HyperAdapt: Simple High-Rank Adaptation

Abel Gurung
Purdue University
gurung1@purdue.edu

Joseph Campbell
Purdue University
joecamp@purdue.edu

Abstract

Foundation models achieve strong general performance on a wide variety of tasks, but fine-tuning is often necessary for tasks requiring specialized outputs, constraints, or data. However fine-tuning the entire model can be computationally prohibitive due to the large number of parameters. In this paper, we introduce HyperAdapt, a parameter-efficient fine-tuning method that significantly reduces the number of trainable parameters compared to state-of-the-art methods like LoRA. Specifically, HyperAdapt fine-tunes a pre-trained weight matrix by applying row-wise and column-wise scaling via diagonal matrices, requiring only $n + m$ trainable parameters for an $n \times m$ matrix. Empirically, we show that HyperAdapt achieves performance comparable to full fine-tuning and existing parameter-efficient methods on widely-used reasoning and arithmetic benchmarks with significantly fewer trainable parameters.

1 Introduction

Large-scale foundation models have shown strong capabilities across tasks such as natural language understanding [10, 33, 4], mathematical reasoning [8], and multimodal learning [1, 32]. However, adapting them to domain-specific tasks often requires fine-tuning, which is computationally expensive. Parameter-efficient fine-tuning (PEFT) addresses this challenge by updating only a small subset of parameters. LoRA [17], a widely used PEFT method, constrains updates to low-rank matrices but grows costly as rank increases. Recent high-rank methods reduce parameter costs, but many rely on large non-trainable matrices [25, 21] or incur expensive forward passes [27], limiting their efficiency.

In this work, we propose HyperAdapt, a novel parameter-efficient fine-tuning method that leverages high-rank update to adapt to downstream tasks efficiently. Specifically, our approach fine-tunes a pre-trained weight matrix by left- and right-multiplying it with trainable diagonal matrices, resulting in row- and column-wise scaling. This adjusts the model's sensitivity to different input features and its emphasis on certain output representations, achieving performance comparable to full fine-tuning and state-of-the-art PEFT methods (see Fig. 1).

Our contributions include:

- **Improved parameter efficiency:** HyperAdapt only uses $n + m$ trainable parameters.
- **Competitive performance:** HyperAdapt achieves performance comparable to full fine-tuning and existing PEFT methods such as LoRA across widely-used NLP benchmarks.
- **High-Rank adaptation:** We provide a theoretical upper bound on HyperAdapt's update rank (Lemma A.1) and validate it empirically (Sec. 5).

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Continual and Compatible Foundation Model Updates (CCFM).

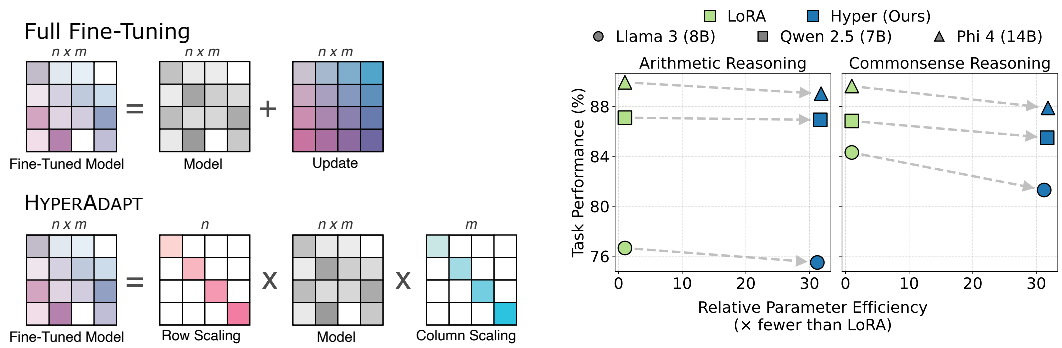

Figure 1: (Left) Our proposed method, HYPERADAPT, fine-tunes a model by learning row-wise and column-wise diagonal matrices. Unlike full fine-tuning, which requires $n \times m$ trainable parameters, our method yields comparable performance yet only requires $n + m$ trainable parameters. Grayscale values represent frozen parameters, while colored values represent trainable parameters. (Right) Our method achieves similar performance to LoRA across common benchmarks while using up to significantly fewer trainable parameters

.

2 Preliminaries

Let f_θ be a pre-trained model with parameters θ, mapping input x to output y. Fine-tuning adapts the model by updating parameters to $f_{\theta'}$ with $\theta' = \theta + \Delta\theta$, where $\Delta\theta$ is task-specific. For large models, updating all parameters is impractical, and prior work shows that modifying only a small subset can yield strong performance.

The intrinsic dimension hypothesis [22, 2] states that solving a specific task to a desired accuracy generally requires adjusting only a minimal subset of parameters within a low-dimensional subspace of the full parameter space. Hu et al. [17] applies this idea at the level of weight matrices. For a pre-trained weight matrix $\mathrm{W}_0 \in \mathbb{R}^{n \times m}$, the adapted weights are

$$\mathrm{W}' = \mathrm{W}_0 + \Delta\mathrm{W}. \tag{1}$$

LoRA parameterizes $\Delta\mathrm{W}$ as BA, with $\mathrm{B} \in \mathbb{R}^{n \times r}$ and $\mathrm{A} \in \mathbb{R}^{r \times m}$ for small rank $r \ll \min(n, m)$. While effective, its parameter cost grows linearly with r, and higher ranks are often needed for stronger adaptation.

