ELO: Efficient Layer-Specific Optimization for Continual Pretraining of Multilingual LLMs

Hangyeol Yoo[1*] **ChangSu Choi**[1,2*†] **Minjun Kim**[3] **Seohyun Song**[1]
SeungWoo Song[1] **Inho Won**[3] **Jongyoul Park**[1] **Cheoneum Park**[4] **KyungTae Lim**[3‡]
[1]Seoul National University of Science and Technology [2]LG CNS
[3]Korea Advanced Institute of Science and Technology [4]Hanbat National University
{hgyoo, choics2623}@seoultech.ac.kr, ktlim@kaist.ac.kr

Abstract

We propose an efficient layer-specific optimization (ELO) method designed to enhance continual pretraining (CP) for specific languages in multilingual large language models (MLLMs). This approach addresses the common challenges of high computational cost and degradation of source language performance associated with traditional CP. The ELO method consists of two main stages: (1) ELO Pretraining, where a small subset of specific layers, identified in our experiments as the critically important first and last layers, are detached from the original MLLM and trained with the target language. This significantly reduces not only the number of trainable parameters but also the total parameters computed during the forward pass, minimizing GPU memory consumption and accelerating the training process. (2) Layer Alignment, where the newly trained layers are reintegrated into the original model, followed by a brief full fine-tuning step on a small dataset to align the parameters. Experimental results demonstrate that the ELO method achieves a training speedup of up to 6.46 times compared to existing methods, while improving target language performance by up to 6.2% on qualitative benchmarks and effectively preserving source language (English) capabilities.

1 Introduction

Recent studies have focused on enhancing multilingual large language models (MLLMs) for specific languages (Zhao et al., 2023). Notably, studies like Chinese-Llama (Cui et al., 2024) and EEVE (Kim et al., 2024) have demonstrated improved performance by continual pretraining (CP) of MLLMs for target languages. However, these models encounter two major challenges. First, enhancing performance on a target language often significantly degrades performance on the primary language, English (Choi et al., 2024). Second, enhancing performance in the target language through CP demands significant time and resources, posing challenges for small-scale researchers (Naveed et al., 2024). To address these issues, lightweight training techniques such as Low-Rank Adaptation (LoRA) have been introduced to enhance model performance by modifying only a portion of the model (Hu et al., 2021). However, even when using LoRA, the time savings compared with full fine-tuning (FFT) are minimal. This is because while it significantly reduces the number of trainable parameters, the forward pass requires computation through both the original model weights and the additional LoRA parameters.

This computational overhead during the forward pass led us to a new hypothesis. Instead of merely limiting the trainable parameters within the full model (like LoRA), what if we could also reduce the computed parameters during CP by training a much smaller, separate model?

This line of inquiry led to our core concept: detaching a small subset of MLLM layers to be trained independently. Following this, we propose an efficient layer-specific optimization (ELO) method that focuses solely on this detached portion of layers for enhancing specific languages. The proposed method comprises two phases: ELO pretraining and layer alignment. First, ELO pretraining involves this detachment and CP process to imbue specific linguistic knowledge. Layer alignment is the phase where the newly acquired knowledge from ELO pretraining is transferred into the original MLLM.

This approach significantly reduces the number of model parameters during CP, thereby minimizing time and resource costs. Experimental results indicate that the training speed of the proposed method was 6.46 times faster. Qualitative evaluations performed similar or up to 6.2% superior

[*] Equal Contribution
[†] Work done during an internship at LG CNS
[‡] Corresponding Author

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 5: Industry Track, pages 752–763

results. The contributions of this study can be summarized as follows:

- We propose an efficient CP method, ELO, for MLLMs, enriching the availability of specific languages.
- Through comprehensive analysis, we have demonstrated the real-world effectiveness of the approach employed in our method.

