Optimizing LLMs with Pruning, Distillation, and Adapter Enhancements

Anonymous Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
`email@domain`

Anonymouser Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
`email@domain`

Anonymousest Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
`email@domain`

Abstract

The increasing size of pre-trained language models (PLMs) has led to requires a better understanding of optimization methods. In this study, we propose three approaches, structured pruning, distillation, and adapters which we apply to the Bidirectional Transformers for Language Understanding (BERT) model. Our goal is to assess the correlation and interdependence, or lack thereof, among these three methods. Our results focus on the GLUE task with benchmarks including model size, accuracy, rejected emission rate, and speedup. For instance, the combination of pruning, distillation, and adapters can reach 4x inference speedup with a size reduction of 85% and 200% emissions savings and an average accuracy of 50 instead of 85 approximately. Before each use, we must consider the performance-reduction-speed trade-off. In this work, we propose several scenarios depending on the desired outcome. For example, if we want to maintain performance and reduce the model by 65%, then we should opt for the combination of adapters based tuning + pruning with a sparsity ratio of 0.6.

1 Introduction

Pre-trained language models shown to be effective for improving many natural language processing task (Devlin et al., 2019). However the growing demands of these models comes with an increase in model size, and therefore high energy costs. Patterson et al. estimates the training of the GPT-3 Large Language Model (LLM) at 552 tonnes of CO_2eq. Or even 30 tCO_2eq for BLOOM (Luccioni et al., 2022). This is equivalent to 30 round-trip flights from Copenhagen to New York city by one passenger[1]. And despite this, it only takes into account the dynamic power consumption and not the energy involved during storage, deployment, and request made by the user.
Thus we can easily understand the urgent need to understand and apply optimisation methods. We will study structured pruning, distillation, and adapter-based tuning, and we attend to answer the following question:

To what extent does the combination of various model size reduction strategies influence the performance, efficiency, and speed of the model?

To answer this question, we will focus on different size reduction methods. We will study the size, speedup, and performance of the final model. We will seek to determine whether some methods are more effective for reduction, for speedup, or those that least affect performance. We will also explore whether certain methods can be combined to be more effective.

2 Related work

Structured pruning is widely used to improve performance and reduce the size of models in NLP, enabling compression that allows models to run on edge devices, such as mobile phones. Recent work has focused on various aspects of structured pruning, including attention head pruning (Michel et al., 2019), intermediate layer pruning (McCarley et al., 2021), pruning entire FFN layers (Prasanna et al., 2020), and both coarse and fine-grained

[1]according to Google flight emissions estimation

pruning (CoFi) (Xia et al., 2022). In this work, we build on the coarse and fine-grained pruning approach proposed by Xia et al. (2022).
Our approach to pruning involves inserting additional trainable parameters, called masks, into a transformer. Each mask is a vector of gate variables $z_{mask}^{(i)} \in \{0,1\}$ that control whether to prune the block parameter or not. This method relies on the pruning of attention heads proposed by Voita et al. (2019) and Michel et al. (2019), as well as the pruning of the intermediate activations in the feed-forward layers proposed by McCarley et al. (2021). The CoFi pruning adds the pruning of entire MHA layers and entire FFN layers in addition to head pruning and intermediate dimension pruning.
Distillation was introduced by Hinton et al. (2015). The concept allows us to reduce the BERT model size by 40% and accelerate its performance by 60%, while retaining 97% of its language understanding capabilities (Sanh et al., 2020b). The approach involves transferring knowledge from a larger teacher model to a smaller student model. The idea is to train the compact student model to replicate the predictions of the highly accurate teacher model. Combining distillation with pre-training and fine-tuning on an unlabeled corpus can further enhance performance, although it is computationally expensive (Turc et al., 2019).
A pertinent question is whether distillation can be effectively combined with pruning. This is explored by Xia et al. (2022), who propose using random initialization. In contrast, (Turc et al., 2019) demonstrate that pre-trained student models benefit more from depth than width, a property not observed in randomly initialized models. Moreover, this approach conserves only four layers. The fewer teacher layers retained in the student model, the greater the efficiency drop (Turc et al., 2019), and distillation becomes more power-intensive.
Adapters-based tuning has emerged as an alternative to fine-tuning (Houlsby et al., 2019). Adapters are new modules inserted between the layers of a pre-trained network, as illustrated in Figure 1. During training on a downstream task, only the adapter parameters are updated, adding just a few trainable parameters per new task and allowing for a high degree of parameter sharing. Adapter-based tuning requires significantly less computational resources compared to fine-tuning. However, some performance loss may occur, although previous studies (Houlsby et al., 2019) have shown that adapter-based tuning often achieves results comparable to fine-tuning.

