On Enhancing Fine-Tuning for Pre-trained Language Models

Betka Abir,[1] Zeyd Ferhat,[4] Barka Riyadh,[3] Boutiba Selma,[2] Zineddine S. Kahhoul,[2]
Tiar M. Lakhdar,[2] Ahmed Abdelali,[5] and Habiba Dahmani[3]
[1] Laboratory of VCS, [2] Laboratory of IL3CUB, University of Biskra, Algeria
[3] Electrical Engineering Department, [4] Department of Electronics, University of M'sila, Algeria
[5] Qatar Computing Research Institute, HBKU, Qatar
{betkaabir, zeydferhatz, barkariyadh06}@gmail.com,
{selma.boutiba, zineddine.kahhoul, mohamedlakhdar.tiar}@univ-biskra.dz
aabdelali@hbku.edu.qa, habiba.dahmani@univ-msila.dz

Abstract

The remarkable capabilities of Natural Language Models to grasp language subtleties has paved the way for their widespread adoption in diverse fields. However, adapting them for specific tasks requires the time-consuming process of fine-tuning, which consumes significant computational power and energy. Therefore, optimizing the fine-tuning time is advantageous. In this study, we propose an alternate approach that limits parameter manipulation to select layers. Our exploration led to identifying layers that offer the best trade-off between time optimization and performance preservation. We further validated this approach on multiple downstream tasks, and the results demonstrated its potential to reduce fine-tuning time by up to 50% while maintaining performance within a negligible deviation of less than 5%. This research showcases a promising technique for significantly improving fine-tuning efficiency without compromising task- or domain-specific learning capabilities.

1 Introduction

Neural based Language Models are functions or algorithms that are trained to predict the likelihood of a sequence of words (Devlin et al., 2019; Radford et al., 2019). These models were trained using large volumes of textual content and are able to provide an accurate approximation for language features and structure. These models provide an important tool for analyzing and understanding the nuance of language, as well as for building applications that rely on natural language understanding (Qiu et al., 2020). Fine-tuning neural language models refers to the process of further training a pre-trained language model on a specific task or domain with a smaller dataset. The pre-trained language model, such as BERT or GPT, has already learned a significant amount of knowledge about natural languages from a large corpus of text. However, it may not have been trained specifically for the task at hand or on the specific domain of interest. Fine-tuning involves updating the pre-trained model's parameters to optimize its performance on the given target so it can learn more task-specific or domain-specific information. Fine-tuning large language models (LLMs) proved to be very effective and efficient to achieve higher accuracy and state of the art numbers in many downstream tasks(Xiao et al., 2020). Various techniques were suggested to ensure that the resulting models achieve optimal accuracy. One of the challenges faced during the fine-tuning of language models is overfitting. Overfitting occurs when the model performs well on the training or fine-tuning data but poorly on new, unseen data. This happens because the model has learned to fit the noise in the training data rather than capturing the underlying patterns. To address overfitting, several regularization techniques were proposed in the literature, such as weight decay and dropout. These methods help prevent the model from memorizing the training data and promote better generalization to unseen data. Additionally, achieving optimal results with fine-tuning involves hyperparameter tuning, where efforts are made to select the best set of hyperparameters for the model. Hyperparameters, such as the learning rate and number of layers, can significantly influence the model's performance and generalization capabilities. Properly tuning these hyperparameters is essential for obtaining the best possible results during fine-tuning (Mosbach et al., 2021; Yang and Ma, 2022). In this research, we pursue a different direction for fine-tuning language models by exploring a methodology that involves limiting backpropagation to a specific number of layers. This approach offers several benefits, including effectively addressing the issue of over-fitting and significantly reducing the fine-tuning time. Our primary objective is to identify the most impactful layers that contribute to achieving the best performance, and then extend this investigation to various pre-trained mod-

Proceedings of the The First Arabic Natural Language Processing Conference (ArabicNLP 2023), pages 405–410

els. The key contributions of this research are as follows:

- We explore **the impact of layer freezing** on pre-trained models with focus on application on tasks in Arabic language.
- Evaluate the effect of layer freezing on different pre-trained models **in terms of performance and speed**.
- Compare **the performance** of models using the proposed approach.
- Contrast **the time needed for fine-tuning** in both layer freezing and no-freezing settings.

The remainder of the article is structured as follows: In the next section, we provide background information on the evolution of language models and natural language processing. Subsequently, in the third section, we present our methodology, introducing the language models and tasks we will be experimenting with. Following that, we present the results and engage in a discussion in the fourth section. Finally, in the fifth section, we present our conclusions and outline the prospects for our ongoing work.