2 Related Work

Efficient Fine-Tuning. Parameter-efficient fine-tuning (PEFT) methods are gaining prominence as language models continue to grow in size. These methods efficiently customize pretrained models for specific languages or tasks (Bai et al., 2024). Among these, LoRA (Hayou et al., 2024; Lialin et al., 2023; Dettmers et al., 2024) is a notable lightweight training method that achieves performance comparable to that of FFT by training a subset of parameters. However, these methods offer minimal training speedup over FFT. This is because while they reduce the number of trainable parameters, the computational cost remains high, as the forward pass must still be computed through all original model weights and the additional adapter parameters (Hu et al., 2021).

Selective Layer Tuning. As the number of layers in LLM increases, research on layer-selective tuning, based on the distinct roles each layer performs, has been proposed. Lad et al. (2024) demonstrated that not all layers serve the same function; the middle layers are responsible for understanding context and sentence structure, whereas the initial and final layers focus on integrating information. In a similar vein, EEVE (Kim et al., 2024) proposed a method for training only specific layers in the target language to improve performance in target language. While selective, this approach (as utilized in EEVE) still operates within the full model architecture. Consequently, it suffers from the same computational overhead as LoRA: the forward pass must still be computed across all model parameters, even if only a subset of layers is being updated (Kim et al., 2024).

3 Efficient Layer-Specific Optimization

As established in Section 2, conventional PEFT methods like LoRA and selective layer training suffer from a significant computational bottleneck. Although they reduce the number of trainable parameters, they still require the forward pass to be computed across the entire model architecture. This results in minimal training speedup over FFT.

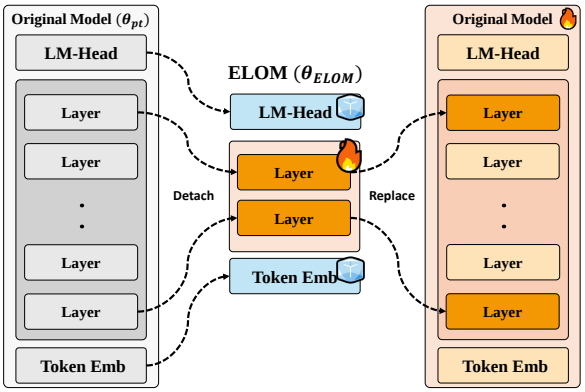

Figure 1: Description of the proposed ELO training process

To overcome this fundamental limitation, we propose Efficient Layer-Specific Optimization (ELO). The core idea of ELO is to detach a small subset of specific layers from the original model before pretraining. This action creates a much smaller, independent model for the CP phase. This layer detachment approach directly solves the overhead problem by drastically reducing not only the trainable parameters but also the total parameters computed during the forward pass. The ELO method comprises two main stages: (1) ELO pretraining and (2) layer alignment.

3.1 ELO Pretraining

The initial stage involves detaching specific layers from the original model for pretraining, as shown in Figure 1. The language model comprises n decoder layers $\mathbf{L} = \{\ell_1, \ell_2, \ldots, \ell_n\}$, the token embedding layer ℓ_e, and the head layer ℓ_h. We define the set of specific layers that comprise the ELO model as $\lambda \subset \mathbf{L}$, where θ^e and θ^h represent parameters for ℓ_e and ℓ_h, respectively, and θ^λ represents parameters for λ. We selected λ to encompass the first and last decoder layers, i.e., $\lambda = \{\ell_1, \ell_n\}$. The pretraining process can be expressed as follows:

$$\theta_{\text{ELOM}} = \{\theta^e, \theta^h, \theta^\lambda\} \quad (1)$$

$$\mathcal{L}_{\text{PT}} = -\sum_{i=1}^{|D_{\text{pt}}|}\sum_{j=1}^{|x_i|} \log P(x_j | x_{<j}; \theta_{\text{ELOM}}) \quad (2)$$

The ELO model was trained using each sample from the pretraining dataset D_{pt}, with the English-{Target Language} ratio set to 1:9 according to Equation 2. In this context, $\mathcal{L}_{PT}$ represents the causal language-modeling loss function, with θ_0

denoting the parameters of the original model. Only the parameters θ^{λ} of the ELO model are trained to infuse knowledge of the target language.

