Fine-tune BERT with Sparse Self-Attention Mechanism

Baiyun Cui, Yingming Li[*], Ming Chen, and Zhongfei Zhang
College of Information Science and Electronic Engineering,
Zhejiang University, China
`baiyunc@yahoo.com,{yingming,funkyblack,zhongfei}@zju.edu.cn`

Abstract

In this paper, we develop a novel Sparse Self-Attention Fine-tuning model (referred as SSAF) which integrates sparsity into self-attention mechanism to enhance the fine-tuning performance of BERT. In particular, sparsity is introduced into the self-attention by replacing softmax function with a controllable sparse transformation when fine-tuning with BERT. It enables us to learn a structurally sparse attention distribution, which leads to a more interpretable representation for the whole input. The proposed model is evaluated on sentiment analysis, question answering, and natural language inference tasks. The extensive experimental results across multiple datasets demonstrate its effectiveness and superiority to the baseline methods.

1 Introduction

Recently, the pre-trained language models obtain new state-of-the-art results on a broad range of tasks, for example ULMFiT (Howard and Ruder, 2018), ELMo (Peters et al., 2018), and OpenAI GPT (Radford et al., 2018). Devlin et al. (2018) proposed BERT, a deep bidirectional language representation model which substantially outperforms the previous methods and has attracted wide attention in natural language processing. Fine-tuning on the pre-trained BERT has shown to be beneficial to improve many downstream tasks such as document classification (Adhikari et al., 2019), extractive summarization (Liu, 2019), question answering (Yang et al., 2019), natural language inference (Liu et al., 2019), and reading comprehension (Xu et al., 2019).

To further understand the impact of BERT on the fine-tuning performance, we exploit the attention behavior of the model when fine-tuning BERT in sentiment analysis task. In particular, the attention matrices of models with different setups are visualized in Figure 1. As shown in Figure 1(b), the attention distribution of fine-tuning with BERT is close to being a band matrix, with interest concentrated on small regions near the diagonal. Compared to the model trained from scratch without pre-training (Figure 1(a)), more interpretability would be provided by fine-tuning with BERT due to its structural attention distribution. This difference in the distributions might be caused by the masked language model task of BERT, where the model is required to predict the original vocabu-

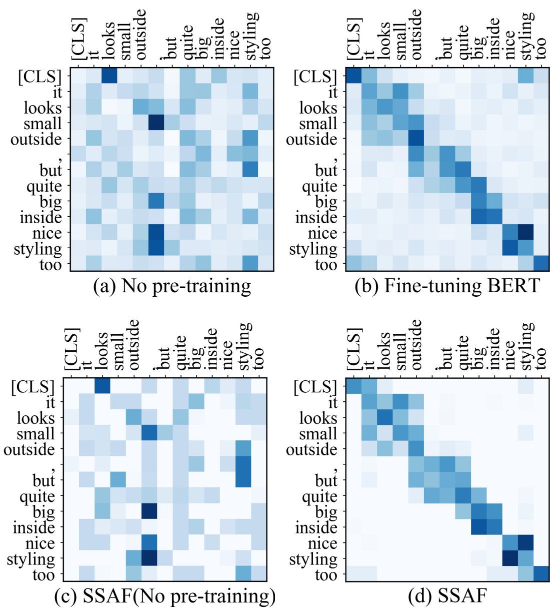

Figure 1: The explorations of attention behavior for four models: No pre-training, Fine-tuning BERT, SSAF(No pre-training), and SSAF. All the attention matrices are extracted from the eighth attention layer of the model for the sentence in Sentube-A dataset. Words on the y-axis are attending to the words on the x-axis. The attention distribution in SSAF presents more structurally sparse patterns compared to the others.

[*]Corresponding author

lary id of the masked token based on its context. Consequently, the local interactions among different words are strengthened and play more important roles in predicting the masked token. Such modeling makes the learned representation more expressive for a diversity of tasks. However, the current fine-tuning models usually employ the pre-trained BERT as initialization of the networks and do not pay enough attention on how to dynamically control this attention distribution to be structurally sparse during fine-tuning process, which might further help improve the interpretability of the framework.