2 Background

2.1 Natural Language Processing

Natural Language Processing (NLP) is an essential branch of artificial intelligence that delves into the intricate realm of human language. Its primary objective is to empower computers with the ability to comprehend, interpret, and manipulate text and words in a manner that mirrors human understanding (Liddy, 2001). The definition of NLP covers a variety of aspects: There are several computational methods for NLP, and they essentially fall into four categories; symbolic, statistical, connectionist, and hybrid. Symbolic methods use a deep analysis of linguistic phenomena,and they are based on the explicit representation of linguistic facts using well-known knowledge representation schemes. Statistical approaches build models of linguistic phenomena using a variety of mathematical techniques and a large text corpus. The major source of evidence for these methods is observable data, with no linguistic or general knowledge added. The connectionist approach construct generalised models using examples of linguistic phenomena, and they employ also variety of representational theories. The text being analysed must come from a language that people use to communicate, and it may be in any language,and in any format oral or written.

In NLP, humans utilize various levels of language to comprehend the content of a document. These levels include Phonology (the study of speech sounds), Morphology (the study of word forms and structure), Lexical (the study of words and their meanings), Syntactic (the study of sentence structure), Semantic (the study of meaning in language), Discourse (the study of how sentences are connected and organized), and Pragmatic (the study of language use in context). The more capable an NLP system is, the more of these levels it will employ to understand and process language effectively. For instance, a sophisticated NLP system will take into account not only the words in a sentence but also their meanings, how they are arranged grammatically, and how the sentences relate to each other in a larger context. However, in practice, current NLP systems often utilize separate modules to handle different levels of language processing. These modules work together to process the language and extract meaningful information.

2.2 Techniques

Among the ground breaking techniques that changed the field of NLP was the introduction of Transformers (Vaswani et al., 2017). Its power to handle sequential data made them dominate the field in recent year. BERT (Bidirectional Encoder Representations from Transformers) is a revolutionary language model that has had a profound impact on Natural Language Processing (NLP). It is designed to understand the context of words in a sentence by considering the surrounding words on both sides, leading to a bidirectional learning process. This innovative approach allows BERT to capture deep contextual relationships and nuances in language, making it exceptionally effective in various NLP tasks. By pre-training on a large corpus of text and then fine-tuning on specific downstream tasks, BERT exhibits remarkable versatility and can be adapted to tasks like text classification, named entity recognition, question answering, and more. Its contextual embeddings have significantly improved the accuracy of language-based applications, and BERT's success has inspired numerous follow-up models that continue to push the boundaries of NLP research and application.

2.3 Freezing

Fine-tuning has become an integral component in the training process, because is less expensive in computational time than pre-training a mode. Additionally, it could solve the problem of overfitting. Limiting the number of layers "freezing" is a natural way to improve fine-tuning performance (Liu et al., 2021). For BERT model, the initial layers learn more general linguistic patterns. However, the later BERT layers learn more task-specific patterns (Clark et al., 2019; Sajjad et al., 2023).

3 Methodology

To explore the extent of the proposed method, we limit the scope of our investigation to the following pre-trained models and tasks, more models would be worth of investigating in the future work.

3.1 Pre-trained models

AraBERTv0.2 Antoun et al. (2020) trained a BERT base model using 200M sentences (77GB) of both Modern Standard Arabic (MSA) and dialectal content mainly from Twitter data. The MSA content includes Arabic Wikipedia Dumps, Arabic Corpus (El-Khair, 2016) and the Open Source International Arabic News Corpus (OSIAN) (Zeroual et al., 2019), in addition to Arabic news content.

CAMelBERT Inoue et al. (2021) created and distributed a pre-trained language model that combined Modern Standard Arabic (MSA), dialectal Arabic (DA), and classical Arabic (CA). The collection included over 167GB of text (17.3B tokens).

QARiB Abdelali et al. (2021) trained a model on a collection of 420 Million tweets and 180 Million sentences of text. The tweets contains both MSA and DA, while the text content is mostly MSA.

MARBERT Abdul-Mageed et al. (2021) created and distributed large-scale pre-trained masked language model focused on both Dialectal Arabic (DA) and Modern Standard Arabic (MSA). It was trained on a dataset of 1 billion Arabic tweets from an in-house dataset of about 6 billion tweets.

3.2 Tasks

Arabic Language Understanding Evaluation (ALUE) (Seelawi et al., 2021) provides a total of eight tasks that address a variety of Arabic dialects and NLP/NLU issues. In this paper, four tasks are used for experimental results.

