Improving BERT computational efficiency

Stanford CS224N Default Project

Julio Oscanoa
Department of Bioengineering
Stanford University
name@stanford.edu

Abstract

In Natural Language Processing (NLP), Bidirectional Encoder Representations from Transformers (BERT) is the usual starting point to build language models. However, the increasing size of the models is increasing the computational demands of training and inference procedures. Herein, a type of model compression techniques, denoted as pruning, is explored to assess the potential to reduce computational cost of BERT-based networks. Two types of pruning (unstructured and unstructured) and three downstream tasks (sentiment analysis, paraphrase detection, and similarity prediction) are considered. Experiments demonstrated the ability of pruning techniques to reduce model size up to 50% without compromising performance. Furthermore, the experiments also demonstrated the ability to improve performance by reducing overfitting.

1 Key Information to include

- Mentor: None
- External Collaborators (if you have any): None
- Sharing project: None

2 Introduction

In Natural Language Processing (NLP), Bidirectional Encoding Transformer (BERT) (Devlin et al., 2018) is the usual starting point for building state-of-the-art language models. However, as the trend in machine learning is to keep scaling up the models to enormous sizes, the computational cost of both training and inference is becoming prohibitively demanding. Therefore, there is an increasing need for techniques that allow to alleviate the computational cost of these networks without compromising performance (Ganesh et al., 2021).

Herein, we explore the use of pruning techniques to alleviate the computational cost of inference in BERT-based models. The technique consists of removing redundant or less important network weights or structures of the fully-trained network. These techniques have been demonstrated to help reducing the model size by considerable amounts without affecting performance on the downstream tasks. Furthermore, pruning methods have shown the ability to improve robustness and even overall performance (Ganesh et al., 2021). Two main pruning types are considered: unstructured and structured.

Unstructured pruning (Chen et al., 2020a; Ganesh et al., 2021) aims to identify and remove the set of less important individual weights. In other words, from all the trainable parameters in the network, pruning aims to identify the set of weights that do not have a considerable effect on the overall performance. Generally, the main advantage of this type of pruning is the capability to allow larger sparsity levels when compared to the other type (structured pruning). However, the main disadvantage is that the method does not yield straightforward practical computational savings.

Structured pruning aims to identify and remove complete structural components that are redundant or less important. For example, in the case of BERT, some works try to reduce the number of encoding units, attention heads or the embedding size (Xia et al., 2022; Michel et al., 2019). The main advantage of this approach is the straightforward practical computational savings. In contrast, the main disadvantage is that the method does not allow as higher sparsity values when compared to the unstructured case.

In this work, these two types of pruning techniques are assessed in BERT-based models for three specific tasks: sentiment analysis, paraphrase detection, and similarity prediction.

3 Related Work

A related work is the lottery ticket hypothesis (Chen et al., 2020b,a) which aims to translate computational savings, obtained via pruning, from inference to training. Previous works state that the weights to be pruned are not only identifiable at the end of the training procedure, but also at the early stages of training. Therefore, the general methodology consists of, first, performing a short and aggressive training. Then, the method identifies the prunable weights and removes them to reduce model complexity. Finally, the training is resumed with a considerably computationally cheaper and faster model.

4 Approach

This Default Project had two main parts as shown in Fig. 1. First, the multitasking BERT was implemented following the default guidelines. The model aimed to performed three tasks simultaneously: sentiment analysis, paraphrase detection, similarity prediction. Second, this project assessed the effectiveness of pruning techniques to reduce the scale of BERT. For this part, Multitasking BERT was split into three separate networks, one for each tasks. Then, structured and unstructured pruning techniques were performed and the impact on performance was measured.

