POST-LORA RESTORATION: UTILIZING TRANSFERABILITY OF LOW-RANK ADAPTER IN QUANTIZED FOUNDATION MODELS

Yuto Kanda, Kenji Hatano
Graduate School of Culture and Information Science
Doshisha University
{kanda@mil,khatano@mail}.doshisha.ac.jp

ABSTRACT

In this study, we consider the transferability of LoRA adapters across quantized foundation models. Specifically, we investigate whether LoRA adapters trained on a low-bit-width foundation model can still perform effectively when merged into a higher-bit-width foundation model. By leveraging this transferability, it becomes possible to construct models with performance comparable to conventional LoRA using QLoRA adapters trained under resource-constrained conditions. This approach not only improves the performance of trained QLoRA models without additional training but also accelerates LoRA fine-tuning.

1 INTRODUCTION

In recent years, the increasing parameter size of Large Language Models (LLMs) has significantly enhanced their performance, achieving success across various tasks (Kaplan et al. (2020); Dubey et al. (2024)). However, it has also led to the challenge of increasing computational costs. As a result, the infrastructure costs associated with deploying LLMs have increased, making it difficult for users or organizations with limited computational resources to deploy and operate LLMs.

Given this background, researchers have been studying techniques that combine Low-Rank Adaptation (LoRA) (Hu et al. (2021)) and Weight Quantization (Yao et al. (2022); Frantar et al. (2023); Lin et al. (2024)) to facilitate the fine-tuning (FT) and inference of LLMs under computational-resource constraints (Dettmers et al. (2023); Xu et al. (2024); Li et al. (2024); Jeon et al. (2024)). In QLoRA, one of the earliest studies in this area, the foundation model is quantized first, and then LoRA is applied. LoRA is a parameter-efficient FT method that trains not the model's weights directly but low-rank matrices, known as adapters, which are merged into each layer of the LLM. LoRA is a popular technique because it significantly reduces computational costs and achieves nearly the same performance as Full FT. On the other hand, Weight Quantization compresses the weights of LLMs, which are typically represented using a 16-bit data type, into 8-bit, 4-bit, or even lower precision, thereby reducing the model's memory requirements. Although quantization introduces quantization errors that slightly degrade the model's performance, the benefit of utilizing larger and more intelligent foundation models within the same memory capacity outweighs this drawback. Combining these two techniques reduces GPU memory usage during both FT and inference of LLMs, making it possible to construct a higher performing LoRA model with limited computational resources.

The existing quantization-LoRA frameworks, such as QLoRA, aim to reduce the computational resource required for FT. However, these frameworks do not consider the resources available during inference, meaning surplus resources cannot be effectively utilized. For example, when FT and inference are executed on the same computational resources, FT requires memory to store gradients and optimizer states. In contrast, inference does not, resulting in a relative surplus of memory during inference. Alternatively, inference efficiency techniques, such as offloading part of the processing to the CPU, could also create surplus resources. Leveraging these available memory surpluses during inference enables the use of foundation models with larger bit-widths, potentially mitigating performance degradation induced by quantization errors.

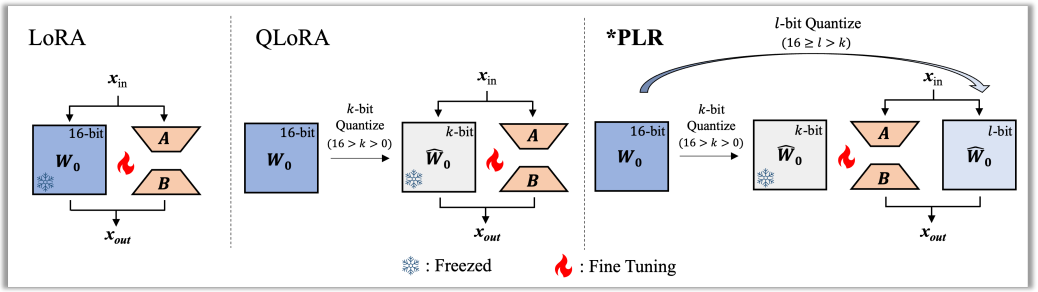

Figure 1: Conceptual diagram of the proposal method (PLR) and existing methods (LoRA, QLoRA). PLR restores the foundation model and merges the adapter trained using QLoRA. Note that PLR can apply not only to QLoRA but also to subsequent studies, such as QA-LoRA (Xu et al. (2024)).