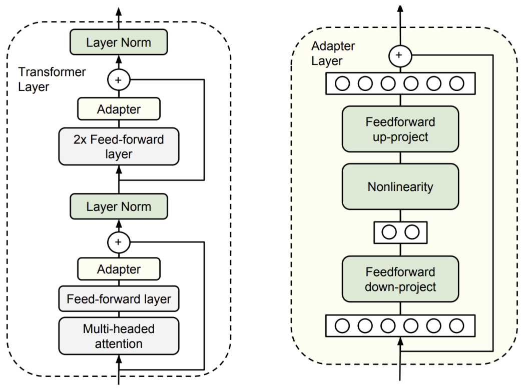

Figure 1: Adapters architecture on BERT model

3 Background

3.1 BERT

BERT (Vaswani et al., 2023) is a Transformer, which follow the encoder-decoder framework using stacked multi-head self-attention and fully connected layers for both the encoder and decoder. We used the pre-trained **google-bert/bert-base-cased** model with 12 layers and 12 attention heads.
The encoder consists of N layers, each containning a multi-head self-attention (MHA) layer and a feed-forward (FFN) layer.
An MHA are N_h heads that takes an input $x \in \mathbb{R}^d$ and ouput :

$$MHA(X) = \sum_{h=1}^{N_h} \text{Att}(W_Q^{(h,l)}, W_K^{(h)}, W_V^{(h,l)}, W_O^{(h,l)}, X)$$

The attention head h in layer l is parametrized by the matrices $W_Q^{(h,l)}, W_K^{(h,l)}, W_V^{(h,l)} \in \mathbb{R}^{d_h \times d}$ and $W_O^{(i)} \in \mathbb{R}^{d \times d_h}$, d_h is typically set to d/N_h and where :

$$Att(Q,K,V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

And the feed-forward layer is composed of 2 projection layers, up and down, parameterized by $W_U \in \mathbb{R}^{d \times d_f}$ and $W_D \in \mathbb{R}^{d_f \times d}$:

$$FFN(X) = ReLU(XW_U)W_D$$

3.2 Pruning

As explained in Section 2, our pruning method is based on structured pruning through CoFi pruning (Xia et al., 2022). This approach involves four masks, presented in the following formula. Let us consider four mask parameters: $z_{head}^{(i)}, z_{MHA}, z_{FFN} \in \{0,1\}$ and $z_{int} \in \{0,1\}^{d_f}$

Attention head pruning :

$$MHA(X) = \sum_{h=1}^{N_h} \mathbf{z_{head}^{(h)}} \cdot \text{Att}(W_Q^{(h,l)}, W_K^{(h)}, W_V^{(h,l)}, W_O^{(h,l)}, X)$$

Intermediate dimension pruning

$$FFN(X) = ReLU(XW_U) \cdot \mathbf{diag}(\mathbf{z_{int}}) \cdot W_D$$

Pruning of entire Multi-head Attention layers

$$MHA(X) = \mathbf{z_{MHA}} \cdot \sum_{h=1}^{N_h} z_{head}^{(h)} \cdot \text{Att}(W_Q^{(h,l)}, W_K^{(h)}, W_V^{(h,l)}, W_O^{(h,l)}, X)$$

Pruning of entire feed-forward layers

$$FFN(X) = \mathbf{z_{FFN}} \cdot ReLU(XW_U) \cdot diag(z_{int}) \cdot W_D$$

We apply L_0 regularization to the mask parameters (Louizos et al., 2018). The L_0 norm, which equals the number of non-zero heads, encourages the model to switch off less important heads. By following L_0 regularization on the mask parameters, we aim to achieve the desired sparsity.

3.3 Distillation

Knowledge distillation involves training a smaller model trying to replicate the behaviour of a larger model. Initially, the student model is configured by retaining layers [1,3,5,7,9,11] of the teacher model, preserving 50% of the original structure.