To overcome the above limitation, in this work, we develop a novel Sparse Self-Attention Fine-tuning model (referred as SSAF) by integrating sparsity into self-attention to refine the attention distribution when fine-tuning the pre-trained BERT. Specifically, we propose sparse self-attention mechanism (SSAM) to induce sparsity by replacing the traditional softmax function with a sparse transformation in self-attention networks. Instead of covering all the connections built between different words as the original fine-tuning model, the sparsity is designed to promote the most essential relationships among them with higher attention weights, and meanwhile eliminate the influence from meaningless relations by truncating their probabilities to exactly 0. As presented in Figure 1(d), more sparse attention structure are obtained by SSAF through fine-tuning BERT with sparsity constraint. Even without the pre-training, the sparse self-attention networks (SSAF No pre-training) also behaves better than the traditional self attention version in respect to sparsity, which is shown in Figure 1(c). As a generic module, SSAM is flexible to be applied to neural network models in a wide variety of tasks.

Extensive experiments are conducted on three NLP tasks to investigate the performances of SSAF, which include sentiment analysis, question answering, and natural language inference. Evaluation results on seven public datasets show that the proposed approach achieves remarkable improvements over other competing models.

2 Fine-tuning with Sparse Self-Attention

In this section, we first clarify the traditional self-attention model, then present a sparse self-attention mechanism, which extends the self-attention by replacing the standard softmax function with a sparse transformation, and finally develop a sparse self-attention fine-tuning model.

2.1 Self-attention mechanism

We start by introducing self-attention mechanism, which is the foundation of Transformer encoder (Vaswani et al., 2017) as well as BERT. Self-attention networks is capable of directly relating tokens at different positions from the sequence by computing the attention score (relevance) between each pair of tokens.

Formally, given an input sequence $\mathbf{x} = (x_1, \cdots x_L)$, the output representation $\mathbf{y} = (y_1, \cdots y_L)$ is constructed by applying weighted sum of transformations of the input elements $\mathbf{x}$ based on the relevance, where the elements $\{x_i, y_i\} \in \mathbb{R}^d$. The i-th output y_i is computed as:

$$y_i = \textstyle\sum_{j=1}^{L} \alpha_{ij}(x_j W_v) \tag{1}$$

$$e_{ij} = \frac{(x_i W_q)(x_j W_k)^T}{\sqrt{d}}, \alpha_{ij} = \rho(e_{ij}) \tag{2}$$

where $W_q \in \mathbb{R}^{d\times d}$, $W_k \in \mathbb{R}^{d\times d}$, and $W_v \in \mathbb{R}^{d\times d}$ denote the trainable parameter matrices. ρ is a probability mapping function. The resulting attention weight α_{ij} represents the relevance between the i-th and j-th input element.

The classical choice for ρ is the softmax transformation, which is calculated as:

$$\rho(e_{ij}) = \text{softmax}(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{t=1}^{L}\exp(e_{it})} \tag{3}$$

However, since softmax function is strictly positive, it produces attention distribution with full support. This results in the dense dependencies between each pair of words and fails to assign exactly zero probability to less meaningful relationships. Further, putting nonzero weight on every relationship would also degrade the interpretability. With such softmax transformation, the self-attention networks does not pay more attention to those important connections while also being easily disturbed by many unrelated words.

2.2 Sparse Self-Attention Mechanism

To address this problem, we employ sparsegen-lin transformation (Laha et al., 2018) to replace softmax in Equation 3, which not only leads to a sparse probability distribution but also offers an explicit control over the degree of sparsity.