This study proposes Post-LoRA Restoration (PLR) as a framework to utilize the surplus GPU memory during inference effectively. Figure 1 illustrates the workflow of the proposed method. In PLR, after applying LoRA training to a quantized foundation model, the model is restored to a larger bit-width for inference. This is based on the hypothesis that, because quantized models are tuned to behave as closely as possible to their originals, an adapter trained on a foundation model at one bit-width will remain effective when transferred to foundation models of other bit-widths. PLR enables the avoidance of performance degradation in the foundation model due to quantization errors during inference and can improve the processing performance of the LoRA model across various tasks.

In the evaluation experiments, we observed performance improvements in QLoRA models with PLR under nearly all conditions, confirming that the transferred adapter functions effectively. Furthermore, it was confirmed that 4-bit and 3-bit QLoRA combined with PLR achieved performance equivalent to conventional 16-bit LoRA. These results suggest that PLR is an effective method for speeding up training by reducing memory costs during learning and enabling larger batch sizes.

2 Post-LoRA Restoration

Post-LoRA Restoration (PLR) restores the precision of the foundation model's weights to a higher bit-width after QLoRA training. Our framework makes it possible to avoid the performance degradation of the foundation model caused by quantization errors.

The specific process of PLR is introduced below. Let $\boldsymbol{W}_{\mathbf{0}}^{\text{16-bit}}$ denote the weights of the foundation model with 16-bit precision, and $\boldsymbol{\Delta W}_{\mathbf{LoRA}}$ denote the weights of the LoRA adapter. Then, the weights of each layer after applying LoRA can be expressed as $\boldsymbol{W} = \boldsymbol{W}_{\mathbf{0}} + \boldsymbol{\Delta W}_{\mathbf{LoRA}}$. In a Quantization-LoRA framework like QLoRA, $\boldsymbol{W}_{\mathbf{0}}^{\text{16-bit}}$ is quantized to k-bit and replaced. Consequently, each layer is given by $\boldsymbol{W} = \boldsymbol{W}_{\mathbf{0}}^{k\text{-bit}} + \boldsymbol{\Delta W}_{\mathbf{LoRA}}$. Here, k satisfies $0 < k < 16$. In PLR, after completing LoRA training, $\boldsymbol{W}_{\mathbf{0}}^{k-bit}$ is replaced with $\boldsymbol{W}_{\mathbf{0}}^{l-bit}$. Consequently, each model layer is given by $\boldsymbol{W} = \boldsymbol{W}_{\mathbf{0}}^{l-bit} + \boldsymbol{\Delta W}_{\mathbf{LoRA}}$ during inference. Here, l satisfies $l > k$ and is chosen based on the surplus computational resources available for inference.

In general, LLMs incur more significant quantization errors and degraded performance as they are quantized to lower bit-width. PLR enables the selection of an optimal quantization bit-width for the model based on the available computational resources during inference, thereby minimizing performance degradation caused by quantization errors.

3 Experiments

In this experiment, we investigate whether the accuracy of each evaluation task improves when PLR is applied to QLoRA.

3.1 Settings

Foundation Models and Datasets In our experiments, we use three models from the Llama 3 family (Dubey et al. (2024)), Llama 3.2-1B, Llama 3.2-3B, and Llama 3.1-8B, as foundation models

Table 1: Accuracy (%) on each task for the LoRA, QLoRA, and QLoRA+PLR models. Note that PLRl means restoring the QLoRA foundation model to l-bit.

