A High Performance Compression Approach for Transformer-Based NLP Tasks

Siduo Jiang
siduojiang@berkeley.edu
UC Berkeley

Cristopher Benge
cris.benge@berkeley.edu
UC Berkeley

Andrew Fogarty
apfogarty@berkeley.edu
UC Berkeley

William Casey King
caseyking@berkeley.edu
UC Berkeley

Alberto Todeschini
todeschini@berkeley.edu
UC Berkeley

Hossein Vahabi
puyavahabi@berkeley.edu
UC Berkeley

Abstract

We present a new BERT compression technique for NLP tasks that significantly reduces the number of parameters used by the model. Our compression method uses information from the hidden state activations of each BERT transformer layer, which is discarded during typical BERT inference. We achieve same accuracy as BERT across a wide variety of GLUE benchmark tasks and SQuAD 2.0 with 209x times less parameters needed. This suggests that the method may be extensible to a wider range of NLP tasks.

1 Introduction

BERT has significantly advanced the state-of-the-art across many Natural Language Processing (NLP) tasks. Given a predefined accuracy or same level accuracy a BERT with fewer parameters can be easily transferred and embedded in small devices and can be suitable for real-time application due to lower inference time. In this work we will present a new compression technique for BERT.

Existing compression methods [19, 20, 22, 29, 37, 38, 41, 54] are largely guided by altering BERT's architecture, either through the training of a new, smaller network, or by replacing or altering its modules. These efforts, dominated by distillation methods, have indeed achieved BERT-like performance while using significantly reduced model size, i.e. models with less parameters. Distillation architectures derive their power from the clever use of student-teacher networks. For example, in the case of MobileBERT [41], a custom Inverted-Bottleneck BERT architecture (the teacher) is first pre-trained using standard masked language modeling, and next sentence prediction. The key here is that two types of data flow exist, intra-block, wider feature maps such as embeddings, that feed in multi-headed self-attention, and inter-block, thinner feature maps such as the output of layer norm that feed into linear layers. The knowledge distillation step comes next, which is a layer-by-layer transfer of knowledge from BERT to MobileBERT [41], a much smaller network consisting of the same encoder blocks, but much fewer parameters. MobileBERT [41] is just as deep as BERT, but significantly thinner, i.e. each block has much fewer parameters.

In this paper we present a new BERT compression technique by taking advantage of BERT's hidden states. Our methodology is as follows. We start by performing the standard fine-tuning of BERT to a downstream task. Then, we train a much smaller model on top of the BERT's hidden states through a linear pooling layer. This layer combines and compresses the information present in the hidden state activations, which are normally discarded during inference. Our model achieve the same accuracy of BERT on a wide variety of GLUE benchmark tasks and SQuAD 2.0 with 209x times less parameters needed. When compared to the state of the art compression methodologies, we have 3x times less parameters than any other existing work while keeping the accuracy as high as BERT.

2 Related Work

Recently many BERT compression techniques have been presented [19, 41]. Compression methods, are obtained through four general approaches today: (1) distillation, (2) quantization, (3) pruning, and (4) module replacing. Knowledge distillation derive their power from the clever use of student-teacher networks where, in the case of MobileBERT [41], a custom BERT$_{large}$ architecture (the teacher) is used to train the MobileBERT [41] over the whole duration of pre-training. In general, distillation methods generally amount to novel loss functions injected at the embedding, self-attention, and hidden state layers and are found during pre-training as well as fine-tuning. DistilBERT[37] and MobileBERT[41], for instance, use distillation methods during pre-training [37], while other efforts like TinyBERT [20] incorporate distillation during pre-training and fine-tuning [20, 40]. Quantization methods aim to lower the floating point precision of the millions of BERT's parameters in order to reduce its

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM. . . $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

memory and compute footprint [38, 54]. Pruning, or the systematic exclusion of weights and layers yielding sub-networks, is a form of model compression [7, 14, 16]. [22, 29] shows that self-attention heads and entire encoder layers can be disabled without suffering substantive drops in performance. Among more recent innovations, researchers have compressed BERT by replacing its large modules with more compressed substitutes. SqueezeBERT[19], for example, finds that the fully-connected layers attached to self-attention and between each encoder block can be efficiently replaced by grouped-convolutions and convolutions, respectively. Separately, BERT-of-Theseus [53] sequentially substitutes BERT's modules with modules containing fewer parameters that are learned in a manner similar to knowledge distillation. Finally, Module addition is another mechanism for achieving parameter efficiency during fine-tuning [17].