3.2 Layer Replacement and Alignment

The second stage transfers knowledge learned during the first stage back to the original model (θ_0). We replace θ^{λ} in the original model θ_0 with θ_{ELOM}. By replacing these two layers, it is possible to inject knowledge of the target language into specific layers while preserving the finely tuned token embedding and head layers, as well as existing layers rich in English knowledge. However, because this method modifies only the parameters of specific layers in the original model, aligning these layers requires further pretraining using a small dataset. Accordingly, we introduce a layer alignment step after replacement, wherein FFT is applied to the entire model using an additional 1GB of training data.

3.3 Bilingual Instruction Tuning

Because the model that has undergone the layer alignment process is a pretrained model, it exhibits limited capability in following user instructions. To efficiently improve instruction-following performance in the target language with less data, we first adopt the chat vector method (Huang et al., 2024).

This method extracts knowledge by calculating the deviation ($\theta_{\text{chat vector}} = \theta_{\text{Inst}} - \theta_{\text{PT}}$) between a pretrained model (θ_{PT}) and an instruction-tuned model (θ_{Inst}). The extracted $\theta_{\text{chat vector}}$ is then integrated into our layer-aligned model to efficiently transfer the instruction-following capabilities. After integrating the chat vector, we then conducted supervised fine-tuning (SFT).

For SFT, we utilized 31K instruction data extracted in a 1:1 ratio in both the target language and English from the ShareGPT-style dataset (Devine, 2024), which contains dialogue records from language models such as GPT4 (OpenAI, 2023). Details of the dataset can be found in Appendix A.3.

4 Experiments

Our experiments are designed to empirically validate the main claims of ELO. We aim to assess whether our layer detachment strategy successfully overcomes the forward-pass bottleneck and leads to significant training speedups compared to FFT and LoRA. We also evaluate whether ELO effectively enhances performance in the target languages (Korean and Japanese) compared to both the base model and traditional FFT, and whether it maintains strong performance in the source language (English) without the significant degradation often seen in CP. To answer these questions, we first introduce the evaluation benchmarks, the models used, and then present our main results. We follow this with an in-depth ablation study to justify our specific design choices, such as layer selection and the alignment process.

4.1 Evaluation Benchmarks

We conducted experiments to evaluate the effectiveness of ELO using Korean and Japanese as target languages, chosen for their distinct differences from English. Our evaluation of the LLMs was divided into quantitative and qualitative assessments (Zhou et al., 2023; Choi et al., 2024). Quantitative evaluation involves scoring based on numerical metrics (e.g., accuracy, F1-score), while qualitative evaluation assesses long-form generative answers using an LLM-as-a-judge (e.g., GPT-4).

For English, we used **MMLU**(Hendrycks et al., 2020) for quantitative evaluation, a benchmark that evaluates knowledge across 57 topics, measuring accuracy. For qualitative evaluation, we used **MT-Bench** (Zheng et al., 2023), a set of 80 challenging multi-turn open-ended questions evaluated by a GPT-4 judge on a 10-point scale.

For Korean, the quantitative benchmark was **KoBEST** (Jang et al., 2022), a suite of 5 NLU tasks requiring advanced Korean knowledge, evaluated using F1-score. The qualitative benchmark was **LogicKor** (Park, 2024), a multi-turn dataset measuring reasoning ability across six domains (e.g., reasoning, mathematics, coding) with 42 prompts, also judged by GPT-4 on a 10-point scale.

For Japanese, we employed **MARC-ja** (Kurihara et al., 2022) for quantitative assessment, a text classification task based on the Multilingual Amazon Reviews Corpus, using accuracy_norm as the metric. For qualitative assessment, we used **MT-Bench(ja)** (Stability-AI, 2024), a Japanese translation of MT-Bench, which similarly uses a GPT-4 judge and a 10-point scale.