3 High-Rank Parameter-Efficient Fine-Tuning

It is a well-known phenomenon that over-parameterization of neural networks facilitates easier optimization [12]. Empirical scaling laws[19, 15] show that increasing parameters reliably decreases loss, suggesting that larger parameter spaces provide models with more flexibility during training. At the matrix level, we hypothesize that this flexibility is related to the rank of the update–the number of independent directions modified during training. In this view, update expressivity grows with rank, but achieving a high rank update typically requires more trainable parameters when learning these directions from scratch. The key observation underlying our work is that the pre-trained weight matrices are already (near) full-rank and encode many useful directions from pre-training. For efficient fine-tuning, rather than learning new update directions as in prior PEFT methods, we can simply exploit the existing directions. We propose a method to induce constrained high-rank updates: scale the rows and columns of an $n \times m$ pre-trained matrix, which requires only $n + m$ trainable scalars to reweight existing directions. This achieves high-rank updates across transformer modules (see Sec. 5), improving adaptability with minimal trainable parameters.

3.1 HYPERADAPT

In this work, we introduce HYPERADAPT, a parameter-efficient fine-tuning method that achieves high-rank transformations by constraining the form of the update rather than its rank. Given a pre-trained weight matrix $\mathrm{W}_0 \in \mathbb{R}^{n\times m}$, we define the fine-tuned $\Delta\mathrm{W}$ to be:

$$\Delta\mathrm{W} = \mathrm{A}\mathrm{W}_0\mathrm{B} - \mathrm{W}_0 \tag{2}$$

where $\mathrm{A} \in \mathbb{R}^{n\times n}$ and $\mathrm{B} \in \mathbb{R}^{m\times m}$ are diagonal matrices. Substituting $\Delta\mathrm{W}$ into Eq. 1 yields:

$$\mathrm{W}' = \mathrm{W}_0 + \mathrm{A}\mathrm{W}_0\mathrm{B} - \mathrm{W}_0 = \mathrm{A}\mathrm{W}_0\mathrm{B}.$$

The resulting fine-tuned weight matrix W' is then the product of the original weight matrix W_0 with two diagonal scaling matrices, A and B. This transformation can be intuitively interpreted as adjusting the latent representations most relevant to the downstream task. Representing W' using diagonal matrices has two primary benefits: Only $n + m$ trainable parameters are required and diagonal matrix multiplication can be calculated using element-wise multiplications rather than full matrix multiplications, which is significantly faster. HYPERADAPT is effective despite using only a minimal number of trainable parameters because it produces high-rank updates without explicitly constraining the rank. This enables HYPERADAPT to efficiently adapt pre-trained models to downstream tasks. The rank of $\Delta\mathrm{W}$ in (2) is upper-bounded by $\min(2 \cdot \mathrm{rank}(\mathrm{W}_0), n, m)$

Lemma 3.1. *Let* $\mathrm{W}_0 \in \mathbb{R}^{n\times m}$ *and let* $\mathrm{A} \in \mathbb{R}^{n\times n}$, $\mathrm{B} \in \mathbb{R}^{m\times m}$ *be diagonal matrices. Define* $\Delta\mathrm{W} := \mathrm{A}\,\mathrm{W}_0\,\mathrm{B} - \mathrm{W}_0$. *Then* $\mathrm{rank}(\Delta\mathrm{W}) \le \min\{2 \cdot \mathrm{rank}(\mathrm{W}_0),\, n,\, m\}$.

The full proof is given in the Appendix (Lemma A.1). This means that the update matrix induced by HYPERADAPT can achieve a high-rank and is upper-bounded only by the rank of W_0. While this transformation cannot increase the rank of W_0, as it involves multiplication with diagonal matrices, it nonetheless utilizes the full rank potential of W_0 to adapt to the downstream objective. Additionally, our empirical results in Sec. 5 show that updates are consistently high-rank in practice. Furthermore, similar to prior works, our method introduces **no additional test-time latency** as the modified weights can be precomputed before deployment.

3.2 Related Work

Low-Rank Adaptation and Variants: LoRA [17], discussed in Sec. 2, is one of the most widely adopted methods for fine-tuning large pre-trained models. A recent extension, DoRA [26], decomposes the pre-trained weight matrix into separate magnitude and direction components which are then fine-tuned separately.

High-Rank Adaptation: Recent methods such as SVFT [25] and VeRA [21] achieve high-rank updates with few trainable parameters but rely on large non-trainable matrices, making them memory-inefficient. HYPERADAPT avoids this overhead by not introducing auxiliary parameters. An alternative, BoFT [27], represents dense orthogonal matrices through butterfly factorization, but its series of matrix multiplications incurs high activation memory costs.