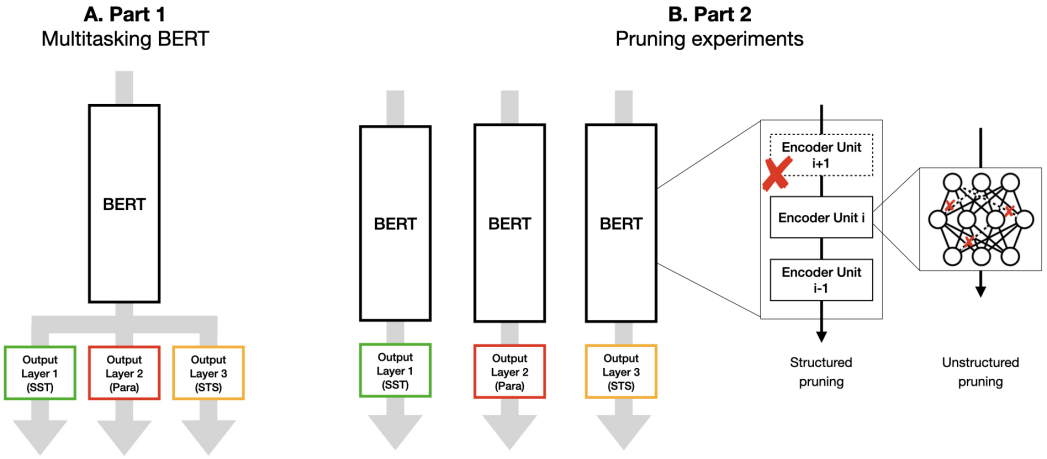

Figure 1: The Default Project had two main parts: **(A)** Multitasking BERT and **(B)** Pruning experiments. **(A)** Multitasking BERT is implemented following the Default Project guidelines. **(B)** For the pruning experiments, Multitaskint BERT is split into three separate networks and two types of pruning techniques are assessed on the resulting three networks.

4.1 Multitasking BERT

Multitasking BERT was implemented following the Default Project guidelines as shown in Fig. 1A. The starting point was a pre-trained BERT model. Output layers were appended for each considered task. For the sentiment analysis, the output layer considered a dropout layer, a linear layer, and a cross-entropy loss. For the paraphrase detection, the BERT outputs were concatenated and followed by a dropout layer, linear layer, and a binary cross-entropy loss. For the similarity prediction, the

BERT outputs were forwarded through dropout layers and linear layers. Then, the similarity was measured by a scaled cosine similarity and the result was forwarded to an $\ell 1$-loss.

4.2 Pruning experiments

Multitasking BERT was split intro three separate networks, one for each task as shown in Fig. 1B. The same output layers were used. Then, two types of pruning were considred.

Structured pruning (Fig. 1B) aims to remove components or structures without impacting performance considerably. The process consists on ranking the structures to prune in order of importance based on a chosen criteria. In this work, the training loss increment was used as criteria following Michel et al. (2019). Then, the ranked layers were removed successively to increase sparsity while measuring performance metrics.

Unstructured pruning (Fig. 1B) aims to remove specific network weights without impacting performance. Similarly, the process ranks the weights in order of importance according to a chosen criteria. In this work, the $\ell 1$-norm was considered as criteria following Chen et al. (2020a). Then, the weights were remove successively to achieve increased levels of sparsity while measuring the performance metrics.

5 Experiments

Two types of experiments were performed for the three considered tasks. First, the multitasking BERT was evaluated through evaluation metrics to assess performance. Second, the pruning techniques were tested on the BERT-based models for each task by gradully increasing the sparsity level and measuring the evaluation metrics to assess the influence on performance.

5.1 Data

We use the three main datasets provided for the default project:

1. Stanford Sentiment Treebank for sentiment analysis task
2. Quora dataset for the paraphrase detection task
3. SemEval STS Benchmark Dataset for the similarity prediction task

5.2 Evaluation method

In both parts of the experiments, the performance is evaluated through accuracy metrics. In the sentiment analysis task, the metrics used are prediction accuracy and F1-score. In the paraphrase detection task, the metric used is the detection accuracy. In the similarity prediction task, the metric used is the Pearson correlation.

Additionally, for the pruning experiments, the metrics per sparsity level were evaluated and the metrics drop (or increase) were assessed to determine the level of sparsity allowed by each pruning technique.

5.3 Experimental details

All the experiments were run on a GPU Titan RTX 24 GB. All BERT models used the base minBERT implementation provided for the Default Project, which was composed of 12 encoder units with 12 attention heads each.

Training for the experiments had the following configurations:

Parameters	Multitasking	Pruning experiments		
		SST	Paraphrase	STS
Batch size	8 (SST, STS) / 40 (Para)	8	8	40
Dropout	30%	30%	30%	30%
Learning rate	1e-5	1e-5	1e-5	1e-5
Epochs	10	10	10	10
Training time	3h	15min	15min	2h

5.4 Results

5.4.1 Multitasking BERT

Table 1 presents the performance metrics for the Multitasking BERT. The metrics are acceptable for the three tasks. Furthermore, the results for Dev and Test datasets are similar, which demonstrates the model's capability to generalize to new datasets. Therefore, the implementation is acceptable.