4.2 Model Description

We compared the proposed ELO method with conventional FFT in terms of efficiency using the following open-source LLMs: Llama 3.1-8B (Dubey et al., 2024), Mistral-7B-v0.3 (Jiang et al., 2023),

Model	CP / ELO pretraining		Layer alignment			Quantitative Eval.		Qualitative Eval. (Out of 10)	
Korean									
	Data size (Tokens)	Time	Data size (Tokens)	Time	Total time	MMLU(**en**)	KoBEST(**ko**)	MT-Bench(**en**)	LogicKor(**ko**)
`Llama3.1-8B-Instruct`	-	-	-	-	-	**68.07**	56.02	**6.96**	6.03
`Llama3.1-FFT`	10 GB (2.8 B)	19.8 h	-	-	19.8 h	67.51	**66.08**	6.70	7.31
`Llama3.1-ELO`	9 GB (2.5 B)	1.5 h	1 GB (0.3 B)	2.0 h	3.5 h	66.69	60.81	6.79	**7.76**
`Mistral-7B-Instruct-v0.3`	-	-	-	-	-	**59.69**	49.01	6.72	4.49
`Mistral-FFT`	10 GB (4.7 B)	31.0 h	-	-	31.0 h	57.47	**61.68**	6.61	6.50
`Mistral-ELO`	9 GB (4.2 B)	1.7 h	1 GB (0.6 B)	3.1 h	4.8 h	58.90	60.56	**6.97**	**6.59**
`Qwen2-7B-Instruct`	-	-	-	-	-	69.89	60.17	**7.67**	6.90
`Qwen2-FFT`	10 GB (3.1 B)	20.5 h	-	-	20.5 h	**70.23**	**72.37**	7.11	6.95
`Qwen2-ELO`	9 GB (2.8 B)	1.8 h	1 GB (0.3 B)	2.1 h	3.9 h	70.11	71.57	7.25	**7.22**
Japanese									
	Data size (Tokens)	Time	Data size (Tokens)	Time	Total time	MMLU(**en**)	MARC-**ja**	MT-Bench(**en**)	MT-Bench(**ja**)
`Llama3.1-8B-Instruct`	-	-	-	-	-	**68.07**	**96.36**	6.96	4.85
`Llama3.1-FFT`	10 GB (2.7 B)	19.4 h	-	-	19.4 h	67.50	96.25	**6.99**	5.38
`Llama3.1-ELO`	9 GB (2.4 B)	1.4 h	1 GB (0.3 B)	2.0 h	3.4 h	67.51	95.35	6.90	**5.58**
`Mistral-7B-Instruct-v0.3`	-	-	-	-	-	**59.69**	83.43	**6.72**	4.36
`Mistral-FFT`	10 GB (4.1 B)	29.7 h	-	-	29.7 h	54.86	80.05	6.26	**5.68**
`Mistral-ELO`	9 GB (3.7 B)	1.7 h	1 GB (0.4 B)	3.0 h	4.7 h	55.19	**89.53**	6.38	**5.68**

Table 1: Performance and training time comparison of the proposed ELO method and FFT for three languages.

and Qwen2-7B (Yang et al., 2024). The model names in Table 1 refer to models trained using the following methods:

`{base_model}-Instruct` The official instruct-tuned models are released by each company.

`{base_model}-FFT` This model refers to one that was first fine-tuned on the `base_model` using the FFT method, followed by instruction tuning, as outlined in Section 3.3. For example, the `Llama3.1-FFT` model in Table 1 was trained with 10GB of CP data, followed by instruction tuning with 31K data.

`{base_model}-ELO` This refers to a model that applied the ELO method proposed in Section 3.

4.3 Experimental Results

Overall. The results presented in Table 1 indicate that both the FFT and ELO configurations significantly outperformed the `{base_model}-Instruct` models in the qualitative evaluation. Specifically, the ELO method achieved a 22.2% improvement in LogicKor performance compared with `Llama3.1-8B-Instruct`. However, in the quantitative evaluations, performance varied considerably across languages and base models. These findings indicate that the proposed pretraining and bilingual instruction tuning methods significantly enhance performance on target languages.