3 Solution

BERT-Vision's architecture is centrally comprised of a pooling module (see Algorithm 1 and Figure 1 below); a method that applies the same linear function to each BERT layer, yielding a learned linear combination of BERT's hidden states.[1] The consequence of this modeling decision is that it learns from the information found across each layer [2, 44] while compressing the number of layers of information down to one. Our algorithm is simple to implement, runs on BERT itself, and reduces the number of parameters involved in forward and backward propagation.

Algorithm 1: BERT-Vision: Adapted for SQuAD

input : Hidden-state activations shaped: $(bs, inp, emb, depth)$

`LinearPooling` $(inp * bs, depth, emb)$
`GELU` $(inp * bs, depth, inp)$
reshape: $(bs, inp, inp, depth)$
`Linear` $(bs, inp, inp * depth)$
Return: span_{start} $(bs, inp, depth = 1)$
Return: span_{end} $(bs, inp, depth = 1)$

Our algorithm also has one notable drawback – it requires BERT to be partially fine-tuned. This is in part necessary as BERT-Vision is only capable of extracting and pooling insights across BERT's layers, in-so-far as it is able to generate them relative to the task at hand. In future work, we intend to study ways in which we can minimize the training costs associated with BERT-Vision such that it is an operation run in *parallel* with BERT vice an operation ran *sequential* to BERT. In comparison to state-of-the-art methods, this drawback is absent as these models have created entirely new architectures that supplant BERT rather than improve BERT.

[1] In the case of BERT_{base}, there are 13 layers, while in the case of BERT_{large}, there are 25.

4 Experiments

In this section we describe our data, experimental setup, and model training strategy. Our experiments were performed on two Microsoft Azure data science virtual machine with 112 GiB on-board ram and an NVIDIA Tesla TITAN series V100 Tensor Core GPU capable of 7 TFLOPS double-precision and tensor performance of 112 TFLOPS and possessed 16 GiB on-board RAM. We evaluated the effectiveness and efficiency of BERT-Vision on two industry benchmark data sets: The General Language Understanding Evaluation [48] (GLUE) benchmark, and the Stanford Question Answering data set(SQuAD) v2.0 [34]. GLUE consists of two single-sentence tasks CoLA and SST-2, three sentence similarity tasks MRPC, STS-B, and QQP, and four natural language inference tasks MNLI, QNLI, RTE, and WNLI. WNLI is known to be a problematic data set for BERT and was excluded from the original study [12]. We do the same here, and instead, replace it with SQuAD 2.0, a reading comprehension task consisting of questions where the answer to every question is a segment of text. As compared to SQuAD 1.1, SQuAD 2.0 contains unanswerable questions.

4.1 Experimental Pipeline

Our training pipeline follows four general steps. First, we set the hyperparameters of our BERT_{base} and BERT_{large} models to settings commonly used by academia and practitioners [2]. Second, we tuned BERT to the task at hand leveraging *hyperopt* and 100 trials to ensure the baseline was performing as best it could; maximizing its performance against the metric of interest on the development data set. Third, with optimized hyperparameters, we fine-tuned BERT against the current task for one epoch and then wrote the full embeddings with shape (layers, batch size, tokens, features) for the entire data set to disk. Fourth, we then tuned BERT-Vision using *hyperopt* and the same number of trials to optimize its hyperparameters against the metric of interest on the development data set using the emitted embeddings from the previous step. Given tuned models and emitted embeddings, we then proceeded with our comparative analysis experiments.

We compared the performance of our fully-tuned BERT-Vision model to the fully-tuned BERT model on each data set for one epoch each across the GLUE and SQuAD tasks, using the canonical metric of interest on the development set. While our chief comparison of interest is against BERT, we limited the scope of this comparison to models generally based on BERT_{base}, while also including the state-of-the-art (MobileBERT) and a highly related module addition model, AdapterBERT. Against non-BERT models, we focused on both the compression rate, as well as performance.