Table 1: Summary of metrics for Multitasking BERT

Dataset	SST Accuracy	Paraphrase Acc.	STS Correlation	Overall
Dev	0.539	0.729	0.479	0.582
Test	0.534	0.731	0.487	0.584

5.5 Pruning experiments

Fig. 2 shows the resulting evaluation metrics for the sentiment analysis network at each level of sparsity for each pruning technique. Both pruning techniques allowed reducing the number of used weights by a considerable amount without compromising the metrics. Furthermore, unstructured pruning allowed higher sparsity values before having a considerable drop.

Fig. 4 shows the resulting evaluation metrics for the paraphrase detection network at each level of sparsity for each pruning technique. The behaviour is similar than in the sentiment analysis case.

Fig. 3 shows the resulting evaluation metrics for the similarity prediction network at each level of sparsity for each pruning technique. In this case, surprisingly, pruning improved the evaluated metrics. This phenomena can be explained with the evaluation metrics on the training set, where a clear overfitting is shown. Therefore, pruning is improving performance by reducing overfitting via model complexity reduction.

6 Analysis

For the three tasks studied, unstructured pruning showed a superior capability to reduce the number of network weights without compromising performance considerably. In both the sentiment analysis and paraphrase detection, the model performed similarly for both unstructured and structured pruning. However, in the similarity prediction task, pruning was able to even improve performance. Both cases are assessed next.

In the sentiment analysis network, unstructured pruning was able to remove around 50% networks weights without seriously impacting the performance, as measured by the accuracy and F1-score, on the validation sets (Fig. 2A,B). Above 50% sparsity, the performance suffered a noticeable drop. It can be hypothesized that the distribution of network's weights resemble a bimodal distribution with two peaks with roughly 50% of the weights each. One peak would correspond to low-valued weights that are not important and do not impact the network performance when pruned. Whereas the other peak would contain high-valued and important weights that can significantly impact performance. When pruning above 50% of the weights, high-valued weights would start to be pruned which would cause the sudden and marked decrease in performance.

Conversely, structured pruning allowed only up to 30% of sparsity before undergoing significant performance decrement (Fig. 2A,B). However, surprisingly, this technique also allowed roughly up to 70% sparsity, which corresponds to pruning 11 out of the 12 encoding layers, before undergoing the same marked performance drop as unstructured pruning. Therefore, it can be hypothesized that