4 Empirical Experiments

To evaluate the effectiveness of our method, we aim to answer two research questions: 1) How does the downstream task performance of models fine-tuned with our method compare to full fine-tuning and existing PEFT methods? 2) How does the number of trainable parameters required by our method compare to these other methods?

To answer these questions, we fine-tune four LLMs of varying sizes: RoBERTa-Large (355M) [28], Llama 3 (8B) [13], Qwen 2.5 (7B) [32], and Phi 4 (14B) [1]. And evaluate them on a wide range of NLP tasks spanning three benchmarks: GLUE [40], Commonsense Reasoning, and Arithmetic Reasoning.

4.1 GLUE Benchmark

We first evaluate our proposed method on the General Language Understanding Evaluation (GLUE) benchmark [40] using RoBERTa-Large. GLUE is a widely used collection of natural language understanding tasks designed to test various aspects of language comprehension and reasoning. To ensure a fair comparison, we follow the same experimental setup as Hu et al. [17] and use a sequence length of 128 and the same number of training epochs for each task. We exclude QQP and MNLI from our benchmark due to their high computational cost.

As shown in Table 1, HYPERADAPT achieves performance comparable to LoRA while using **4 times fewer** trainable parameters. Moreover, it matches the performance of full fine-tuning despite requiring over **1700 times** fewer parameters. Similarly, VeRA also demonstrates strong performance with minimal trainable parameters; however, it introduces 0.5M additional non-trainable parameters during fine-tuning. In contrast, HYPERADAPT achieves 86.0 average with 0.1M trainable parameters without additional non-trainable matrices, while achieving performance comparable to both LoRA and full fine-tuning.

Table 1: GLUE task performance results for RoBERTa-Large. We report Matthew's correlation for CoLA, Pearson correlation for STS-B, and accuracy for other tasks; higher is better. The values for Full FT and LoRA are taken from prior work [17]. For VeRA, we also report additional non-trainable parameters with red text.

Method	# Params	SST-2	MRPC	CoLA	QNLI	RTE	STS-B	Avg.
Full FT	355.0M	96.4	90.9	68.0	94.7	86.6	92.4	88.2
LoRA	0.8M	96.2 ± 0.5	90.2 ± 1.0	68.2 ± 1.9	94.8 ± 0.3	85.2 ± 1.1	92.3 ± 0.5	87.8
DoRA	0.8M	96.0 ± 0.2	89.3 ± 0.6	65.8 ± 0.3	94.6 ± 0.1	83.5 ± 1.1	91.0 ± 0.4	86.7
VeRA	0.06M | 0.5M	95.8 ± 0.3	89.4 ± 0.5	65.3 ± 1.5	94.1 ± 0.2	79.3 ± 3.4	89.5 ± 0.8	85.6
Hyper (Ours)	**0.1M**	96.2 ± 0.2	89.8 ± 0.3	64.9 ± 0.7	93.8 ± 0.1	80.8 ± 1.3	90.2 ± 0.3	86.0

4.2 Arithmetic Reasoning Benchmark

To evaluate the impact of HYPERADAPT on arithmetic reasoning, we compare our method against widely used PEFT baselines. Following the experimental settings in Hu et al. [18], we fine-tune each model on the Math10K dataset[18], which comprises training examples from GSM8K[8] and AQuA [24]. Models are then evaluated on six downstream arithmetic reasoning tasks, refer to Sec. B.2 for details. Because Math10K includes only two of these sub-tasks, this benchmark also assesses generalization to out-of-distribution arithmetic problems.

Table 2: Arithmetic Reasoning results. We report accuracy for all tasks. For all tasks, higher value is better. For VeRA, we also report additional non-trainable parameters with red text.

Model	Method	# Params (%)	AddSub	SingleEq	GSM8K	AQuA	MultiArith	SVAMP	Avg
Llama-3-8b	LoRA$_{r=1}$	0.03	70.1	87.6	55.3	37.4	86.5	58.9	66.0
	LoRA	1.03	89.6	95.9	64.4	40.9	94.2	74.9	76.7
	DoRA	1.05	90.1	95.9	64.3	41.3	93.2	77.4	77.0
	VeRA	0.02 | 0.37	73.7	85.6	55.0	41.3	85.0	59.5	66.7
	Hyper (Ours)	**0.03**	86.3	96.7	61.9	44.1	94.2	69.8	75.5
Qwen-2.5-7B	LoRA$_{r=1}$	0.03	93.4	98.4	82.6	63.4	97.5	86.5	87.0
	LoRA	1.05	92.7	98.2	79.8	70.1	98.2	83.6	87.1
	DoRA	1.07	90.9	98.6	79.7	67.7	98.7	83.4	86.5
	VeRA	0.02 | 0.51	94.2	98.6	82.3	66.9	98.7	88.1	88.1
	Hyper (Ours)	**0.03**	92.7	98.6	79.9	68.1	98.8	83.4	86.9
Phi-4-14B	LoRA$_{r=1}$	0.02	93.7	98.0	86.8	68.5	98.3	87.7	88.8
	LoRA	0.75	95.2	99.0	87.5	69.3	98.7	89.9	89.9
	DoRA	0.77	95.2	99.2	87.0	69.7	98.5	89.9	89.9
	VeRA	0.02 | 0.75	93.4	96.6	84.7	73.2	95.8	87.9	88.6
	Hyper (Ours)	**0.02**	93.9	99.0	86.5	66.9	98.3	89.4	89.0

HYPERADAPT achieves comparable performance with most methods while using fewer parameters. Among the models compared, performance with Llama-3-8B [13] stands out for HYPERADAPT when comparing against VeRA and LoRA$_{r=1}$, showing robustness across models.