Qualitative Evaluation Effect of ELO. As shown in the Qualitative Evaluation column of Table 1, the models trained with the ELO consistently outperformed those trained with FFT in the qualitative assessments for Korean, and Japanese. Notably, the ELO models achieved higher scores in LogicKor, with a 0.45p improvement for Llama3.1 and a 0.27p improvement for Qwen2, than their FFT counterparts. Also, ELO has demonstrated substantial efficiency, reducing the training time by an average of 5.88-fold compared with that of FFT.

Quantitative Evaluation Effect of ELO. In the quantitative evaluations, performance varied with respect to the source language (English) and target languages. For the English MMLU evaluation, the base (-Instruct) models generally achieved the highest performance. However, the average performance difference compared with ELO was only 2.42%, suggesting that the impact was minimal. This is likely because both ELO and FFT were more focused on target languages using a 1:9 ratio in CP. Supporting this, the Korean quantitative evaluation (KoBEST) results show that the ELO models consistently outperformed the base (-Instruct) models by margins ranging from 8.55% to 23.57%.

Resource Efficiency of ELO. ELO has demonstrated substantial efficiency, reducing the training time by an average of 5.88-fold compared with FFT. The strength of ELO lies in its ability to achieve comparable or superior qualitative performance to that of FFT while using fewer computational resources. When trained on 10GB of PT data, ELO accelerates training by 5.26 to 6.46 times compared to FFT. For example, as shown in Table 1, the ELO-enabled model outperformed the Llama3.1-FFT model by 6.2% on LogicKor while achieving a 5.66-fold speedup.

Furthermore, Figure 3 shows that ELO significantly outperforms LoRA in training speed, empirically validating our hypothesis from Sections 1 and 2. While Figure 3 confirms that LoRA pro-

Model	Param	Data	Single	Multi	Total
Llama3-8B-Instruct	8 B	-	2.09	2.54	2.32
Llama3-8B-Instruct-SFT	8 B	-	6.26	5.45	5.86
Llama3-FFT	8 B	10 GB	6.14	6.21	6.18
Llama3-ELO	8 B	10 GB	6.40	5.95	6.18
Llama3-ELO	8 B	50 GB	6.40	6.36	6.38
Llama3-ELO	8 B	200 GB	**6.95**	**7.00**	**6.97**
Llama3-70B-Instruct	70 B	-	2.62	3.00	2.76
Llama3-ELO	70 B	50 GB	7.52	7.24	7.38
Llama3.1-70B-Instruct	70 B	-	7.66	7.90	7.78
Llama3.1-ELO	70 B	50 GB	**8.79**	**8.52**	**8.65**

Table 2: Internal evaluation results using LogicKor.

vides minimal time savings over FFT, ELO is 5.29 times faster than LoRA when trained with 50GB of data. This efficiency gap widens as the data size increases; with 200GB of data, ELO is 10.72 times faster than LoRA. These results demonstrate that ELO's layer detachment strategy successfully overcomes the forward-pass computational bottleneck that limits LoRA.

5 Ablation Study

In Section 4, we demonstrated that ELO achieves superior efficiency and performance compared to existing methods. However, critical questions regarding the optimal configuration and the underlying mechanisms of ELO remain. In this section, we conduct an in-depth analysis using the Korean qualitative benchmark, LogicKor, to address these inquiries. We first investigate whether the performance gain scales with the amount of pretraining data or if the limited capacity of the detached layers poses a bottleneck. We then examine if the improvements are consistent across different model sizes, such as 70B parameters, and disentangle the contribution of bilingual instruction tuning from the ELO pretraining itself. Furthermore, we provide an empirical justification for our selection of the first and last layers and analyze the sensitivity of performance to this choice. Finally, we verify the necessity of the layer alignment phase and determine the optimal amount of data required for this step.

ELO with More Pretraining. We now raise the question of whether increasing the volume of pretraining data diminishes learning effectiveness owing to the limited capacity of the layers to accommodate information. After examining the results in Table 2, we evaluated the performance of the `Llama3-ELO` model by pretraining it with volumes of data ranging from 10 to 200GB. Based on these findings, we observed that as the volume of pretraining data increased, there were substantial improvements in performance.