[2] github.com/pytorch/fairseq/blob/master/examples/roberta/README.glue.md

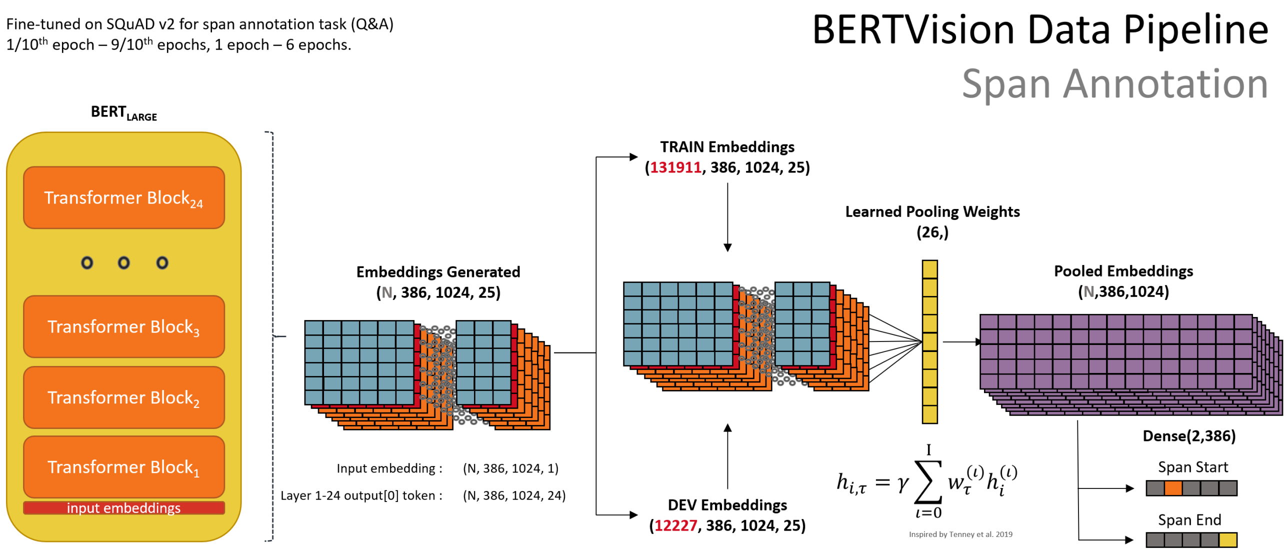

Figure 1. BERTVision span annotation data pipeline

5 Results

In this section, we present our results comparing BERT-Vision against BERT across the GLUE and SQuAD data sets.

In GLUE case, we compare BERT-Vision against BERT$_{base}$ and BERT$_{large}$ on the GLUE benchmark's development data sets. While there are many other state-of-the-art post-BERT models, our primary research interest and direct comparison is against BERT itself. In a forthcoming section, we draw comparisons between BERT-Vision and other state-of-the-art models that make use of varied compression techniques. The table below shows that BERT-Vision is competitive with BERT$_{base}$ and BERT$_{large}$ on the GLUE benchmark, judging from our overall GLUE score of 0.810 that beats BERT$_{base}$ by 0.001. Further, BERT-Vision$_{base}$ is 209x smaller than BERT$_{base}$ and n times faster. In contrast, BERT-Vision$_{large}$ falls substantially short in comparison with BERT$_{large}$, which may be owed to BERT-Vision's current architecture limitations which only allows layer pooling, thereby complicating its ability to extract compressed regularity from 25 layers vice 13. Furthermore, we find that BERT-Vision tends to perform better on smaller data sets such as RTE. This is perhaps due to the fact that more epochs is required to fully fine-tune BERT, a complication that is overcome by BERT-Vision.

In a similar fashion, we compare BERT-Vision against BERT$_{base}$ and BERT$_{large}$ on the SQuAD 2.0 benchmark's development data set. Our results below show that BERT-Vision$_{base}$ is competitive with BERT$_{base}$, according to the exact and F1 metrics below, as our model beats BERT$_{base}$ by roughly half a point and a whole point respectively. With less stark of a difference, BERT-Vision$_{large}$ falls marginally short of BERT $_{large}$'s performance. The precise reason why BERT-Vision$_{large}$ performs much closer to BERT $_{large}$ in span annotation tasks as compared to the varied tasks represented by GLUE is unclear. However, we hypothesize that slight architectural differences accounted for in the algorithm pseudocode above is perhaps responsible and is worthy of future research.

5.1 Model Analysis

BERT-Vision outperforms BERT in some tasks but not in others – but in which tasks and why is it scoring the way it does? In this section, we conduct a comparative error analysis across two GLUE data sets: MRPC and RTE. We chose these data sets for three reasons: First, MRPC and RTE differ in their central task. The former represents sentence-pair similarity while the latter represents natural language inference. Second, these two data sets also differ substantially in the extent to which BERT can learn from the data as it is much easier to get a high evaluation score in MRPC than it is to get the same in RTE. Third, BERT-Vision$_{base}$'s performance against BERT$_{base}$ varies across the two tasks. On MRPC, BERT$_{base}$ outperforms BERT-Vision$_{base}$ while on RTE, BERT-Vision$_{base}$ outperforms BERT$_{base}$. Taken collectively, these two data sets provide ideal inroads into better understanding the shortcomings and differences between both models. Further, analyzing these two data sets may tell us more about our architectural choices and why it is that, through compression, BERT-Vision$_{base}$ is able to beat the full expressive power of BERT$_{base}$ on a challenging data set.