4.3 Commonsense Reasoning Benchmark

For commonsense reasoning benchmarks, we first fine-tuned the models on the Commonsense170K dataset [18], an aggregated dataset consisting of eight sub-tasks. These tasks evaluate a model's ability to reason about everyday scenarios and implicit world knowledge that may not be directly stated in the text. The results are shown in Table 3. HYPERADAPT again yields competitive results across all model sizes while using fewer trainable parameters.

Table 3: Commonsense Reasoning results. We report accuracy for all tasks. For all tasks, higher value is better. For VeRA, we also report additional non-trainable parameters with red text.

Model	Method	# Params (%)	ARC-c	ARC-e	WinoGrande	SIQA	OBQA	BoolQ	PIQA	HellaSwag	Avg
Llama-3-8B	$\text{LoRA}_{r=1}$	0.03	76.5	89.7	77.4	75.2	78.8	60.8	84.9	89.9	79.1
	LoRA	1.03	79.4	90.3	83.0	79.8	86.0	72.5	87.9	95.5	84.3
	DoRA	1.05	79.6	90.8	83.8	80.1	84.2	73.2	87.9	95.5	84.4
	VeRA	0.02 ∣ 0.37	74.4	89.0	74.0	73.3	78.8	61.6	84.0	84.5	77.5
	Hyper (Ours)	**0.03**	78.2	89.3	79.4	76.4	80.6	67.9	86.3	92.4	81.3
Qwen-2.5-7B	$\text{LoRA}_{r=1}$	0.03	88.1	95.8	76.0	78.9	87.4	70.1	88.0	92.8	84.6
	LoRA	1.05	88.6	95.8	83.5	80.2	89.2	72.7	89.8	94.9	86.8
	DoRA	1.07	88.5	95.9	82.4	79.8	89.6	72.8	89.6	94.6	86.7
	VeRA	0.02 ∣ 0.51	88.0	95.4	74.8	78.6	87.8	69.2	88.4	92.6	84.3
	Hyper (Ours)	**0.03**	88.3	95.3	80.1	78.8	90.4	68.6	88.7	93.7	85.5
Phi-4-14B	$\text{LoRA}_{r=1}$	0.02	92.8	97.9	83.5	79.6	91.6	73.4	90.5	94.0	87.9
	LoRA	0.75	93.5	98.0	87.5	81.9	93.8	74.6	92.6	95.1	89.6
	DoRA	0.77	93.9	98.2	87.3	82.0	94.8	75.1	92.4	89.9	89.2
	VeRA	0.02 ∣ 0.75	92.6	97.8	58.4	79.3	90.8	69.9	83.6	93.6	83.2
	Hyper (Ours)	**0.02**	93.3	97.7	83.1	81.0	91.2	69.7	92.6	94.5	87.9

Table 3 summarizes the results of different methods on commonsense reasoning benchmarks. HYPERADAPT achieves competitive performance despite using significantly fewer trainable parameters compared to LoRA and DoRA with Llama3-8B and Qwen2.5-7B.

4.4 Fine-Tuning With Reasoning Traces

To further evaluate the performance of HYPERADAPT in low-data and long-context settings, we fine-tune Qwen-2.5-7B on the S1 dataset [30], which contains $1,000$ high-quality reasoning traces and solutions collected from Gemini's "thinking" model [14]. Following Muennighoff et al. [30] setup, we train only on the reasoning traces and solutions, but not the question itself, using the Transformer Reinforcement Learning (TRL) library [39]. We set the cut-off length for a given sequence to 16K tokens to assess robustness across longer sequences. The fine-tuned models are evaluated on GSM8K [8] and MATH500 [23].

Table 4: Performance of fine-tuned reasoning models over math benchmarks. We report accuracy for all tasks (higher is better). For VeRA, we also report additional non-trainable parameters with red text.

Method	# Params (%)	GSM8K	MATH500
$\text{LoRA}_{r=1}$	0.03	75.7	62.6
LoRA	1.05	88.8	63.6
DoRA	1.07	88.9	65.0
VeRA	0.02 ∣ 0.51	80.3	60.6
Hyper	0.03	89.0	64.0

As shown in Table 4, HYPERADAPT attains 89.0 on GSM8K and 64.0 on MATH500, effectively matching low-rank baselines with an order of magnitude fewer parameters. Additionally, with the same number of trainable parameters set as HYPERADAPT, $\text{LoRA}_{r=1}$, substantially underperforms compared to HYPERADAPT, showing that naively shrinking rank is not an adequate substitute for properly fine-tuning models in such constrained trainable parameter settings.

