Analyzing Methods to Increase Training Efficiency of Transformer-Based Language Models

Ege Atay
DS 680
New Jersey Institute of Technology
ea353@njit.edu

Abstract

Transformer-based language models have revolutionized NLP by allowing for better understanding and generation. Since their introduction, models like BERT or GPT have been very successful. As these models become increasingly complex, they require vast amounts of training data and computational resources, leading to challenges in terms of scalability.

This presents a critical need for analysis to optimize transformer models' application and address drawbacks. In this project, I experimented with various methods to speed up training without a significant negative impact on model accuracy.

1 Introduction

Transformer-based language models, like BERT and GPT, have transformed natural language processing by enhancing text understanding and generation. They are vital in many NLP tasks but face issues with large data needs and computational demands. This project aims to explore ways to train models more efficiently.

Various methods have been in use to increase training speed and efficiency of language models, and many more are put forward regularly. Such methods have enabled training more accurate models faster. In this paper, I will explore the performance and resource impacts of implementing such methods, as well as exploring other methods or parameters that optimize the training cycle for attention based models.

This report includes sections on the history of this topic, related past studies, experimentation methodology, results, and final conclusions; explaining the approach and findings in improving transformer efficiency for NLP tasks.

2 Related Work

2.1 Background

The practice of using data science on text sources is not brand new. The field of Natural Language Processing has been evolving rapidly with both hardware innovations and theoretical advancements in machine learning. There are many sources one could refer in order to learn more about this field.

The most common target in NLP could perhaps be the problem of sentiment analysis, labeling sentences on whether the content is positive or negative. This task is often used as a benchmark to compare the performances of different models.

2.2 Literature Review

One settled source on methods to increase training speed and accuracy is the publication by Maas et al. (2011), demonstrating that learning word vectors can significantly enhance sentiment analysis performance. An issue with this paper is that it is over a decade old and thus does not take into account many recent developments in machine learning.

The study that brought generational improvement to this field is by Vaswani et al. (2023), which set the standards for the Transformer architecture that we are trying to optimize in this project. The transformer architecture works by utilizing solely the attention mechanism. Exploring the workings behind this architecture is highly relevant to optimizing its performance.

There have been various academic publications in the past few years focusing on how to streamline the model training process. The publication by Gao et al. (2024) explores various methods to increase the long-distance context learning of language models. The study found that training with a sequence length beyond the evaluation length boosts long-context performance. However, this study focuses on improving already massive language models, with billions of parameters, making

the content of this study infeasible to experiment with using simple hardware.

Another recent publication is (Zaken et al., 2022), which takes a different approach and demonstrates the effectiveness on focusing on training only the biases of each layer instead of both the weights and biases. This study demonstrates the impact of making models much simpler to train, and it is a method that could be applied to many models across the board. The findings of this study are intriguing and I will try to demonstrate the impact of the methods put forward in this publication.

3 Methodology

3.1 Dataset Exploration

To implement this project, I experimented with different datasets with various features and sizes. I used the following datasets:

IMDB dataset by Maas et al. (2011) from Stanford. This is a binary classification dataset that is used extensively in Natural Language Processing tasks as a benchmark. With this dataset, the model's objective is to categorize movie reviews as either positive or negative. It has 100,000 samples.

AG News dataset by Zhang and LeCun (2005). This is a multi-class classification dataset. The goal of the model with this dataset is to correctly classify the news article into the correct class from World, Sports, Business, or Science & Technology. It has over 127,000 samples.

The Yelp Polarity dataset by Zhang et al. (2015). The *Large Yelp Review Dataset* is conceptually similar to the IMDB dataset in that it is used for binary sentiment classification. The task involves determining whether a Yelp review, which typically covers restaurants, is positive or negative. Like the IMDB dataset's focus on categorizing movie reviews, the binary Yelp dataset aids in training models to differentiate between favorable and unfavorable sentiments using labeled data. This is a much larger dataset than the others used, coming in at almost 600,000 samples.