ELO with Bigger Size Model. Another question regarding the ELO method is whether similar performance improvements can be observed in larger models. The performance results for the 70B model are shown in Table 2. A comparison between `Llama3-70B-Instruct` and `Llama3-ELO`, both based on the 70B model, demonstrated significant performance improvements with the ELO model. However, since Llama3.1 showed substantial improvements in Korean language performance compared to version 3.0, additional experiments were needed to compare `Llama3.1-70B-Instruct` and `Llama3.1-ELO`. These experiments also revealed a notable performance increase of 10% with ELO.

Impact of Bilingual Instruction Tuning. In Table 2, `Llama3-8B-Instruct-SFT` refers to the model fine-tuned on `Llama3-8B-Instruct` using the instruction data outlined in Section 3.3. This significant performance improvement highlights the impact of instruction data. Moreover, the performance gap between the ELO model, which uses relatively small amounts of PT data, and `Llama3-8B-Instruct-SFT` was minimal. This suggests that increasing the volume of PT data is crucial to fully leveraging the benefits of ELO.

Model	MT-Bench (en)	LogicKor (ko)
Llama3.1-ELO(1,32)	6.96	7.76
Llama3.1-ELO(1,16,32)	7.11	7.69
Llama3.1-ELO(1,16)	7.03	5.79
Llama3.1-ELO(8,24)	7.19	5.00
Llama3.1-ELO(16,17)	7.00	5.47

Table 3: Impact of layer selection on ELO.

Why were the first and last layers selected? Our experiments revealed that applying ELO to the first and last layers yields the best performance. As shown in Table 3, the proposed $\lambda = \{\ell_1, \ell_n\}$ configuration, specifically `Llama3.1-ELO (1,32)`, significantly improved the LogicKor score to 7.76. This configuration, along with `Llama3.1-ELO (1,16,32)`, outperformed all others, indicating that the first and last layers are the most critical. This aligns with Lad et al. (2024)'s findings, which highlight the importance of these layers in synthesizing and aggregating information.

In contrast, other configurations were less effective. Training intermediate layers, such as $\lambda = \{\ell_8, \ell_{24}\}$ (`Llama3.1-ELO(8,24)`), resulted in a notably poor score of 5.0. This suggests that different

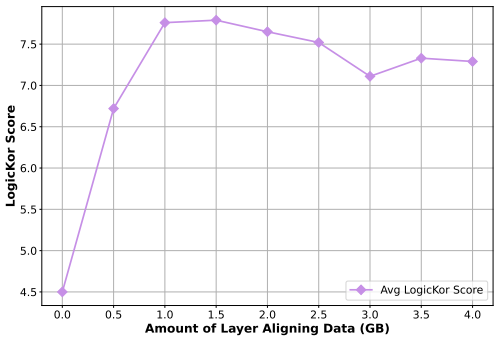

Figure 2: Performance variation of LogicKor based on the amount of PT data for its layer aligning

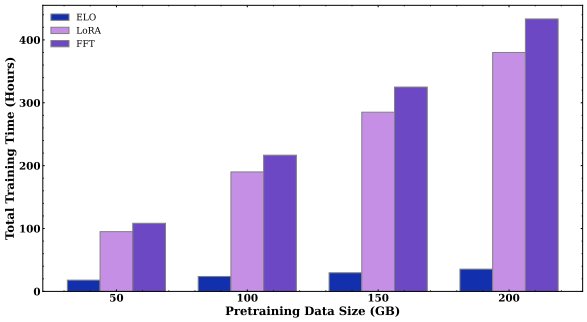

Figure 3: Comparison of the training time across ELO, FFT, and LoRA training methods

layers vary in importance when incorporating new knowledge. Furthermore, using only the 1st and 16th layers (`Llama3.1-ELO(1,16)`) led to minimal improvements, suggesting that the first layer alone struggles to maintain consistent knowledge flow.