Sparsegen-lin formulation projects the attention scores $\mathbf{e}_i = (e_{i1}, e_{i2}, \cdots e_{iL})$ onto the probability simplex $\mathbf{p}_i$ and introduces coefficient $\lambda < 1$ to influence the regularization strength:

$$\rho(\mathbf{e}_i) = \underset{\mathbf{p}_i \in \Delta^{L-1}}{\operatorname{argmin}} \left\{ ||\mathbf{p}_i - \mathbf{e}_i||^2 - \lambda ||\mathbf{p}_i||^2 \right\} \quad (4)$$

where $\Delta^{L-1} := \left\{ \mathbf{p}_i \in \mathbb{R}^L | \sum_{j=1}^{L} p_{ij} = 1, \mathbf{p}_i \geq 0 \right\}$. The sparse attention distribution is then computed as follows:

$$\rho(e_{ij}) = p_{ij} = \max \left\{ 0, \frac{e_{ij} - \tau(\mathbf{e}_i)}{1 - \lambda} \right\} \quad (5)$$

where $j \in \{1, \cdots L\}$ and $\tau : \mathbb{R}^L \rightarrow \mathbb{R}$ is the threshold function.

More specifically, let $e_{i(1)} \geq e_{i(2)} \geq \cdots \geq e_{i(L)}$ be the sorted attention scores of $\mathbf{e}_i$ and $k(\mathbf{e}_i) := \max \left\{ k \in \{1, \cdots, L\} | 1 - \lambda + k e_{i(k)} > \sum_{j \leq k} e_{i(j)} \right\}$. The threshold $\tau(\mathbf{e}_i)$ is obtained as:

$$\begin{aligned} \tau(\mathbf{e}_i) &= \frac{(\sum_{j \leq k(\mathbf{e}_i)} e_{i(j)}) - 1 + \lambda}{k(\mathbf{e}_i)} \\ &= \frac{(\sum_{j \in S(\mathbf{e}_i)} e_{ij}) - 1 + \lambda}{|S(\mathbf{e}_i)|} \end{aligned} \quad (6)$$

where $S(\mathbf{e}_i)$ is the support of $\rho(\mathbf{e}_i)$, i.e., a set of the indices of nonzero coordinates. As in Equation 5, all the coordinates in the $S(\mathbf{e}_i)$ will be modified, and the others will be truncated to zero, thus providing a sparse solution. The choice of λ helps control the cardinality of the support set $S(\mathbf{e}_i)$ which influences the sparsity of attention distribution.

By introducing sparsity to refine the attention weight, our sparse self-attention mechanism (SSAM) strengthens the most important relations among different words such as local interactions, and assigns zero probability to those meaningless connections. This enables us to achieve a more expressive representation for the whole input.

2.3 Sparse Self-Attention Fine-tuning model

In this part, we propose a sparse self-attention fine-tuning model (SSAF). In particular, this fine-tuning model with BERT is composed of N sparse self-attention layers, where each layer learns a representation by taking the output from the previous layer:

$$\tilde{\mathbf{h}}^n = \mathrm{LN}(\mathbf{h}^{n-1} + \mathrm{SSAM}(\mathbf{h}^{n-1})) \quad (7)$$

$$\mathbf{h}^n = \mathrm{LN}(\tilde{\mathbf{h}}^n + \mathrm{FFN}(\tilde{\mathbf{h}}^n)) \quad (8)$$

where SSAM is adopted to replace the traditional self-attention mechanism, $\mathbf{h}^0 = \mathrm{embed}(\mathbf{x})$ denotes the representation for the input sequence $\mathbf{x}$ which is the sum of token embeddings and the position embeddings, and LN is the layer normalization operation.

Relationships with existing methods Although several sparse formulations have been developed in the literature such as sparsemax (Martins and Astudillo, 2016), fusedmax (Niculae and Blondel, 2017), and constrained sparsemax (Malaviya et al., 2018), they are mostly applied in the classification layer or in the attention-based encoder-decoder architecture. Instead, in this work, we introduce sparsity into self-attention based transformer encoder. Other than concentrating on words as the existing approaches do, we take the advantage of sparsity to identify the most essential relationships among words to capture a better sequence representation. Further, more interpretability would be obtained by our method due to the structurally sparse attention distribution.