For the RTE data set, BERT-base predicts 67.5% of the examples to be positive, with a recall of 0.787. BERT-Vision on the other hand predicts 44.0% of the examples to be positive, with a recall of 0.568. Overall, the improvement in accuracy

Model	MNLI	QNLI	QQP	RTE	SST	MSR	CoLA	STS-B	GLUE Average	SQuAD-EM
BERT-base	.823	.902	.896	.639	.920	.820	.534	.874	.802	.694
AdapterPooler-base	.822	**.903**	.886	**.726**	**.927**	**.840**	**.600**	.862	**.819**	**.701**
BERT-large	**.852**	.907	.896	.531	**.929**	.764	.207	.862	.744	.776
AdapterPooler-large	.849	**.910**	**.897**	.592	**.929**	.837	.432	**.880**	.791	.769

Table 1. Model Performance: BERT (base/large) vs. BERT-Vision (AP)

Research	Compression	Performance	Base Model	Evaluation
BERT-base [12]	1x	100%	BERT-12	GLUE, SQuAD
BERTVision	**209x**	**100%**	**BERT-12**	**GLUE, SQuAD**
BERT-48 [55]	62x	87%	BERT-12	MNLI, MRPC, SST-2
BERT-192 [55]	5.7x	93%	BERT-12	MNLI, MRPC, SST-2
SqueezeBERT [19]	2.3x	96%	BERT-12	GLUE
MobileBERT [41]	4.3x	100%	BERT-24	GLUE, SQuAD
AdapterBERT [17]	0.8x	99%	BERT-24	GLUE, SQuAD

Table 2. Comparative Performance Analysis

with BERT-Vision comes from far superior performance in recognizing negative examples (non-entailment). BERT-base predicts correctly only 45.0% of negative examples, while BERT-Vision correctly predicts 70.2% of negative examples as such. For the correctly predicted examples examples, the average length for BERT-base and BERT-Vision were similar for both sentence 1 and sentence 2, averaging about 42 words for sentence 1, and 8 words for sentence 2. These are very close to the global mean length for all examples, indicating that neither model is performing better or worse based on length alone. The overlap of correctly predicted examples is 45.5%, indicating that less than half of the examples both models correctly predicted. This leads to a potential area of improvement for BERT-Vision, which correctly predicting more of the examples BERT gets correctly.

For the MSR data set, BERT-base predicts 69.1% of the examples to be positive, with a recall of 0.873. BERT-Vision on the other hand predicts 70.7% of the examples to be positive, with a recall of 0.877, a minor difference. The improvement in accuracy with BERT-case comes from this and a slightly improved performance in recognizing negative examples. BERT-base predicts correctly only 67.0% of negative examples, while BERT-Vision correctly predicts 63.9% of negative examples as such. For the correctly predicted examples examples, the average length for BERT-base and BERT-Vision were similar for both sentence 1 and sentence 2, averaging about 19 words for sentence 1, and 19 words for sentence 2. These are very close to the global mean length for all examples, again indicating that neither model is performing better or worse based on length alone. The overlap of correctly predicted examples is 71.5%, a much larger fraction than RTE. A comparison of performance between models can be seen in table [2].

Table 2 reports the comparison to other high-performing models, we find that BERTVision is competitive, judging from its parameter size reduction and on-average performance across tasks.

6 Conclusion and Future Work

In this paper, we introduce a new method that compresses the hidden state activations emitted by all encoder layers of BERT during fine-tuning and extracts useful information typically disregarded by researchers and end-users during inference. Extensive experiments show that BERTVision is a parameter efficient approach that exceeds or achieves BERT performance across a wide range of GLUE and SQuAD 2.0 tasks, including question-answering, sentence similarity, and natural language inference tasks. It also shows that researchers should pay more attention to the ways in which the fine-tuning process may be best optimized rather than re-engineering pretraining regimens. In future work, we intend to study ways in which we can minimize the training costs associated with BERTVision such that it is an operation run in *parallel* with BERT vice an operation ran *sequential* to BERT.