5 Rank Analysis:

To quantify how many orthogonal directions are utilized during fine-tuning, we analyze the empirical rank of the weight update $\Delta\mathrm{W}$. Specifically, we compute the difference between the fine-tuned and pre-trained weight matrices: $\Delta\mathrm{W} = \mathrm{W}' - \mathrm{W}_0$, where W_0 is the original pre-trained weight matrix and W' is the fine-tuned weight matrix. We then compute the singular value decomposition (SVD) of $\Delta\mathrm{W}$ to obtain its singular values Σ. The number of non-negligible (i.e., significantly non-zero) singular values indicates the empirical rank of the update. Taking numerical precision into account, we only consider $\left\{\sigma_i \in \Sigma \;\middle|\; \sigma_i \geq 10^{-2}\right\}$ to be non-trivial. Finally, we normalize this empirical rank by $\mathrm{rank}(\mathrm{W}_0)$, yielding the normalized rank of the update.

We compute the normalized rank for Qwen-2.5-7B after fine-tuning on Commonsense170K [18]. Fig. 2 empirically supports our theoretical claim that HYPERADAPT induces high-rank update. As seen in the figure, most of the modules use almost all the available orthogonal directions.

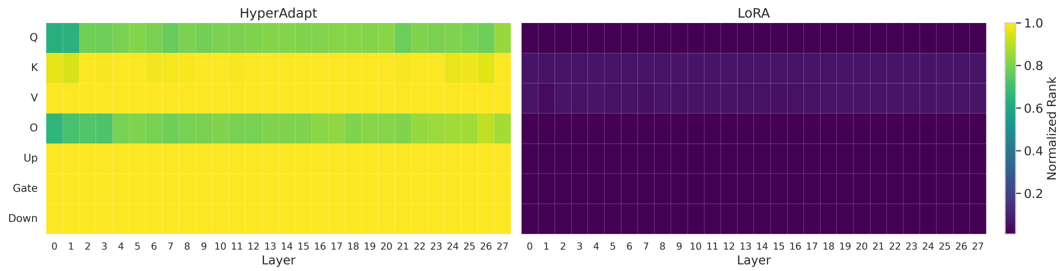

Figure 2: Normalized update rank across all layers of Qwen-2.5-7B after fine-tuning on Commonsense170K. HYPERADAPT produces high-rank update across most modules effectively utilizing a large fraction of available orthogonal directions.

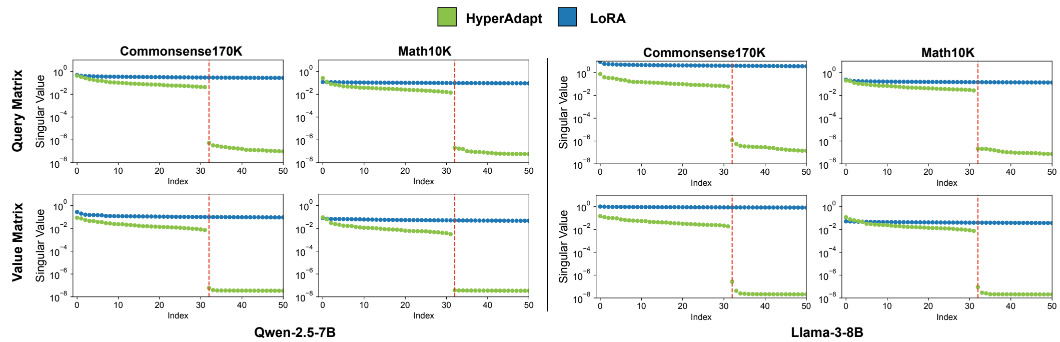

Figure 3: Singular-value spectra of the update matrix $\Delta\mathrm{W}$ as given by HYPERADAPT and LoRA for Qwen-2.5-7B and Llama-3-8B. We visualize the first 50 singular values of the update matrix in log scale; values above 1×10^{-2} are considered to be non-negligible and contribute to the update's rank. The red dashed line indicates the rank r of LoRA, showing that all values beyond this are negligible. In contrast, HYPERADAPT exhibits a slower decay, reflecting a higher-rank update. The top row corresponds to the Query matrix $\Delta\mathrm{W}_Q$ of the 13th layer, and the bottom row corresponds to the Value matrix $\Delta\mathrm{W}_V$ of the 13th layer.

Singular Value Trend. Additionally, we analyze the spectrum of the update matrices, $\Delta\mathrm{W}$, plotting the singular values for the query ($\Delta\mathrm{W}_Q$) and value ($\Delta\mathrm{W}_\mathrm{V}$) matrices. Fig. 3 shows the spectra for Qwen-2.5-7B and Llama-3-8B fine-tuned on Commonsense170K and Math10K. HYPERADAPT exhibits a slower decay of singular values across different weight matrices, models, and datasets. In contrast, LoRA shows a rapid drop in singular values after its r^{th} singular value which corresponds to its rank, as expected given its low-rank structure.