Datasets	Models	16-bit LoRA	8-bit QLoRA		4-bit QLoRA			3-bit QLoRA				2-bit QLoRA				
			QLoRA	**PLR16**	QLoRA	**PLR8**	**PLR16**	QLoRA	**PLR4**	**PLR8**	**PLR16**	QLoRA	**PLR3**	**PLR4**	**PLR8**	**PLR16**
GSM8k	Llama 3.2-1B	21.60	**22.13**	21.83	18.19	**18.95**	**19.41**	11.22	**15.31**	**16.91**	**16.53**	**3.18**	3.11	3.03	2.81	2.43
	Llama 3.2-3B	42.91	**43.06**	42.07	37.75	**41.32**	**42.38**	29.56	**34.50**	**37.76**	**37.30**	4.17	**6.60**	**15.61**	**15.39**	**15.85**
	Llama 3.1-8B	59.66	58.75	**58.98**	56.56	**58.15**	**58.00**	50.34	**57.31**	**58.98**	**59.59**	5.16	**18.27**	**33.35**	**33.73**	**34.34**
	Avg.	41.39	**41.31**	40.96	37.50	**39.47**	**39.93**	30.37	**35.71**	**37.88**	**37.81**	4.17	**9.33**	**17.33**	**17.31**	**17.54**
SCC	Llama 3.2-1B	69.21	**67.29**	63.33	25.84	**30.11**	**31.60**	23.90	22.19	**37.69**	**37.63**	0.00	0.00	**0.22**	**0.22**	**0.19**
	Llama 3.2-3B	72.30	**83.02**	82.79	75.22	**80.49**	**80.41**	68.65	68.15	**75.54**	**75.44**	2.97	**32.93**	**39.51**	**35.83**	**35.87**
	Llama 3.1-8B	84.96	**84.30**	84.21	84.38	**84.47**	**84.46**	77.35	**79.46**	**80.04**	**79.94**	4.81	**36.73**	**32.20**	**36.13**	**36.20**
	Avg.	75.49	**78.20**	76.78	61.81	**65.02**	**65.49**	56.63	56.60	**64.42**	**64.34**	2.59	**23.22**	**23.98**	**24.06**	**24.09**
Avg.	Llama 3.2-1B	45.40	**44.71**	42.58	22.02	**24.53**	**25.51**	17.56	**18.75**	**27.30**	**27.08**	1.59	1.55	**1.63**	1.52	1.31
	Llama 3.2-3B	57.61	**63.04**	62.43	56.49	**60.91**	**61.40**	49.11	**51.33**	**56.65**	**56.37**	3.57	**19.77**	**27.56**	**25.61**	**25.86**
	Llama 3.1-8B	72.31	71.53	**71.60**	70.47	**71.31**	**71.23**	63.85	**68.39**	**69.51**	**69.77**	4.99	**27.50**	**32.78**	**34.93**	**35.27**
	Avg.	58.44	**59.76**	58.87	49.66	**52.25**	**52.71**	43.50	**46.15**	**51.15**	**51.07**	3.38	**16.27**	**20.65**	**20.69**	**20.81**

in our experiments. For the dataset of FT, we select Grade School Math 8K (GSM8k) (Cobbe et al. (2021)) and SQL Create Context (SCC) (b mc2 (2023)). GSM8k comprises approximately 7,500 training samples and around 1,300 evaluation samples, while SCC includes roughly 78,600 training samples. Accordingly, for LoRA fine-tuning and evaluation, GSM8k's training set is split with 10% held out for validation. For SCC, 10% of its training data is used for validation and an additional 10% for evaluation.

Training Setting In LoRA and QLoRA, the adapter's rank r and α are set to 16 for all models, and LoRA is applied to all attention layers. The learning rate is 2e-4 and the batch size is 32. Among the 10 training epochs, we use the weights from the epoch that achieved the lowest loss on the validation data for evaluation. For quantizing the foundation model in LoRA, four quantization variations were prepared 8, 4, 3, and 2-bit width. Each quantization was performed using GPTQ (Frantar et al. (2023)). GPTQ is a quantization technique that, after quantizing the weights, employs calibration data to correct quantization errors and thereby mitigate performance degradation. We use 500 samples randomly drawn from the training data as the calibration data.