An interesting observation is that, regardless of which layers were trained with ELO, the MT-Bench (English) scores remained stable. This likely reflects the fact that the layers not involved in ELO training (e.g., 30 layers in the (1,32) configuration) retained their English knowledge, preserving performance. However, when ELO was trained exclusively on the target language without bilingual training, we observed a decline in performance.

Effect of Layer Aligning To examine whether layer aligning is necessary and how much data is required for optimal performance, we progressively increased the amount of alignment data from 0GB to 4GB in 0.5GB increments using the `Llama-ELO` model described in Table 1. As shown in Figure 2, omitting layer aligning yielded the lowest LogicKor score (4.5), whereas even 1GB of data improved performance substantially to 7.76. The best result was obtained with 1.5GB (7.78), but further increases did not provide meaningful gains and in some cases slightly reduced performance. These results demonstrate that layer aligning is a crucial component of the ELO method and that only a small amount of bilingual data (approximately 1GB) is sufficient to achieve near-optimal performance. Consequently, we adopt 1GB as the default alignment size throughout this study to balance efficiency and effectiveness.

Comparison of Training Speed Based on Pretraining Data Size As mentioned in Section 2 and Section 3, limiting the number of trainable parameters in a model does not significantly reduce training time unless the model's overall size is reduced. Additionally, as the amount of training data increases, larger models require substantially more training time compared to smaller models. Figure 3 presents a comparison of the time taken to train `Llama3.1-8B` using ELO, LoRA, and FFT methods. When trained with 50GB of data, the ELO method is 5.29 times faster than LoRA and 6.04 times faster than FFT. However, when trained with 200GB of data, this difference increases to 10.72 times and 12.23 times. Therefore, the proposed method of enhancing specific languages through selective layer training becomes increasingly efficient as the amount of training data grows.

6 Conclusion

In this paper, we proposed Efficient Layer-Specific Optimization (ELO) to address the computational bottleneck of continual pretraining (CP) in MLLMs. Existing PEFT methods like LoRA offer minimal training speedup because they must compute the forward pass across the entire model. ELO overcomes this via a layer detachment strategy By training a small subset of critical layers (the first and last) as a smaller, independent model, ELO drastically reduces the parameters computed during the CP phase. This approach minimizes GPU memory consumption during this pretraining phase and enables significant acceleration.

Our experimental results demonstrate that ELO achieves a training speedup of up to 6.46 times compared to FFT. It also yields superior qualitative performance in target languages by up to 6.2%, while effectively preserving source language capabilities. This work establishes ELO as a highly efficient and effective alternative for multilingual adaptation.

7 Limitations

The ELO method minimizes GPU memory usage during the pretraining phase and accelerates the overall training process; however, it still has the following two limitations.

Even a minimal amount of FFT is required While our experiments have shown that layer alignment with a minimal amount of data, such as 1GB, is sufficient, the layer alignment phase remains essential. Since this phase requires training all the parameters of the original model, it demands more GPU memory than ELO pretraining. Therefore, it does not reduce the peak GPU memory requirement in the overall training process.

Investigation of CP experiments with over 1TB of data The performance of the FFT and ELO methods has not been verified when the data size exceeds 1TB. Unfortunately, it was impossible to conduct experiments with larger, high-quality datasets, as finding such data was difficult. Additionally, it is estimated that experimenting with larger models and larger datasets would require a significant amount of time, making these experiments infeasible. Therefore, while the ELO method demonstrated superior performance over FFT with datasets up to 200GB, further experiments with larger data sizes are necessary.

8 Acknowledgment

We would like to thank the reviewer for their insightful feedback throughout the study. This research was supported by the LG CNS collaborative research project, "Domain Expansion via Adaptive Policy Acquisition in Multi-Agent Systems" and Institute of Information & communications Technology Planning & Evaluation (IITP) grant, funded by the Korea government (MSIT) (No.RS-2024-00456709, A Development of Self-Evolving Deepfake Detection Technology to Prevent the Socially Malicious Use of Generative AI). We have used GPUs from Artificial intelligence industrial convergence cluster development project funded by the Ministry of Science and ICT (MSIT, Korea) & Gwangju Metropolitan City awarded to KyungTae Lim.