6 Conclusion

In this work, we introduced HYPERADAPT, a simple yet effective parameter-efficient fine-tuning method that leverages diagonal scaling to achieve high-rank update with minimal trainable parameters. Across multiple benchmarks, HYPERADAPT delivers performance comparable to full fine-tuning and leading PEFT methods like LoRA, while requiring orders of magnitude fewer parameters. These results highlight that high-rank adaptation can be achieved without expensive auxiliary structures or large ranks, offering a scalable and efficient alternative for adapting foundation models.

Limitations: This work focuses exclusively on Transformer-based language models; extending HYPERADAPT to other data domains and models (diffusion models) remains an open direction. HYPERADAPT also assumes that the model is pre-trained, making it an effective tool for adaptation. However, when the model is not pre-trained (random initialization), HYPERADAPT cannot bootstrap and exploit the matrix's representation, which leads to poor learning.

References

[1] Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang, and Yi Zhang. Phi-4 technical report, 2024. URL https://arxiv.org/abs/2412.08905.

[2] Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning, 2020. URL https://arxiv.org/abs/2012.13255.

[3] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language, 2019. URL https://arxiv.org/abs/1911.11641.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

[5] Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In Steven Bethard, Marine Carpuat, Marianna Apidianaki, Saif M. Mohammad, Daniel Cer, and David Jurgens, editors, *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2001. URL https://aclanthology.org/S17-2001/.

[6] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions, 2019. URL https://arxiv.org/abs/1905.10044.

[7] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

[8] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

[9] Ido Dagan, Dan Roth, Fabio Zanzotto, and Graeme Hirst. *Recognizing Textual Entailment*. Morgan & Claypool Publishers, 2012. ISBN 1598298348.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, 2019.

[11] William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005. URL https://aclanthology.org/I05-5002/.

[12] Simon S. Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks, 2019. URL https://arxiv.org/abs/1810.02054.

[13] Aaron Grattafiori et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

[14] Google. Gemini 2.0 flash thinking mode (gemini-2.0-flash-thinking-exp-1219), December 2024. URL https://cloud.google.com/vertex-ai/generative-ai/docs/thinking-mode.

[15] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

[16] Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. Learning to solve arithmetic word problems with verb categorization. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1058. URL https://aclanthology.org/D14-1058/.

[17] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

[18] Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Ka-Wei Lee. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models, 2023. URL https://arxiv.org/abs/2304.01933.

[19] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

[20] Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597, 2015. doi: 10.1162/tacl_a_00160. URL https://aclanthology.org/Q15-1042/.

[21] Dawid J. Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. Vera: Vector-based random matrix adaptation, 2024. URL https://arxiv.org/abs/2310.11454.

[22] Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes, 2018. URL https://arxiv.org/abs/1804.08838.

[23] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

[24] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation : Learning to solve and explain algebraic word problems, 2017. URL https://arxiv.org/abs/1705.04146.

[25] Vijay Lingam, Atula Tejaswi, Aditya Vavre, Aneesh Shetty, Gautham Krishna Gudur, Joydeep Ghosh, Alex Dimakis, Eunsol Choi, Aleksandar Bojchevski, and Sujay Sanghavi. Svft: Parameter-efficient fine-tuning with singular vectors, 2024. URL https://arxiv.org/abs/2405.19597.

[26] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation, 2024. URL https://arxiv.org/abs/2402.09353.

[27] Weiyang Liu, Zeju Qiu, Yao Feng, Yuliang Xiu, Yuxuan Xue, Longhui Yu, Haiwen Feng, Zhen Liu, Juyeon Heo, Songyou Peng, et al. Parameter-efficient orthogonal finetuning via butterfly factorization. *arXiv preprint arXiv:2311.06243*, 2023.

[28] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[29] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering, 2018. URL `https://arxiv.org/abs/1809.02789`.

[30] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL `https://arxiv.org/abs/2501.19393`.

[31] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems?, 2021. URL `https://arxiv.org/abs/2103.07191`.

[32] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL `https://arxiv.org/abs/2412.15115`.

[33] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI*, 2019. URL `https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf`.

[34] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, 2016. URL `https://arxiv.org/abs/1606.05250`.

[35] Subhro Roy and Dan Roth. Solving general arithmetic word problems, 2016. URL `https://arxiv.org/abs/1608.01413`.

[36] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019. URL `https://arxiv.org/abs/1907.10641`.

[37] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions, 2019. URL `https://arxiv.org/abs/1904.09728`.

[38] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL `https://aclanthology.org/D13-1170/`.

[39] Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`, 2020.

[40] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In Tal Linzen, Grzegorz Chrupała, and Afra Alishahi, editors, *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL `https://aclanthology.org/W18-5446/`.

[41] Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural network acceptability judgments, 2019. URL `https://arxiv.org/abs/1805.12471`.

[42] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019. URL `https://arxiv.org/abs/1905.07830`.

A Bound for Rank of Update Matrix

Lemma A.1. *Let* $\mathrm{W}_0 \in \mathbb{R}^{n\times m}$ *and let* $\mathrm{A} \in \mathbb{R}^{n\times n}$, $\mathrm{B} \in \mathbb{R}^{m\times m}$ *be diagonal matrices. Define* $\Delta\mathrm{W} := \mathrm{A}\,\mathrm{W}_0\,\mathrm{B} - \mathrm{W}_0$. *Then* $\mathrm{rank}(\Delta\mathrm{W}) \leq \min\{2\cdot\mathrm{rank}(\mathrm{W}_0),\, n,\, m\,\}$.