References

[1] Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A Gers. 2019. How does bert answer questions? a layer-wise analysis of transformer representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1823–1832.

[2] Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How does BERT answer questions? *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*.

[3] Roy Bar-Haim, Ido Dagan, Iddo Greental, Idan Szpektor, and Moshe Friedman. 2007. Semantic inference at the lexical-syntactic level for textual entailment recognition. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 131–136.

[4] Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, Medea Lo Leggio, and Bernardo Magnini. 2009. Considering discourse references in textual entailment annotation. In *Proceedings of the 5th International Conference on Generative Approaches to the Lexicon (GL 2009)*.

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. 2018. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*.

[6] Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

[7] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained bert networks. *arXiv preprint arXiv:2007.12223*.

[8] Zhiyu Chen, Mohamed Trabelsi, Jeff Heflin, Yinan Xu, and Brian D. Davison. 2020. Table search using a deep contextualized language model. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[9] François Chollet. 2016. Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357.

[10] Alexis Conneau, Holger Schwenk, Yann Le Cun, and Löc Barrault. 2017. Very deep convolutional networks for text classification. In *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 - Proceedings of Conference*.

[11] Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

[12] Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*.

[13] William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

[14] Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

[15] Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9.

[16] Song Han, Jeff Pool, John Tran, and William J Dally. 2015. Learning both weights and connections for efficient neural networks. *arXiv preprint arXiv:1506.02626*.

[17] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

[18] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. *CoRR*, abs/1902.00751.

[19] Forrest N Iandola, Albert E Shaw, Ravi Krishna, and Kurt W Keutzer. 2020. Squeezebert: What can computer vision teach nlp about efficient neural networks? *arXiv preprint arXiv:2006.11316*.

[20] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

[21] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-Aware neural language models. In *30th AAAI Conference on Artificial Intelligence, AAAI 2016*.

[22] Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of bert. *arXiv preprint arXiv:1908.08593*.

[23] Alex R. Kuefler. 2016. Merging Recurrence and Inception-Like Convolution for Sentiment Analysis.

[24] Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*. Citeseer.

[25] Girish Limaye, Manish Pandit, and Sawal Vinay. 2019. BertNet: Combining BERT language representation with Attention and CNN for Reading Comprehension.

[26] Min Lin, Qiang Chen, and Shuicheng Yan. 2014. Network in network. In *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*.

[27] Xiaodong Liu, Wei Li, Yuwei Fang, Aerin Kim, Kevin Duh, and Jianfeng Gao. 2018. Stochastic answer networks for SQuAD 2.0. *CoRR*, abs/1809.09194.

[28] Xiaofei Ma, Zhiguo Wang, Patrick Ng, Ramesh Nallapati, and Bing Xiang. 2019. Universal text representation from BERT: An empirical study.

[29] Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *arXiv preprint arXiv:1905.10650*.

[30] Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. pages 311–318.

[31] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations.

[32] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

[33] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. *CoRR*, abs/1806.03822.

[34] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

[35] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuad: 100,000+ questions for machine comprehension of text. In *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*.

[36] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. 2019. Stand-alone self-attention in vision models.

[37] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.

[38] Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2020. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings*

of the AAAI Conference on Artificial Intelligence, volume 34, pages 8815–8821.

[39] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

[40] Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. 2019. Patient knowledge distillation for bert model compression. *arXiv preprint arXiv:1908.09355*.

[41] Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. 2020. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

[42] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2014. Going deeper with convolutions. *CoRR*, abs/1409.4842.

[43] Danny Takeuchi and Kevin Tran. 2019. Improving SQUAD 2.0 Performance using BERT + X.

[44] Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

[45] Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

[46] Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? Probing for sentence structure in contextualized word representations. *CoRR*, abs/1905.06316.

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

[48] Alex Wang, Amanpreet 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*.

[49] Alex Warstadt and Samuel R. Bowman. 2020. Linguistic analysis of pretrained sentence encoders with acceptability judgments.

[50] Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. The multi-genre nli corpus.

[51] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

[52] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

[53] Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020. Bert-of-theseus: Compressing bert by progressive module replacing. *arXiv preprint arXiv:2002.02925*.

[54] Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. *arXiv preprint arXiv:1910.06188*.

[55] Sanqiang Zhao, Raghav Gupta, Yang Song, and Denny Zhou. 2019. Extreme language model compression with optimal subwords and shared projections. *arXiv preprint arXiv:1909.11687*.

[56] Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tie-Yan Liu. 2020. Incorporating BERT into neural machine translation. *ArXiv*, abs/2002.06823.