3 Experiments

In the following sections, we empirically evaluate the effectiveness of SSAF for three NLP tasks on seven public datasets.

3.1 Datasets

Sentiment Analysis (SA) The goal of sentiment analysis is to determine the sentiment classification of a piece of text. Accuracy is used as the evaluation metric. We evaluate our model on five datasets in this task, which are listed as follows. 1) SST-1: The Stanford Sentiment Treebank (Socher et al., 2013) consists of sentences extracted from movie reviews with five classes. We follow the previous works (Kim, 2014; Gong et al., 2018) to train the model on both phrase and sentence level; 2) SST-2: This dataset is constructed from the same data with SST-1 but without the neutral reviews. We adopt the dataset version provided by GLUE (Wang et al., 2018); 3) SemEval: The SemEval 2013 Twitter dataset (Nakov et al., 2013) contains tweets with three classes: positive, negative, and neutral; 4) Sentube-A, Sentube-T: The SenTube datasets (Uryupina et al., 2014) are texts obtained from YouTube comments with two sentiment classes: positive and negative.

Question Answering (QA) This task is to predict the answer text span in the paragraph according to the question. We adopt the SQuAD v1.1 dataset (Rajpurkar et al., 2016). Since BERT$_{\text{BASE}}$ is re-

Corpus	Task	Train	Dev.	Test	Class
SST-1	SA	8.5k	1.1k	2.2k	5
SST-2	SA	67.3k	0.8k	1.8k	2
Sentube-A	SA	3.3k	0.2k	0.9k	2
Sentube-T	SA	4.9k	0.3k	1.3k	2
SemEval	SA	6.0k	0.8k	2.3k	3
SQuAD	QA	87.5k	10.1k	10.1k	-
SciTail	NLI	23.5k	1.3k	2.1k	2

Table 1: Summary of seven datasets used in our experiments. Train, Dev., and Test: The size of train, development, and test set respectively.

Models	SST-1	SST-2	SenTube-A	SenTube-T	SemEval
Ave	42.3	81.1	61.5	64.3	63.6
LSTM	46.5	82.9	57.4	63.6	67.6
BiLSTM	47.1	83.7	59.3	66.2	65.1
CNN	41.9	81.8	57.3	62.1	63.5
SSAN	48.6	85.3	62.5	68.4	72.2
Np	50.1	85.2	66.8	69.6	70.0
SSAF(Np)	50.7	86.4	68.1	70.3	71.2
BERT$_{\text{BASE}}$	55.2	93.5	70.3	73.3	76.2
SSAF	**56.2**	**94.7**	**72.4**	**75.0**	**77.3**

Table 2: Experimental results of classification accuracy for different methods with five datasets on sentiment analysis task.

ported on the development set of SQuAD in Devlin et al. (2018), we follow them to evaluate our model on the same set. The Exact Match (EM) and F1 are two evaluation metrics.

Natural Language Inference (NLI) The task involves assessing if two sentences entail or contradict each other. We use SciTail dataset (Khot et al., 2018) which is derived from a science question answering (SciQ) dataset. The evaluation metric is accuracy.

Further statistics about datasets are illustrated in Table 1.

3.2 Implementation details

We adopt the pre-trained BERT$_{\text{BASE}}$ [1] as the basis for our experiments. We choose BERT$_{\text{BASE}}$ in our work rather than another larger pre-trained model BERT$_{\text{LARGE}}$ due to the resource limitations and computation cost. The coefficient λ which controls the sparisty in Equation 4 is set to -3 in SST-1 and SemEval, -4 in SST-2 and SenTube-T, -6 in SenTube-A and SciTail, and -7 in SQuAD. We investigate the influence of different λ settings in the experiment analysis part. We adopt Adam as our optimizer with a learning rate of 2e-5 and batch size is 16. The maximum number of training epoch is 2 for SQuAD, 3 for SciTail, 4 for SST-1 and SST-2, 5 for SenTube-A, 6 for SenTube-T, and 8 for SemEval.

