT.

Discussion Although a lot of time has passed since BERT became a fundamental framework in the NLP domain, what it learns and why it is the best are not clarified yet. This is somehow nonsense since more than hundreds of studies have been conducted to identify the vagueness of BERT. However, BERTology implies that the superiority originates from the mixture of its components in the architecture. This is the reason why we are struggling with breaking down BERT and specifying what exactly each component does.

I believe that we still need to analyze BERT at a lower level. Complementary to the paper in Section 5, more supportive studies should be done to describe the common patterns of attention heads by applying to different kinds of text data. The literature utilized various sorts of statistics to distinguish the unique pattern. Additionally, varying the way to visualize the behaviors would be influential both to the analyses of BERT and to optimization for the downstream task. These efforts would be meaningful to the emerging BERT fandom.

Conclusion These days, the size reduction of the natural language understanding model has evolved as the mainstream. To reduce the number and size of parameters, we necessarily choose which parameter effect should be diminished. To make a decision, we have to sufficiently understand the role of each parameter and the information it learns.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

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.

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 of Learning Representations*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. 2017. Comparative study of CNN and RNN for natural language processing. *CoRR*, abs/1702.01923.