Anger Detection The Affect in Tweets dataset proposed in (Mohammad et al., 2018) consists of five subtasks. We will only use the Emotion Classification task (SEC), in which a tweet is classified as anger, anticipation, contempt, fear, joy, love, optimism, pessimism, sad, surprise, and trust. We concentrate on the anger emotion, we detect if a tweet contains that emotion or not.

Text Similarity In the Semantic Question Similarity task (McCann et al., 2017), two questions are considered to be semantically similar if they have the exact same response and significance. The dataset includes question pairings and the degrees of similarity between them. There are two questions in each question pair. Each question pair's similarity score is shown as a value between 0 and 5, which was determined by human evaluations.

IDAT@FIRE2019 Irony Detection Task (FID) The purpose of this task is to detect irony in Arabic tweets (Ghanem et al., 2019). Each tweet is labeled with a "1" when it contains irony or sarcasm. Otherwise, a label of "0" is assigned.

MADAR Shared Task Subtask 1 (Dialect Detection) The Multi Arabic Dialect Applications and Resources (MADAR)[1]. The first MADAR's subtask was a parallel corpus of 25 Arabic city dialects in the field of travel (Bouamor et al.). The MSA is given a 26th label. We focus only on two classes; the dialects of Algiers and Amman.

OSACT4 Shared Task-A: offensive The task (Mubarak et al., 2020) was designed for the purpose of detecting offensive speech in Arabic tweets. Each tweet is labeled with a "1" when it contains offensive speech. Otherwise, "0".

OSACT4 Shared Task-B: hate speech detection The purpose of this task is to detect hate speech in Arabic tweets (Mubarak et al., 2020). Each tweet is labeled with a "0" when it contains hate speech. Otherwise, a label of "1" is assigned.

Cross-lingual Sentence Representations The goal of this task is to use a dataset containing 7,500 pairs of sentences to classify them into one of the following categories: "commitment," "ambivalence," or "neutral." (Conneau et al., 2018)

4 Results and Discusson

4.1 Optimal Settings

We investigate the optimal parameters for layer freezing. To identify the best configuration, we perform a comprehensive grid search, exploring all possible combinations. Although this approach

[1]https://sites.google.com/nyu.edu/madar/

may seem exhaustive, it allows us to evaluate all layers efficiently. For this step, we use the MADAR dataset, chosen as an exemplary task due to its large size and multitude of labels. Specifically, this is a multi-class classification with 26 class labels, each representing the dialect associated with different city. We explore a combination of freezing both n top and m bottom layers while recording the performance at each combination. Figure 1 represents the results of the exploration.The evidence shows that unfreezing all layers leads to achieving the state-of-the-art (SOTA) performance. However, even by freezing up to 3 layers from the bottom and four layers from the top, the model still attains performance levels very close to the best performance. Figure 1 shows the F1 results of freezing all combinations on MADAR task.

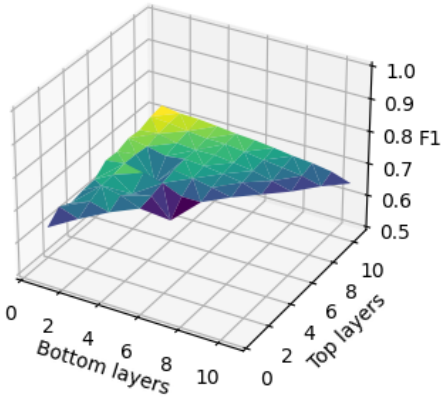

Figure 1: Layers freezing results on MADAR.

4.2 Layer Freezing

Given the promising results obtained from the previous experiments. We further expand our experimentation to benchmark an actual four downstream tasks. Appendix Tables 1, 2, 3 4, 5, 6 and 7 show the performance of training and evaluation of BERT models on different tasks, in terms of F1 and training time. While the performance loss in all the seven tasks rarely surpassed 6%, the gain in time reached up to 50%. In few instances, the performance improved further see MARBERT models results in table 6 and 7. The results summarized in Figure 2 shows clearly the large difference between the gain in runtime versus the performance loss.

4.3 Discussion

This research focuses on optimizing the computation time required for fine-tuning large language models, considering the substantial impact of computation costs across various applications and disciplines. To achieve this objective, we introduced the "layers freezing" approach, which effectively reduced the runtime needed for fine-tuning. Through our experiments, we observed remarkable results, demonstrating a significant reduction of up to 50% in fine-tuning time (See Appendix Table 4) compared to traditional approaches. This substantial improvement in efficiency offers new possibilities for researchers, developers, and organizations, enabling them to deploy and fine-tune large language models more rapidly and effectively.