3.3 Baselines

To demonstrate that SSAF truly improves the fine-tuning performance, we compare SSAF against BERT$_{\text{BASE}}$ from (Devlin et al., 2018) on all the tasks. In particular, for sentiment analysis, we follow (Ambartsoumian and Popowich, 2018) to make comparisons with the following representative methods: Ave, LSTM, BiLSTM, and CNN from (Barnes et al., 2017); SSAN from (Ambartsoumian and Popowich, 2018). We report the results of these five models with Google 300-dimensional word2vec embeddings [2] on all datasets. For SST-1 and SST-2, we reproduce the compared methods according to the different dataset version.

For thorough comparison, besides the approaches proposed in the existing literature, we further implement another two models to verify the ability of sparse self-attention mechanism:

No pre-training (Np): It has the same architecture as BERT$_{\text{BASE}}$ but without pre-training.

SSAF(Np): We train SSAF from scratch only with the sparse self-attention module.

3.4 Results

The experimental results on sentiment analysis are reported in Table 2, and the performances on question answering and natural language inference are summarized in Table 3. Results show that SSAF achieves the best performance across all three tasks with a significant improvement over the previous approaches.

Compared with other competing methods in the experiments, BERT$_{\text{BASE}}$ provides a strong baseline owing to the pre-training procedure. However, this model still suffers from its inefficiency by taking all the relationships between each pair of words into consideration when constructing the representation. Our model outperforms BERT$_{\text{BASE}}$ by a stable margin especially in SST-2, SenTube-A, SenTube-T, and SciTail, with the improvements of 1.2%, 2.1%, 1.7%, and 1.5%,

[1]https://github.com/google-research/bert

[2]https://code.google.com/archive/p/word2vec/

Models	SQuAD		SciTail
	EM	F1	Accuracy
Np	65.3	74.1	78.6
SSAF(Np)	66.2	74.6	78.9
$\text{BERT}_{\text{BASE}}$	80.8	88.5	92.0
SSAF	**81.6**	**88.8**	**93.5**

Table 3: Experimental results on question answering and natural language inference tasks.

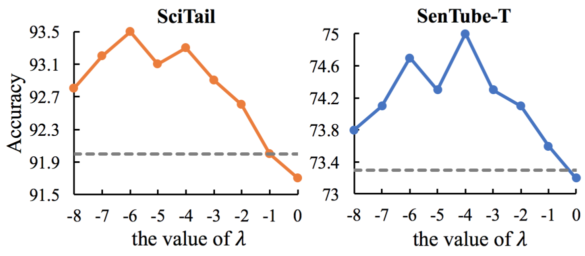

Figure 2: Experimental results on varying the coefficient λ for SSAF in SciTail and SenTube-T datasets. Grey dash line denotes the result of $\text{BERT}_{\text{BASE}}$.

respectively. Meanwhile, even without the pre-training, SSAF(No pre-training) still surpasses No pre-training across the board which clearly proves the effectiveness and universality of incorporating sparsity into self-attention model. Combining the strength of sparsity and the pre-trained language model, SSAF stands out in performance with other baselines.

Moreover, we study how the value of coefficient λ in SSAF affects the fine-tuning performance. We explore different settings from -8 to 0. Figure 2 shows the comparison results on the test set of SciTail and SenTube-T. Results summarize that $\lambda = -6$ is superior to other settings for SciTail and the performance on SenTube-T reaches the best when λ is set to -4. According to the experimental results above, we could found that on both of the datasets, the values from -7 to -3 can reach comparable performance while both larger and smaller values are more likely to decrease the accuracy to some extent though they still outperform previous work by a large margin. To understand why different values of λ have such different effects, we visualize the attention matrices extracted from SSAF with a series of λ and also visualize the one from $\text{BERT}_{\text{BASE}}$ for comparison. As shown in Figure 3, the different structural patterns among these five sub-figures lie in the different strength of the sparsity constraint. It shows that too large coefficient λ makes the attention distribution of SSAF extremely sparse, which may even overlooks some important interactions between different words. With the decrease of λ , the influence from other unnecessary connections increases, which reduces the interpretability of the model. Thus, it is important to have appropriate parameter to encourage a desirably sparse attention distribution while achieving competing accuracy.