3.2 RESULTS

Table 1 shows the accuracy for each task of the LoRA, QLoRA, and QLoRA+PLR models, which were trained under the settings above. In the table, PLRl means restoring the QLoRA foundation model to l-bit.

Looking at the overall averages for all models and datasets in the table, PLR demonstrates improved performance compared to QLoRA in almost every case, thereby confirming the effectiveness of PLR. On the other hand, in the 8-bit scenario, PLR is the only method that exhibits a performance decline. Here, focusing on 16-bit LoRA, both 8-bit QLoRA and 8-bit QLoRA+PLR16 exhibit higher accuracy than 16-bit LoRA. This result suggests that as a foundation model for LoRA, the 8-bit one is superior to the 16-bit one, which can be interpreted as why 8-bit QLoRA outperform PLR16 restoring the foundation model to 16-bit. This phenomenon can be attributed to a regularization effect induced by 8-bit quantization, which allowed the adapter to learn its task more efficiently on the 8-bit model than on the 16-bit model.

Next, when comparing the results by model size, we observed that the effectiveness of PLR increases as the model size becomes larger and its performance improves. Notably, the accuracy of the 3-bit QLoRA+PLR16 on GSM8K and the 4-bit QLoRA+PLR16 on SCC is nearly identical to or even exceeds that of 16-bit LoRA. This fact suggests that on foundation models quantized to 4-bit or 3-bit precision, the adapter can learn to solve tasks at a level comparable to conventional LoRA. Our experimental results indicate that the performance degradation observed with QLoRA compared to LoRA is attributable to the foundation model's performance deterioration due to quantization errors, and that the adapter itself is being trained successfully up to a specific bit-width.

On the other hand, in the 2-bit scenario, the accuracy of QLoRA is remarkably low, and the accuracy recovery achieved by PLR remains insufficient compared to 16-bit LoRA. Since the adapter training may not proceed as expected, accurately verifying the effectiveness of PLR requires validation using quantization-aware LoRA methods such as QA-LoRA (Xu et al. (2024)).

4 DISCUSSION

Comparison of PLR and QLoRA The evaluation experiments demonstrate that PLR can improve QLoRA models' performance without requiring any additional training after FT. On the other hand, as PLR requires additional memory during inference, it is necessary to consider the trade-off between memory requirements and performance improvements carefully.

Figure 2 is a plot showing both the performance and the memory requirements of each model used in the evaluation experiments. For example, when applied to the 3-bit QLoRA model of Llama 3.1-8B, PLR4 increases memory usage by only 14% while reducing performance degradation relative to 16-bit LoRA by as much as 54% without requiring any additional training.

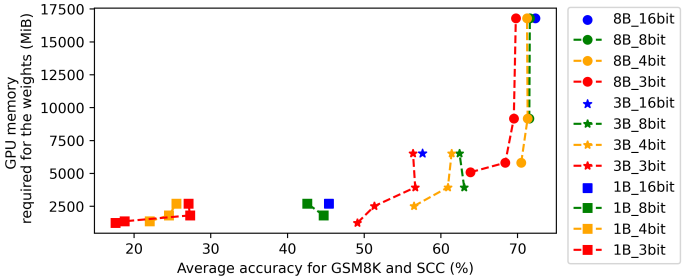

Figure 2: Trends in the accuracy of each (Q)LoRA model and the memory usage of its weights when PLR is applied.

Moreover, even if surplus memory for PLR is lacking, it may still be possible to apply PLR by utilizing techniques such as pruning (Zhang et al. (2024)) or offloading (Rasley et al. (2020); Alizadeh et al. (2023)). However, these techniques may lead to decreased inference speed or degraded performance, so we must thoroughly investigate their impact when combined with PLR.