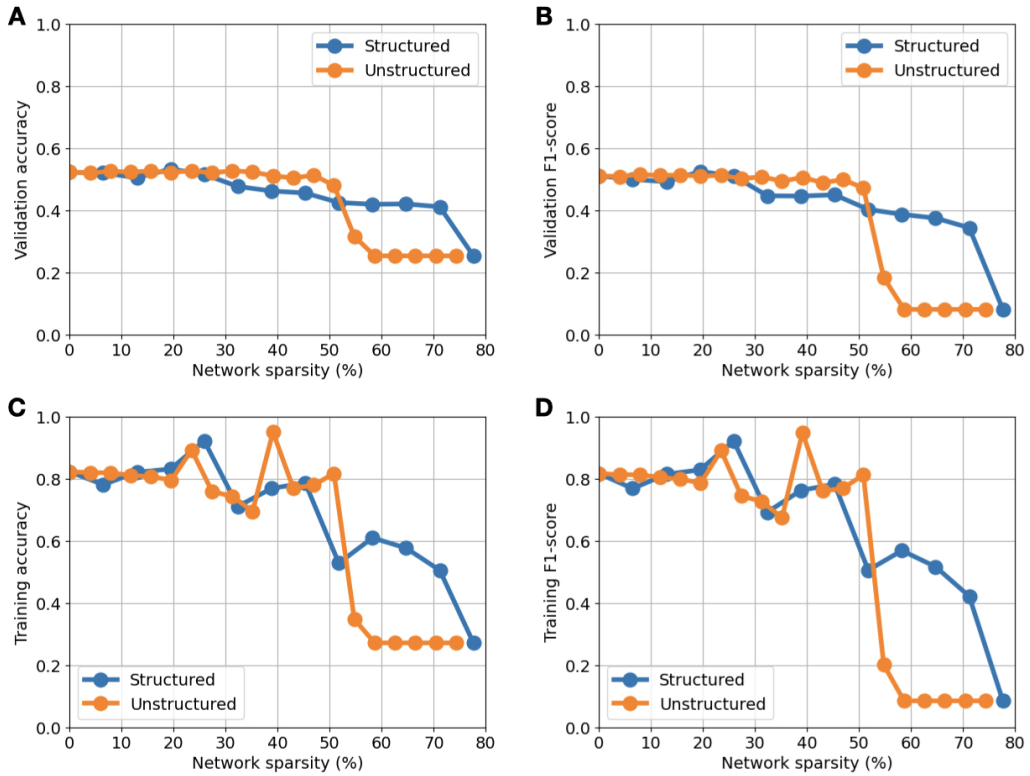

Figure 2: Pruning on sentiment analysis network. The performance metrics of the pruned network versus network sparsity for the validation (Dev) dataset are shown in **(A,B)**. The performance metrics for the traning dataset are shown in **(C,D)**.

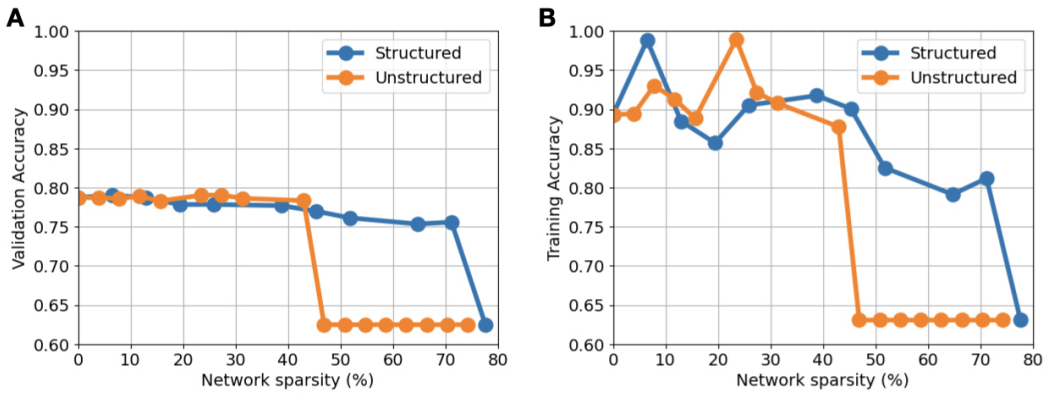

Figure 3: Pruning on paraphrase detection network. The performance metrics of the pruned network versus network sparsity on the **(A)** validation (Dev) dataset and **(B)** training dataset are shown.

BERT with only one encoding layer can actually perform the task reasonably well as suggested by the work by **?**.

In the paraphrase detection task, both pruning techniques performed similarly. Unstructured pruning allowed up to roughly 40% sparsity before having the same marked performance drop. Thus, it can be inferred that the weights have the same bimodal distribution than in the sentiment analysis network. Similarly, structured pruning allowed up to roughly 45% sparsity before significantly impacting performance and up to 70% sparsity before the sudden and marked performance drop at the end.