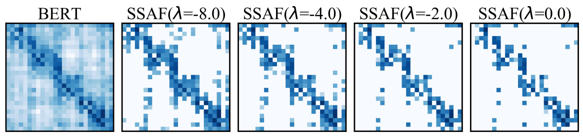

Figure 3: Visualization of attention matrices for $\text{BERT}_{\text{BASE}}$ and SSAF with different values of λ on SenTube-T dataset. With higher λ, the attention distribution in SSAF becomes sparser.

4 Conclusion

In this paper, we develop a novel Sparse Self-Attention Fine-tuning model (referred as SSAF) which integrates sparsity into self-attention mechanism to enhance the fine-tuning performance of BERT. We conduct extensive experiments on sentiment analysis, question answering, and natural language inference tasks with seven public datasets. The proposed approach substantially improves the performance over the strong baseline methods, demonstrating its effectiveness and universality while achieving higher interpretability for the framework.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China (No. 61702448, 61672456), Zhejiang Lab (2018EC0ZX01-2), the Fundamental Research Funds for the Central Universities in China (No. 2017FZA5007, 2019FZA5005), the Key Program of Zhejiang Province, China (No. 2015C01027), Artificial Intelligence Research Foundation of Baidu Inc., the funding from HIKVision and Horizon Robotics, and ZJU Converging Media Computing Lab. We thank all reviewers for their valuable comments.

References

Ashutosh Adhikari, Achyudh Ram, Raphael Tang, and Jimmy Lin. 2019. Docbert: Bert for document classification. *arXiv preprint arXiv:1904.08398*.

Artaches Ambartsoumian and Fred Popowich. 2018. Self-attention: A better building block for sentiment analysis neural network classifiers. *arXiv preprint arXiv:1812.07860*.

Jeremy Barnes, Roman Klinger, and Sabine Schulte im Walde. 2017. Assessing state-of-the-art sentiment models on state-of-the-art sentiment datasets. *arXiv preprint arXiv:1709.04219*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jingjing Gong, Xipeng Qiu, Shaojing Wang, and Xuanjing Huang. 2018. Information aggregation via dynamic routing for sequence encoding. *arXiv preprint arXiv:1806.01501*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL*, pages 328–339.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *AAAI*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Anirban Laha, Saneem Ahmed Chemmengath, Priyanka Agrawal, Mitesh Khapra, Karthik Sankaranarayanan, and Harish G Ramaswamy. 2018. On controllable sparse alternatives to softmax. In *NeurIPS*, pages 6422–6432.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*.

Yang Liu. 2019. Fine-tune BERT for extractive summarization. *arXiv preprint arXiv:1903.10318*.

Chaitanya Malaviya, Pedro Ferreira, and André F. T. Martins. 2018. Sparse and constrained attention for neural machine translation. In *ACL*, pages 370–376.

Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *ICML*, pages 1614–1623.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. In *Proceedings of the 7th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT*, pages 312–320.

Vlad Niculae and Mathieu Blondel. 2017. A regularized framework for sparse and structured neural attention. In *NeurIPS*, pages 3338–3348.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*, pages 2227–2237.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *EMNLP*, pages 2383–2392.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642.

Olga Uryupina, Barbara Plank, Aliaksei Severyn, Agata Rotondi, and Alessandro Moschitti. 2014. Sentube: A corpus for sentiment analysis on youtube social media. In *LREC*, pages 4244–4249.

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 *NeurIPS*, pages 5998–6008.

Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Hu Xu, Bing Liu, Lei Shu, and Philip S Yu. 2019. Bert post-training for review reading comprehension and aspect-based sentiment analysis. *arXiv preprint arXiv:1904.02232*.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with bertserini. *arXiv preprint arXiv:1902.01718*.