Dataset	**IMDB**	**AGN**	**Yelp**
Size	83 MB	20 MB	274 MB
Rows	100,000	127,600	598,000
Classes	2	4	2

Table 1: Properties of the datasets used in the experiments

3.2 Model Implementation

To accurately compare the difference in efficiency, performance, and accuracy of the training methods, I standardized the model types and parameters. The architecture also includes sequence padding and truncation through the tokenizer, ensuring consistent input dimensions for the model during training and evaluation.

The chosen model architecture is an instance of the *BertForSequenceClassification* from the transformers library (Wolf et al., 2020), with the corresponding BERT variant (Turc et al., 2019) (Bhargava et al., 2021) which are selected from reduced-size version of the standard BERT model. The models are loaded as pre-trained, but without any trained classifier tokens.

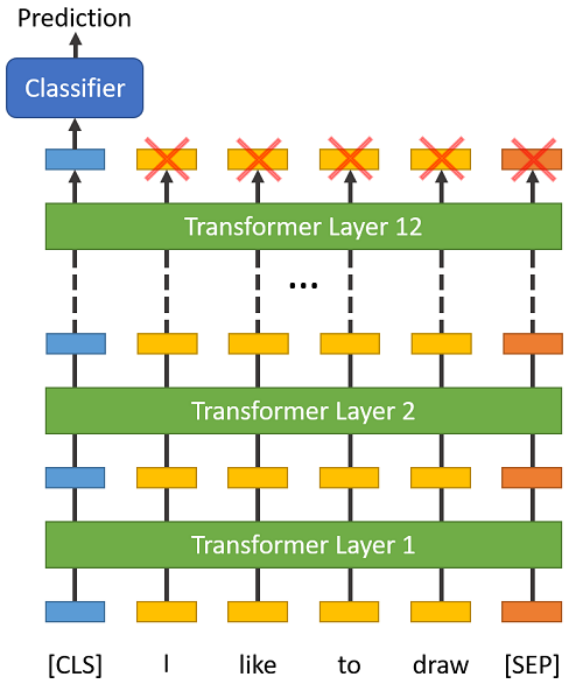

Figure 1: Diagram of the classification model, with BERT as an feature extractor backbone and classification head is appended to the last layer's CLS token embedding. McCormick and Ryan (2019)

BERT was chosen as a baseline model because of its effectiveness in NLP tasks. BERT is also able to capture contextual information well thanks its attention architecture. The focus of this study is finding methods to increase the efficiency of training models of this architecture.

The choice of reduced-size versions of BERT with fewer layers and parameters were to make it more computationally efficient and suitable for experimentation on available hardware. Using var-

ious sizes of reduced-size BERT allows us to explore the impact of these applications with respect to the used model's layer size.

The optimizer used is *AdamW*, which is an extension of Adam optimizer with weight decay. This was used for better generalization performance.

The train batch size and evaluation batch size are both set to 8. Weight decay is set at 0.01 to prevent overfitting by penalizing larger weights. Cosine annealing scheduler is employed to adjust the learning rate over the training process, which is known to increase accuracy (Loshchilov and Hutter, 2017).

3.3 Experiments

In the experiments, I systematically evaluated the performance and efficiency of the modified trainable parameters method compared to traditional fine-tuning approaches. This was done by training a sequence classification model, and then running the same training loop with the modified trainable parameters. The sequence classification model was instantiated from a reduced size BERT variant.

The experiments run were with the combinations of numerous variables. Metrics from each experimentation cycle were recorded and analyzed, including The models loss value on the training dataset, loss value on the validation dataset, and models accuracy percentage score on the validation dataset. The loss value on the training dataset was recorded to monitor any tendency for over-fitting, and loss value on the validation dataset along with accuracy percentage scores were recorded to effectively contrast between the parameter combinations.

This comparative analysis was conducted across the IMDB, AG News, and Yelp Polarity datasets to assess the generalization of the approach across tasks of varying complexity, class size, and row count.

After the training cycles were executed for each of the parameters and dataset combinations, the metrics were logged and analyzed.

4 Results and Analysis

4.1 Results

In all the comparisons made across model size and dataset combinations, the bias-only model's training cycles were *consistently* faster, by total training time and measured iterations per second. On average, the bias-only training was 16% faster than end-to-end training. See Figure 2 as an example. This establishes that training only the bias of layers is faster than training the biases and the weights.