Comparison of PLR and LoRA Our experiments indicate that 4-bit QLoRA + PLR16 and 3-bit QLoRA + PLR16 can achieve performance equivalent to vanilla 16-bit LoRA. Based on this finding, our method can also be applied to accelerate LoRA training. For example, when using 4-bit QLoRA combined with PLR16 on Llama 3.1-8B, up to 75% of training memory can be saved compared to 16-bit LoRA, even though the restored accuracy remains nearly identical. Therefore, by utilizing the saved memory to increase the batch size, training speed can be improved by reducing the number of training steps.

On the other hand, the performance improvements achieved with PLR are inconsistent, and it remains unclear which QLoRA bit width should be used depending on the task and model. For example, in GSM8K with Llama 3.1-8B, 3-bit QLoRA+PLR16 outperforms 4-bit QLoRA+PLR16, which is counterintuitive. In order to effectively utilize PLR for accelerating training, it is essential to enhance the consistency of both PLR and QLoRA across various bit-widths.

5 CONCLUSION

In this study, we proposed PLR, a novel Quantization-LoRA framework that leverages the transferability of adapters across foundation models with different bit-widths. For the increasingly large LLMs of recent years, PLR holds significant promise, as it enables both faster training and higher accuracy without incurring additional training compared to LoRA.

Finally, we outline the future work. One of the limitations of our approach is that we can only apply to models with the same architecture. It is necessary to devise a method that leverages the transferability of adapters to support not only different bit-widths of the same model but also different bit-widths across models with different architectures. Another research topic is to investigate the applicability of PLR to successor techniques of LoRA, such as DoRA (Liu et al. (2024)) and Transformer-Squared (Sun et al. (2025)). By solving these issues, we will further expand the scope of the application of PLR, and it will be possible to construct high-performance fine-tuned LLM under a broader range of computing resource settings.

REFERENCES

Keivan Alizadeh, Iman Mirzadeh, Dmitry Belenko, Karen Khatamifard, Minsik Cho, Carlo C Del Mundo, Mohammad Rastegari, and Mehrdad Farajtabar. Llm in a flash: Efficient large language model inference with limited memory. *arXiv preprint arXiv:2312.11514*, 2023.

b mc2. sql-create-context dataset, 2023. URL `https://huggingface.co/datasets/b-mc2/sql-create-context`. This dataset was created by modifying data from the following sources: Zhong et al. (2017); Yu et al. (2018).

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: efficient finetuning of quantized llms. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pp. 10088–10115, 2023.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2021.

Hyesung Jeon, Yulhwa Kim, and Jae joon Kim. L4q: Parameter efficient quantization-aware fine-tuning on large language models, 2024.

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. *arXiv preprint arXiv:2001.08361*, 2020.

Yixiao Li, Yifan Yu, Chen Liang, Nikos Karampatziakis, Pengcheng He, Weizhu Chen, and Tuo Zhao. Loftq: Lora-fine-tuning-aware quantization for large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6: 87–100, 2024.

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. *arXiv preprint arXiv:2402.09353*, 2024.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3505–3506, 2020.

Qi Sun, Edoardo Cetin, and Yujin Tang. Transformer-squared: Self-adaptive llms. *arXiv preprint arXiv:2501.06252*, 2025.

Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhengsu Chen, XIAOPENG ZHANG, and Qi Tian. Qa-lora: Quantization-aware low-rank adaptation of large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35:27168–27183, 2022.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*, 2018.

Mingyang Zhang, Hao Chen, Chunhua Shen, Zhen Yang, Linlin Ou, Xinyi Yu, and Bohan Zhuang. Loraprune: Structured pruning meets low-rank parameter-efficient fine-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pp. 3013–3026, 2024.

Victor Zhong, Caiming Xiong, and Richard Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103, 2017.

ACKNOWLEDGMENTS

This work was partially supported by the Grants-in-Aid for Academic Promotion, Graduate School of Culture and Information Science, Doshisha University, and JSPS KAKENHI Grant Number JP23K21726.