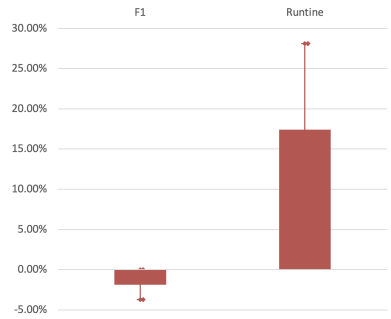

Figure 2: F1 and Runtime averages cross tasks.

5 Conclusion and Future Work

Our results suggest that freezing limited numbers of layers from the bottom in combination with top layers provide an optimal performance. It successfully addressed the challenge of time-consuming fine-tuning for large language models. This indicate that the perturbation from the fine-tuning can be controlled best using this approach; further, the approach might generalized better for out of domain data, as it keeps all the knowledge learnt during the pre-training. By introducing the layers freezing, we were able to achieve impressive time savings that reached up to 50% of time required for fine-tuning compared to conventional methods. This achievement in computation time optimization adds to the major advancement in the field of NLP and deep learning in general. It not only empowers researchers to conduct experiments and iterate more swiftly but also enhances the practicality of implementing large language models in real-world applications. For future work, we plan to expand this research to cover more tasks to ground these findings. More models with different architecture will be needed as well as applications in other languages. In other direction, we plan to explore the impact of the approach on generalization to out of domain and unseen data. Such explorations will validate the approach and demonstrate its merits.

References

Ahmed Abdelali, Sabit Hassan, Hamdy Mubarak, Kareem Darwish, and Younes Samih. 2021. Pre-training bert on arabic tweets: Practical considerations. *arXiv preprint arXiv:2102.10684*.

Muhammad Abdul-Mageed, AbdelRahim Elmadany, and El Moatez Billah Nagoudi. 2021. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In *Proceedings of the 59th Annual Meeting of the ACL and the 11th IJCNL Processing*, Online.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. Arabert: Transformer-based model for arabic language understanding. *arXiv preprint arXiv:2003.00104*.

Houda Bouamor, Sabit Hassan, and Nizar Habash. The MADAR shared task on arabic fine-grained dialect identification. In *Proceedings of the 4th Arabic Natural Language Processing Workshop*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

Alexis Conneau, Guillaume Lample, Ruty Rinott, Adina Williams, Samuel R Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Ibrahim Abu El-Khair. 2016. Abu el-khair corpus: A modern standard arabic corpus. *International Journal of Recent Trends in Engineering & Research (IJRTER)*, 2(11):5–13.

Bilal Ghanem, Jihen Karoui, Farah Benamara, Véronique Moriceau, and Paolo Rosso. 2019. Idat at fire2019: Overview of the track on irony detection in arabic tweets. In *Proceedings of the 11th Annual Meeting of the FIRE*, pages 10–13.

Go Inoue, Bashar Alhafni, Nurpeiis Baimukan, Houda Bouamor, and Nizar Habash. 2021. The interplay of variant, size, and task type in arabic pre-trained language models. *arXiv preprint arXiv:2103.06678*.

Elizabeth D Liddy. 2001. Natural language processing.

Yuhan Liu, Saurabh Agarwal, and Shivaram Venkataraman. 2021. Autofreeze: Automatically freezing model blocks to accelerate fine-tuning. *arXiv preprint arXiv:2102.01386*.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. *Adv. in neural information processing systems*, 30.

Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 1–17.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines. In *9th International Conference on Learning Representations, ICLR 2021*. OpenReview.net.

Hamdy Mubarak, Kareem Darwish, Walid Magdy, Tamer Elsayed, and Hend Al-Khalifa. 2020. Overview of osact4 arabic offensive language detection shared task. In *Proceedings of the 4th Workshop on OSACT, with a Shared Task on Offensive Language Detection*, pages 48–52.

XiPeng Qiu, TianXiang Sun, YiGe Xu, YunFan Shao, Ning Dai, and XuanJing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. 2023. On the effect of dropping layers of pre-trained transformer models. *Computer Speech & Language*, 77:101429.