It is common to observe a softmax transformation as the final layer of a neural network. Softmax normalizes the model's outputs so they sum to 1, allowing them to be interpreted as probabilities. The softmax output layer converts the logit z_i, computed for each class into a probalbility, q_i by comparing z_i with other logits. And the temperrature T controls the output distribution:

$$p_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$$

The pretraining of the student model follow different objective functions. The cross-entropy loss is computed between the predicted and the true distribution, where the student's weights updated through backpropagation.

(Sanh et al., 2020b) introduced a distillation loss, $L_{ce} = \sum_i t_i * log(s_i)$, where t_i and s_i represent the estimated probabilities of the teacher and student distributions, respectively.

4 Experiments

We aim to compare different combinations by examining all possible scenarios involving the combinations of pruning (P), distillation (D), and adapter-based tuning (A). All possibilities are documented in the truth table presented in Table 1. Each row represents a training scenario where the value 1 indicates the activation of a specific optimization method and 0 indicates that the method is not applied in that training instance. This approach allows us to systematically evaluate and compare the impact of each combination of pruning, distillation, and adapter-based tuning on the BERT model.

Our goal is to analyze the behavior of the BERT model in relation to the pruning methods used. We seek to identify favorable configurations based on specific priorities, such as size reduction, performance, and model speed. This will help determine which combinations are most suitable for different use cases, depending on whether the focus is on minimizing model size, enhancing performance, or increasing speed.

We will only focus on the following four configurations: $\mathbf{P+D+A}$, $\mathbf{P+A}$, $\mathbf{D+A}$, $\mathbf{A}$ alone and finally the classic fine tuning (FT). Of course, we will compare our results to the baseline model (0,0,0). The other cases are

Pruning	Distillation	Adapters
1	**1**	**1**
1	1	0
1	**0**	**1**
1	0	0
0	**1**	**1**
0	1	0
0	**0**	**1**
0	0	0

Table 1: Compilation of All Optimization Possibilities for the BERT Model

studied in another article : D (Sanh et al., 2020a), P + D (Xia et al., 2022), P (Michel et al., 2019).

GLUE We have divided the tasks based on the number of trainable and test parameters. On one side, MRPC, RTE, STSB, and CoLA are considered to have fewer parameters. On the other side, MNLI, QQP, QNLI, and SST-2 have more parameters. Consequently, the "GLUE low" tasks will be trained over 100 epochs with an evaluation step of 50, whereas the "GLUE high" tasks will be trained over 20 epochs with an evaluation step of 500.

Generalization and Uncertainty We introduce uncertainty in certain values because we have run the training and testing multiple times to validate our results. This approach helps highlight any divergent values, as noted by (McCoy et al., 2020). By incorporating uncertainty, we can better assess the robustness and reliability of our findings.

5 Results and analysis

In this section we evaluate the different models obtained. The objective is also to evaluate the different parameters based on the sparsity ratio. We'll look at the following variables, efficiency, model size, speedup and emissions.

For the P+D+A combination, a sparsity level of 0.95 was not achieved for a large number of tasks. Due to the lack of stability in the results, it was not possible to include them in the graphs.

5.1 Efficiency

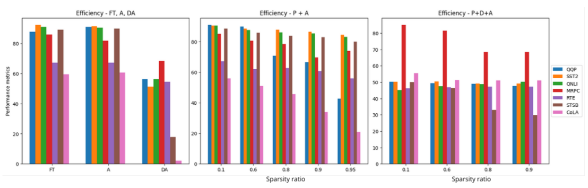

Figure 2: Evaluation of the model's performance. For the QQP, SST2, QNLI, MRPC, and RTE tasks, the evaluation metric is accuracy. For STSB (resp. CoLA), the metric is the combined score (resp. Matthew's correlation).

We note that thanks to the use of adapters, we achieve results as effective as fine-tuning. However, the addition of distillation leads to a drop in performance. Thus, the DA and PDA cases have accuracies below 60%, except for the MRPC task.

To maintain optimal performance, it is therefore recommended to focus on cases involving adapters and avoid the addition of distillation.

5.2 Model size

Model size is not modified when pruning is not involved, thus the cases FT, A and DA retains the same size.