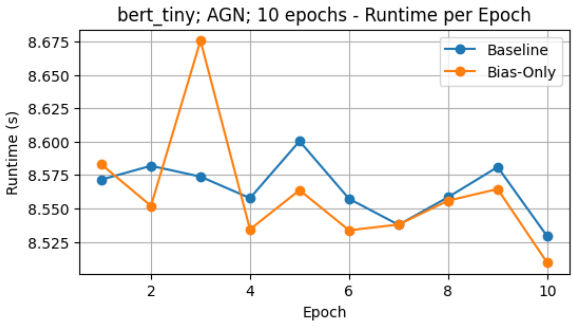

Figure 2: Runtime per epoch of reduced size BERT variant (2 Layer, 128 Hidden) on AG News dataset

This is a study on transformer model efficiency. In this domain, efficiency is a measure of training time as well as accuracy. The authors of BitFit study (Zaken et al., 2022) claim that this could increase model accuracy as well. The results of the experiments accuracy recordings are tabulated below.

BERT Size	Setting	# Params	IMDB	AGN	YELP
2L / 128H	E2E	4,386,178	76.84	89.78	93.56
	Bias only	3,330	82.28	88.21	88.32
4L / 256H	E2E	11,171,074	48.92	25.09	49.88
	Bias only	12,290	85.04	89.91	92.14
4L / 512H	E2E	28,764,674	50.48	24.76	50.12
	Bias only	24,578	88.92	91.02	94.23

Table 2: Results of experiments run on both baseline (E2E) models and bias-only trained models. Values under dataset are the validation accuracy percentages. Under *BERT Size*, *L* signifies the number of transformer layers, and *H* signifies the dimension of the vector representations in each layer.

4.2 Analysis

The results of the experiments require some interpretation. Although there are varying levels of performance between each dataset, the focus of this study is on the impact of implementing the bias-only training method.

See Figure 3 for the comparison of implementing the bias-only training method on the smallest size BERT variant. The graphs on the left side are for the IMDB dataset. This is the most streamlined output to analyze: the result of this experiment is that the bias-only model has a lower loss value and higher accuracy across the board on all 10 epoch runs. The graphs on the right side of Figure 3 are

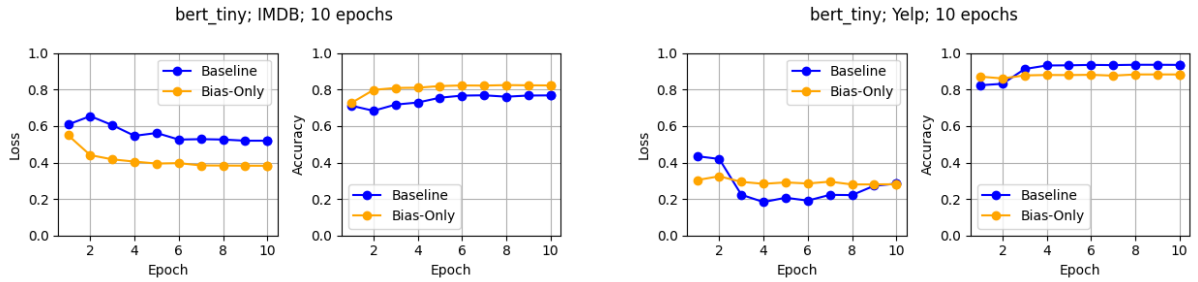

Figure 3: Comparison of reduced size BERT variant (2 Layer, 128 Hidden) on datasets of different size.