Proof. Let r be the $\mathrm{rank}(\mathrm{W}_0)$. We know that for any conformable matrix X and Y, $\mathrm{rank}(\mathrm{X}+\mathrm{Y}) \leq \mathrm{rank}(\mathrm{X}) + \mathrm{rank}(\mathrm{Y})$. Therefore:

$$\mathrm{rank}(\mathrm{AW_0B} - \mathrm{W}_0) \leq \mathrm{rank}(\mathrm{AW_0B}) + \mathrm{rank}(-\mathrm{W}_0)$$

Since we define $\Delta\mathrm{W}$ to be $\mathrm{AW_0B} - \mathrm{W}_0$, we substitute this definition:

$$\mathrm{rank}(\Delta\mathrm{W}) \leq \mathrm{rank}(\mathrm{AW_0B}) + \mathrm{rank}(-\mathrm{W}_0)$$

Here, $\mathrm{rank}(\mathrm{AW_0B}) \leq \mathrm{rank}(\mathrm{W}_0) = r$ because $\mathrm{rank}(\mathrm{XY}) \leq \min(\mathrm{rank}\,\mathrm{X}, \mathrm{rank}\,\mathrm{Y})$. Therefore, $\mathrm{rank}(\mathrm{AW_0B}) \leq \mathrm{rank}(\mathrm{W_0B}) \leq \mathrm{rank}(\mathrm{W}_0)$, and similarly $\mathrm{rank}(\mathrm{AW_0B}) \leq \mathrm{rank}(\mathrm{AW}_0) \leq \mathrm{rank}(\mathrm{W}_0)$.

$$\mathrm{rank}(\Delta\mathrm{W}) \leq r + \mathrm{rank}(-\mathrm{W}_0)$$

Additionally, $\mathrm{rank}(-\mathrm{W}_0) = \mathrm{rank}(\mathrm{W}_0) = r$:

$$\mathrm{rank}(\Delta\mathrm{W}) \leq r + r = 2r$$

Furthermore, for any matrix its rank is always upper-bounded by its dimensions so the $\mathrm{rank}(\Delta\mathrm{W}) \leq \min(2\cdot\mathrm{rank}(\mathrm{W}_0), n, m)$. ☐

B Experiments and Hyperparameters

B.1 GLUE Benchmark

The General Language Understanding Evaluation (GLUE) benchmark [40] is a collection of various natural language processing (NLP) tasks designed to evaluate the generalization capabilities of language models. It includes single-sentence classification, sentence-pair classification, and similarity tasks. We primarily use six of it's sub-tasks: CoLA (Corpus of Linguistic Acceptability) determines whether a given sentence is grammatically acceptable [41]. SST-2 (Stanford Sentiment Treebank) classifies movie reviews as positive or negative [38]. MRPC (Microsoft Research Paraphrase Corpus) identifies whether two sentences are semantically equivalent [11]. STS-B (Semantic Textual Similarity Benchmark) measures the similarity of two sentences [5]. QNLI (Question Natural Language Inference) evaluates whether a given passage contains the answer to a question [34]. RTE (Recognizing Textual Entailment) is a binary classification task for textual entailment [9].

To keep results consistent, we use the same † setup as Hu et al. [17] for RoBERTa Large. All of our GLUE experiments have the same max sequence length and epochs for each task.

B.2 Arithmetic Reasoning

Arithmetic Reasoning Benchmarks consists of six evaluation tasks. AddSub [16] tests simple addition and subtraction word problems, SingleEq [20] involves solving word problems that translate to a single algebraic equation, GSM8K [8] features multi-step grade school problems. AQuA [24] focuses on multiple-choice algebra questions, MultiArith[35] requires sequential arithmetic steps, and SVAMP [31] evaluates robustness through perturbed math problems. For fine-tuning DoRA, we use the hyperparamerters provided by Liu et al. [26].

B.3 Commonsense Reasoning

Commonsense Reasoning benchmark consists of eight evaluation tasks. Arc-challenge and Arc-easy [7] consist of science exam questions drawn from a variety of sources, Winogrande[36] evaluates pronoun resolution in challenging contexts, SocialInteractionQA (SIQA) [37] assesses social and situational reasoning, OpenBookQA (OBQA)[29] focuses on science-related multiple-choice questions, BoolQ[6] contains yes/no questions from real-world queries, PhysicalInteractionQA(PIQA) [3] tests physical commonsense, and HellaSwag [42] challenges models with grounded commonsense questions.

Table 5: The hyperparameters used for RoBERTa-Large on the GLUE benchmark.