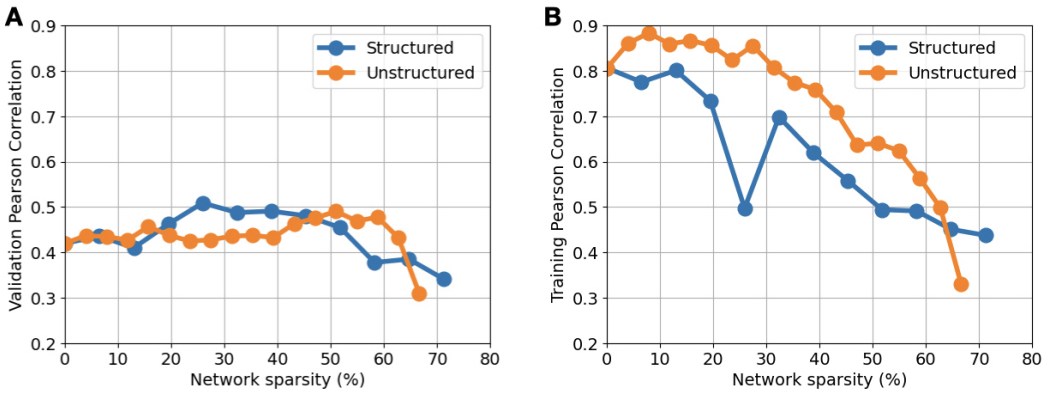

Figure 4: Pruning on similarity prediction network. The performance metrics of the pruned network versus network sparsity on the **(A)** validation (Dev) dataset and **(B)** training dataset are shown.

Likewise, the last result suggests that BERT is able to perform reasonably well in this task with only one encoding layer.

The similarity prediction tasks yielded different results. Pruning did not only maintained performance but also improved it. In Fig. 3A, both pruning techniques improved the performance metric (Pearson correlation) on the validation set. Unstructured pruning improved performance up to roughly 65% sparsity before a sudden performance drop. Specifically, from 40% to 65% sparsity, the network showed a considerably superior performance. Likewise, structured pruning improved performance from 20% to roughly 50% sparsity. These results can be explained by analyzing the performance on the training set (Fig. 3B), where the network is clearly overfitting to the data. Performance for both structured and unstructured pruning at low sparsity values are above 80% with peaks around 90% accuracy, whereas the accuracy in the validation set (3A) is around 40%. A common technique to reduce overfitting is to reduce the network complexity such that is more difficult to memorize datasets. Thus, in this case, pruning is reducing model complexity by reducing the number of weights (unstructured) or encoding units (structured), which would explain the performance improvement.

7 Conclusion and Future Work

Through multiple experiments, this work demonstrated that pruning techniques can help reduce model complexity by considerable amounts without seriously impacting performance. In both the sentiment analysis and paraphrase detection tasks, the models were able to prune a large portion of the networks (50% for unstructured and 30% for structured) without compromising validation accuracy. Furthermore, in the similarity prediction task, it was shown that pruning can actually improve performance by reducing overfitting.

However, the work had multiple limitations which are discussed next. First, the experiments consider only one training run. Since the training process considers a stochastic optimization and random initialization, a better approach would be to execute multiple training runs and report the statistics such as medians and standard deviations. Second, it can be argued that since structured and unstructured had different weight ranking criteria (training loss increase versus $\ell 1$-norm), the comparison is not completely fair. For example, a better approach would have been to have the same criteria for both techniques such as the $\ell 1$-norm. However, the intention of this work was to assess with proven strategies and, thus, the works of Chen et al. (2020a) and Michel et al. (2019) were followed for the unstructured and structured pruning respectively.

Third, although the main motivation of this work was to improve computational efficiency of BERT, the work did not include computational workload measurements due to time constraints. This part will be executed as part of the future work and improvements. However, from the literature, it can be inferred that unstructured pruning will not yield significant improvement even at 50% sparsity. Nevertheless, structured pruning is expected to improve computational efficiency considerably since this work demonstrated BERT's ability to perform reasonably well with only one encoding layer.

Finally, another possible direction for future experiments is to explore the translation of computational savings to the training procedure via lottery ticket hypothesis works (Chen et al., 2020b,a). This hypothesis states that the unpruned weights (or structures) are not only identifiable at the end of the training procedure, but also at the very early stages. Therefore, a common approach is to execute a short and aggressive early training, with enough iterations to be able to identify the important weights. Then, the pruning procedure should follow and the training procedure should resume with a model with considerably less complexity.

References

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020a. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems*, 33:15834–15846.

Xiaohan Chen, Yu Cheng, Shuohang Wang, Zhe Gan, Zhangyang Wang, and Jingjing Liu. 2020b. Earlybert: Efficient bert training via early-bird lottery tickets. *arXiv preprint arXiv:2101.00063*.

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*.

Prakhar Ganesh, Yao Chen, Xin Lou, Mohammad Ali Khan, Yin Yang, Hassan Sajjad, Preslav Nakov, Deming Chen, and Marianne Winslett. 2021. Compressing large-scale transformer-based models: A case study on bert. *Transactions of the Association for Computational Linguistics*, 9:1061–1080.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models. *arXiv preprint arXiv:2204.00408*.