Haitham Seelawi, Ibraheem Tuffaha, Mahmoud Gzawi, Wael Farhan, Bashar Talafha, Riham Badawi, Zyad Sober, Oday Al-Dweik, Abed Alhakim Freihat, and Hussein Al-Natsheh. 2021. Alue: Arabic language understanding evaluation. In *Proceedings of the Sixth WANLP*, pages 173–184.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Dongling Xiao, Han Zhang, Yukun Li, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-gen: An enhanced multi-flow pre-training and fine-tuning framework for natural language generation. *arXiv preprint arXiv:2001.11314*.

Chenghao Yang and Xuezhe Ma. 2022. Improving stability of fine-tuning pretrained language models via component-wise gradient norm clipping. In *Proceedings of the 2022 Conference on EMNLP*.

Imad Zeroual, Dirk Goldhahn, Thomas Eckart, and Abdelhak Lakhouaja. 2019. Osian: Open source international arabic news corpus-preparation and integration into the clarin-infrastructure. In *Proceedings of the fourth arabic natural language processing workshop*, pages 175–182.

A Appendix A

Detailed results for the selected tasks from ALUE.

Table 1: Anger Detection

	AraBERT		CAMeLBERT		QARiB		MARBERT	
	F1	Runtime	F1	Runtime	F1	Runtime	F1	Runtime
No Freeze	0.711	21.570	0.752	19.615	0.829	18.516	0.825	21.94
Freeze	0.648	16.389	0.756	14.314	0.814	13.702	0.831	16.80
Δ	-8.86%	24.02%	0.53%	27.02%	-1.81%	26.00%	0.73%	23.43%

Table 2: Question to Question Semantic Similarity (Shared Task 8)

	AraBERT		CAMeLBERT		QARiB		MARBERT	
	F1	Runtime	F1	Runtime	F1	Runtime	F1	Runtime
No Freeze	0.548	124.112	0.580	101.875	0.577	120.702	0.591	106.683
Freeze	0.580	101.269	0.581	88.164	0.582	86.650	0.597	97.987
Δ	5.84%	18.41%	0.17%	13.46%	0.87%	28.21%	1.02%	8.15%

Table 3: Irony Detection

	AraBERT		CAMeLBERT		QARiB		MARBERT	
	F1	Runtime	F1	Runtime	F1	Runtime	F1	Runtime
No Freeze	0.742	48.135	0.788	37.100	0.839	36.152	0.828	35.689
Freeze	0.786	38.107	0.768	28.242	0.836	27.824	0.835	27.73
Δ	5.93%	20.83%	-2.54%	23.88%	-0.36%	23.04%	0.84%	22.30%

Table 4: MADAR Shared Task Subtask 1 (Dialect Detection)

	AraBERT		CAMeLBERT		QARiB		MARBERT	
	F1	Runtine	F1	Runtine	F1	Runtine	F1	Runtine
No Freeze	0.670	1453.080	0.707	1289.394	0.700	1298.000	0.696	156.771
Freeze	0.633	668.153	0.690	1010.240	0.687	1020.360	0.695	159.179
Δ	-5.52%	54.02%	-2.40%	21.65%	-1.86%	21.39%	-0.14%	-1.54%

Table 5: Offensive Speech Detection

	AraBERT		CAMeLBERT		QARiB		MARBERT	
	F1	Runtime	F1	Runtime	F1	Runtime	F1	Runtime
No Freeze	0.974	136.949	0.974	126.627	0.979	119.450	0.974	119.09
Freeze	0.976	108.939	0.976	100.423	0.982	94.322	0.980	94.99
Δ	0.20%	20.45%	0.20%	20.69%	0.30%	21.04%	0.62%	20.24%

Table 6: Hate Speech Detection

	AraBERT		CAMeLBERT		QARiB		MARBERT	
	F1	Runtime	F1	Runtime	F1	Runtime	F1	Runtime
No Freeze	0.770	137.422	0.746	126.999	0.856	119.492	0.834	119.432
Freeze	0.767	109.245	0.759	100.671	0.847	94.768	0.854	95.17
Δ	-0.39%	20.50%	1.74%	20.73%	-1.05%	20.69%	2.40%	20.31%

Table 7: Cross-lingual Sentence Representations

	AraBERT		CAMeLBERT		QARiB		MARBERT	
	F1	Runtime	F1	Runtime	F1	Runtime	F1	Runtime
No Freeze	0.525	98.101	0.599	91.603	0.521	92.475	0.448	94.110
Freeze	0.494	91.738	0.571	82.284	0.505	90.435	0.547	88.919
Δ	-5.90%	6.49%	-4.67%	10.17%	-3.07%	2.21%	22.10%	5.52%