Method	Dataset	SST-2	MRPC	CoLA	QNLI	RTE	STS-B
	Optimizer			AdamW			
	Warmup Steps			10			
	LR Schedule			Constant			
	Epochs	10	20	20	10	20	10
	Batch Size			128			
	Target Layers			Q, V			
	Max Seq. Len.			128			
HYPERADAPT	Learning Rate	3E-03	8E-03	8E-03	3E-03	8E-03	3E-03
vera	Learning Rate	3E-03	8E-03	8E-03	3E-03	8E-03	3E-03
	Rank			256			
DoRA	Learning Rate	4E-04	3E-04	2E-04	2E-04	4E-04	2E-04
	Rank			8			
	LoRA α			16			

Table 6: The hyperparameters used for the Arithmetic Reasoning Benchmark.

Method	Models	Llama-3-8b	Qwen-2.5-7B	Phi-4-14B
	Optimizer		AdamW	
	Warmup Steps		100	
	Max Grad Norm		1.0	
	LR Schedule		Cosine	
	Max Seq. Len		512	
	Batch Size		256	
	Target Layers		Q, K, V, O, Gate, Up, Down	
	Epochs		3	
HYPERADAPT	Learning Rate		3e-3	
vera	Learning Rate		3e-3	
	Rank	1024	1024	2048
BoFT	Learning Rate		3e-4	
	Block Size		4	
	Butterfly Factor		4	
$\text{LoRA}_{r=1}$	Learning Rate		1e-4	
	Rank		1	
	LoRA α		2	
	LoRA Dropout		0.05	
LoRA	Learning Rate		1e-4	
	Rank		32	
	LoRA α		64	
	LoRA Dropout		0.05	
DoRA	Learning Rate		1e-4	
	Rank		32	
	LoRA α		64	
	LoRA Dropout		0.05	

B.4 Fine-Tuning With Reasoning Traces

We use the following hyperparameters for fine-tuning for the over reasoning traces. The learning rates are based on the suggestions from the original paper.

Table 7: The hyperparameters used for the Common Sense Reasoning Benchmark.

Method	Models	Llama-3-8b	Qwen-2.5-7B	Phi-4-14B
	Optimizer		AdamW	
	Warmup Steps		100	
	Max Grad Norm		1.0	
	LR Schedule		Cosine	
	Max Seq. Len		256	
	Batch Size		256	
	Target Layers		Q, K, V, O, Gate, Up, Down	
	Epochs		2	
HyperAdapt	Learning Rate		3e-3	
vera	Learning Rate		3e-3	
	Rank	1024	1024	2048
$\text{LoRA}_{r=1}$	Learning Rate		1e-4	
	Rank		1	
	LoRA α		2	
	LoRA Dropout		0.05	
LoRA	Learning Rate		1e-4	
	Rank		32	
	LoRA α		64	
	LoRA Dropout		0.05	
DoRA	Learning Rate		1e-4	
	Rank		32	
	LoRA α		64	
	LoRA Dropout		0.05	

Table 8: The hyperparameters used for the Math Benchmark.

Method	Models	Qwen-2.5-7B
	Optimizer	AdamW
	Warmup Steps	10
	Max Grad Norm	1.0
	LR Schedule	Cosine
	Max Seq. Len	16384
	Batch Size	64
	Target Layers	Q, K, V, O, Gate, Up, Down
	Epochs	5
HyperAdapt	Learning Rate	3e-3
vera	Learning Rate	3e-3
	Rank	1024
$\text{LoRA}_{r=1}$	Learning Rate	1e-4
	Rank	1
	LoRA α	2
	LoRA Dropout	0.05
LoRA	Learning Rate	1e-4
	Rank	32
	LoRA α	64
	LoRA Dropout	0.05
DoRA	Learning Rate	1e-4
	Rank	32
	LoRA α	64
	LoRA Dropout	0.05

B.5 Decoding Hyperparameters

For commonsense reasoning, we generate at most 32 new tokens. For Arithmetic reasoning, we generate at most 512 new tokens. For math benchmark after fine-tuning on reasoning traces, we generate at most 1024 new tokens.

Table 9: Decoding hyperparameters used for text generation.

Parameter	Value
Temperature	0.05
Top-p	0.40
Top-k	40

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In this work, we show that for a given pre-trained weight matrix fine-tune it using diagonal matrices and empirically show that we efficiently achieve comparable performance with other PEFT methods with lower trainable parameter count.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, we discuss that the focus on this paper only was limited to large language models and the impact of its application in other domains remain open.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.

- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Yes, we provide proof on why our method is high-rank with a tight upper bound on the rank of the update.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Yes, we provide sufficient details in the main paper regarding the datasets, methods, and procedures to reproduce experiments. We provide specific details regarding hyperparameters and model configurations in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example

(a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will open-source our code and publicly release it upon acceptance.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Yes, along with the specific model and dataset used for experiments, we also provide all the hyperparameters used for the experiment.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Standard errors and statistical significance are not reported due to the immense computational resources required to fine-tune large language models.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide details regarding the amount of compute used for experiments in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Our method is strictly a parameter-efficient optimization method for neural networks, and in itself does not provide a direct path to positive or negative societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: NA

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

Justification: While we use open-source models and code in our experiments, we cite all relevant papers and give proper attribution.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We release our anonymized code as part of the supplementary material, and include documentation and instructions along with it.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: NA

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: NA

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.