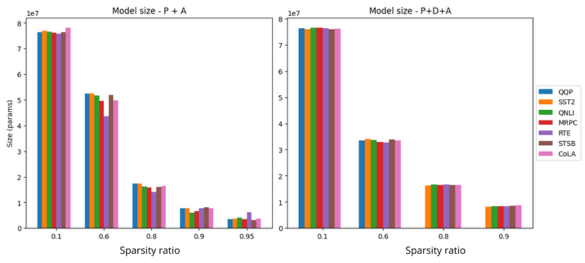

Figure 3: Model's size graph depending on the sparsity ratio of the pruning

By observing the model sizes in the P+A and P+D+A cases, we can highlight the advantages of using distillation. Indeed, for identical sparsity ratios, we see a significant reduction in volume. For a sparsity of 0.6, we save nearly 20 megabytes, which represents 22% of the total volume.

Thus, the addition of distillation is useful

Corpus	Task	$\|Train\|$	$\|Test\|$	Metrics
CoLA	acceptability	8.5k	1k	Matthews corr.
SST-2	sentiment	**67k**	1.8k	acc.
MRPC	paraphrase	3.7k	1.7k	acc.
QQP	paraphrase	**364k**	**391k**	acc.
STS-B	sentence similarity	7k	1.4k	Comb. corr.
RTE	NLI	2.5k	3k	acc.
QNLI	QA/NLI	**105k**	5.4k	acc.

Table 2: GLUE tasks descriptions : single sentence ; similarity and paraphrase ; inference

when we aim to drastically reduce a model's size at the cost of a reduction in its performance.

5.3 Speedup, emissions

correlation between speedup and emissions. We can consider a correlation. Give 1 +

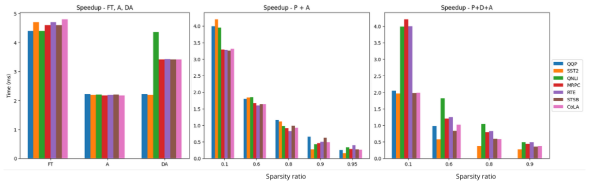

Figure 4: Caption

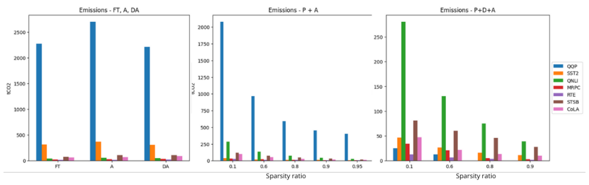

Figure 5: Caption

Regarding speedup, we observe that the addition of adapters significantly reduces inference time, cutting it in half. It also appears that the combination of pruning and distillation further decreases inference time. However, the effect of adding distillation is not consistent across all tasks. While tasks like SST2 and STSB experience a significant speedup, the QNLI task does not see any change in time. The combination of distillation and adapters is highly favored for reducing emissions. There is a strong link between inference time and emissions, as emission calculations are based on the time required for processing.

We note that the emissions measured for the P+D+A case are 10 times lower than those recorded in all other cases studied. It seems that distillation delivers excellent results in terms of reducing emissions.

5.4 Layers

In this section, we refer to Figure 6 available in the appendix. The notations MHA correspond to the Query, Key, Value, and Output layers, while FFN corresponds to the Up and Down layers. Given that $size(Query) = size(Key) = size(Value) = transpose(Output)$ and $size(Down) = transpose(Up)$.

In this section, we observe the layers that have been entirely removed, which are indicated by blacked-out boxes. The layers that are not blacked out are not necessarily preserved intact; they may be reduced to a much smaller dimension.

The intuition is that by adding the adapters, the essential information is concentrated within them. This makes the "other" layers less necessary, as they no longer carry the primary information, leading to more extensive pruning of these layers.

We observe that MHA layers are mostly preserved, while FFN layers are typically the first to be removed. Additionally, layers 0, 1, and 2 are generally kept intact, while the last layers are removed first and more extensively. The hypothesis concerning models using adapters is confirmed, as they have more layers removed compared to other cases like distillation and pruning (see (Xia et al., 2022)).

6 Conclusion

Carbon Impact Statement. This work contributed ... kg of CO2eq to the atmosphere and used 249.068 kWh of electricity. Copenhagen (Denmark), indice : . Result delivered by CodeCarbon (citer src).