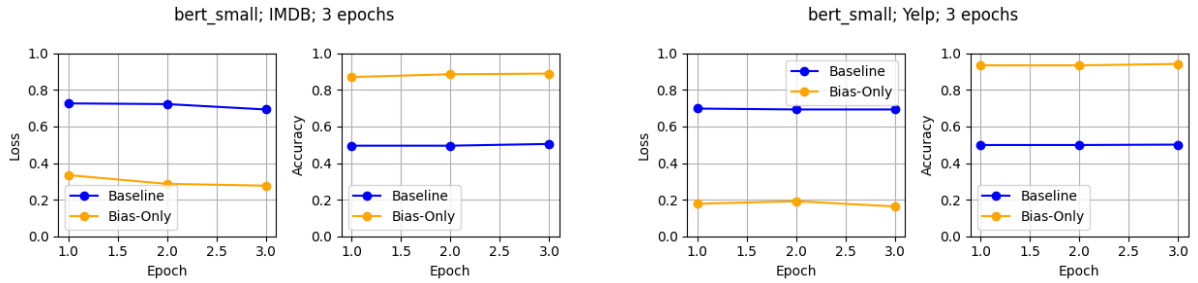

Figure 4: Comparison of reduced size BERT variant (4 Layer, 512 Hidden) on datasets of different size.

for the much larger Yelp dataset, with six times the size of the former, and here the implementation of the bias-only training method yields results in a different way where in the first few epochs the baseline E2E model outperforms the bias-only trained model but the end-to-end baseline model is able to catch up after many epochs.

Now consider the content of Figure 4 for the comparison of implementing the bias-only training method on the largest size BERT variant. On this figure, we notice that there is not much progress among epochs in either dataset. Although the only difference between the models in each plot is which parameters require training, the loss and accuracy values are vastly different. I believe this is because training 28,764,674 parameters is simply not converging on my hardware while training 24,578 parameters is efficient enough to yield results. Do note that the formerly addressed 10 epochs on IMDB dataset took less than 9 minutes, while the training runtime of the Yelp Polarity dataset on this model took over 4 hours on the same hardware. Although this is among the reduced-size variants of BERT, it is the largest model I experimented with.

Although the progress appears different between BERT variants of different sizes, we conclude that the reasoning behind the behaviors of the metrics are the same: Training fewer parameters is vastly more efficient and thus leads to better performance in otherwise identical environments.

5 Conclusion

5.1 Summary

In this study, we conclude that although generational improvements in hardware and new model architectures bring vast amounts of change in this field, simple methods we can implement also have significant impact on the real-world performance of various models in NLP.

We demonstrated that training only the bias in language models greatly increase accuracy while decreasing training time and hardware demand. This is especially the case when the model has a high number of parameters compared to what our hardware can feasibly handle.

The implication of these findings is that even with the increasing complexity of state-of-the-art language models, practitioners in the field of NLP can achieve significant efficiency gains by applying clever methods. This highlights the importance of exploring not just the architecture of models but also the methods used for training them, especially in resource-constrained environments. By reducing the number of training parameters, we can improved performance metrics and improve training time without requiring vast amounts of computing power.

5.2 Future Directions

Further studies could be done on the impact of these methods in other fields of machine learning. The methods I applied are relevant to all practices where a trained base model is being fine-tuned for a specific application.

My implementation was done solely on the hardware I have at hand, and many of the training for the leading models are done on hardware magnitudes of times more powerful. Implementing these methods on full sized models would require serious hardware, which I would consider the very next step.

5.3 Project Reflection

Working on this project was an opportunity for me research the newest publications in this field. This also allowed me to get hands on experience with various NLP tools to create a real working implementation. I believe the experience of creating a concept and then detailing it in writing has been the most informative process of this semester.

From this project I gained both technical and academic knowledge. I learned how to apply theoretical concepts outlined in research publications as well as how to search for specific knowledge in the academic space.

References

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in nli: Ways (not) to go beyond simple heuristics. *Preprint*, arXiv:2110.01518.

Tianyu Gao, Alexander Wettig, Howard Yen, and Danqi Chen. 2024. How to train long-context language models (effectively). *Preprint*, arXiv:2410.02660.

Ilya Loshchilov and Frank Hutter. 2017. Sgdr: Stochastic gradient descent with warm restarts. *Preprint*, arXiv:1608.03983.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Chris McCormick and Nick Ryan. 2019. Bert fine-tuning tutorial with pytorch. `http://www.mccormickml.com`.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *CoRR*, abs/1908.08962.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *Preprint*, arXiv:2106.10199.

Xiang Zhang and Yann LeCun. 2005. Ag news classification dataset. `http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html`.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. *arXiv:1509.01626 [cs]*.