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv:1810.04805 [cs].

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. ArXiv:1503.02531 [cs, stat].

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. ArXiv:1902.00751 [cs, stat].

Christos Louizos, Max Welling, and Diederik P. Kingma. 2018. Learning Sparse Neural Networks through L_0 Regularization. ArXiv:1712.01312 [cs, stat].

Alexandra Sasha Luccioni, Sylvain Viguier, and Anne-Laure Ligozat. 2022. Estimating the Carbon Footprint of BLOOM, a 176B Parameter Language Model. ArXiv:2211.02001 [cs].

J. S. McCarley, Rishav Chakravarti, and Avirup Sil. 2021. Structured Pruning of a BERT-based Question Answering Model. ArXiv:1910.06360 [cs].

R. Thomas McCoy, Junghyun Min, and Tal Linzen. 2020. BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance. ArXiv:1911.02969 [cs].

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are Sixteen Heads Really Better than One? ArXiv:1905.10650 [cs].

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon Emissions and Large Neural Network Training.

Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When BERT Plays the Lottery, All Tickets Are Winning. ArXiv:2005.00561 [cs].

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020a. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. ArXiv:1910.01108 [cs].

Victor Sanh, Thomas Wolf, and Alexander M. Rush. 2020b. Movement Pruning: Adaptive Sparsity by Fine-Tuning. ArXiv:2005.07683 [cs].

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-Read Students Learn Better: On the Importance of Pretraining Compact Models. ArXiv:1908.08962 [cs].

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. ArXiv:1706.03762 [cs].

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned. ArXiv:1905.09418 [cs].

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured Pruning Learns Compact and Accurate Models. ArXiv:2204.00408 [cs].

7 Annexe

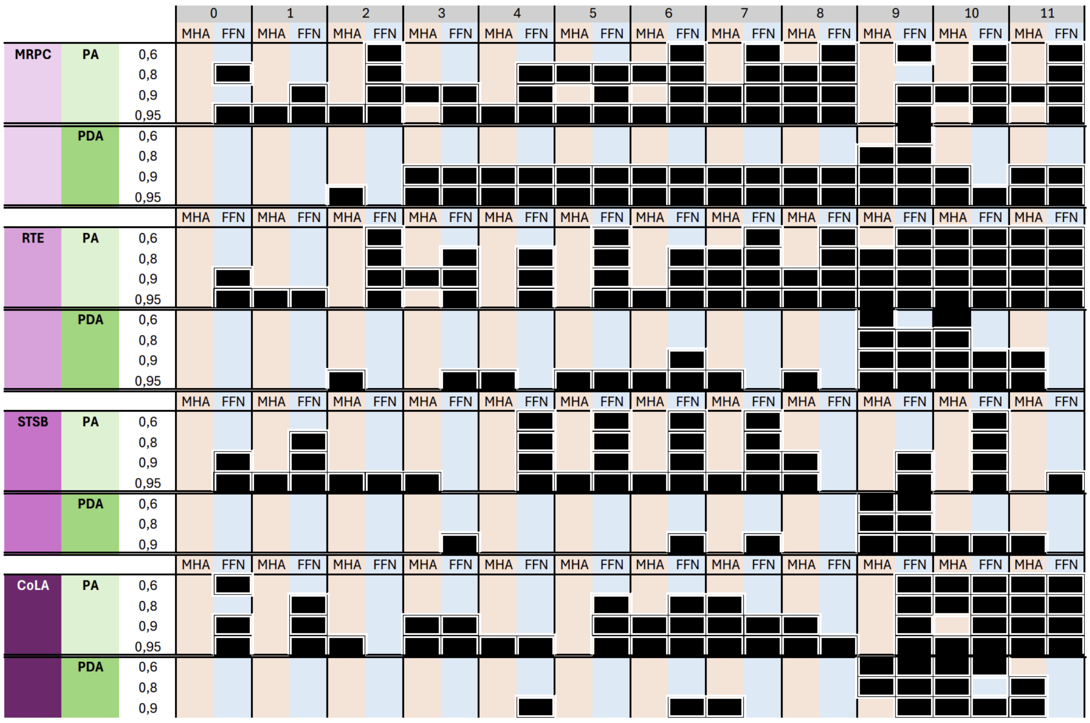

Figure 6: Caption