References

Guangji Bai, Zheng Chai, Chen Ling, Shiyu Wang, Jiaying Lu, Nan Zhang, Tingwei Shi, Ziyang Yu, Mengdan Zhu, Yifei Zhang, Carl Yang, Yue Cheng, and Liang Zhao. 2024. Beyond efficiency: A systematic survey of resource-efficient large language models. *Preprint*, arXiv:2401.00625.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, and 1 others. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

ChangSu Choi, Yongbin Jeong, Seoyoon Park, Inho Won, HyeonSeok Lim, SangMin Kim, Yejee Kang, Chanhyuk Yoon, Jaewan Park, Yiseul Lee, HyeJin Lee, Younggyun Hahm, Hansaem Kim, and KyungTae Lim. 2024. Optimizing language augmentation for multilingual large language models: A case study on Korean. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 12514–12526, Torino, Italia. ELRA and ICCL.

Yiming Cui, Ziqing Yang, and Xin Yao. 2024. Efficient and effective text encoding for chinese llama and alpaca. *Preprint*, arXiv:2304.08177.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Peter Devine. 2024. Tagengo: A multilingual chat dataset. *arXiv preprint arXiv:2405.12612*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, and 1 others. 2021. A framework for few-shot language model evaluation. *Version v0. 0.1. Sept*, page 8.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. 2024. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Shih-Cheng Huang, Pin-Zu Li, Yu-Chi Hsu, Kuang-Ming Chen, Yu Tung Lin, Shih-Kai Hsiao, Richard Tzong-Han Tsai, and Hung yi Lee. 2024. Chat vector: A simple approach to equip llms with instruction following and model alignment in new languages. *Preprint*, arXiv:2310.04799.

Myeongjun Jang, Dohyung Kim, Deuk Sin Kwon, and Eric Davis. 2022. KoBEST: Korean balanced evaluation of significant tasks. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3697–3708, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and 1 others. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Seungduk Kim, Seungtaek Choi, and Myeongho Jeong. 2024. Efficient and effective vocabulary expansion towards multilingual large language models. *arXiv preprint arXiv:2402.14714*.

Kentaro Kurihara, Daisuke Kawahara, and Tomohide Shibata. 2022. Jglue: Japanese general language understanding evaluation. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2957–2966.

Vedang Lad, Wes Gurnee, and Max Tegmark. 2024. The remarkable robustness of llms: Stages of inference? *Preprint*, arXiv:2406.19384.

Vladislav Lialin, Sherin Muckatira, Namrata Shivagunde, and Anna Rumshisky. 2023. Relora: High-rank training through low-rank updates. In *Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ NeurIPS 2023)*.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2024. A comprehensive overview of large language models. *Preprint*, arXiv:2307.06435.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Jeonghwan Park. 2024. Logickor:korean language model multidisciplinary reasoning benchmark.

Stability-AI. 2024. Logickor:korean language model multidisciplinary reasoning benchmark.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, and 1 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, and 3 others. 2023. A survey of large language models. *Preprint*, arXiv:2303.18223.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. Lima: Less is more for alignment. *Preprint*, arXiv:2305.11206.

Appendix

A Data Analysis

A.1 Details of the Benchmark Dataset

The evaluation of the LLM was divided into quantitative and qualitative assessments (Zhou et al., 2023; Choi et al., 2024). Quantitative evaluation involves scoring based on numerical metrics, which is done automatically. For instance, multiple-choice questions, such as true/false or four-option questions, were categorized under quantitative evaluation. The datasets used for this include MMLU, KoBEST, and MARC-ja. In contrast, qualitative evaluation was applied to tasks requiring the assessment of long-form answers, which were evaluated either by human judges or through automated evaluation using GPT. The datasets used for qualitative evaluation include MT-Bench, LogicKor, and MT-Bench(ja). Below is an introduction to the detailed evaluation datasets for each language.

English

- **MT-Bench:** MT-bench is a set of challenging 80 multi-turn open-ended questions for evaluating chat assistants . To automate the evaluation process, MT-bench prompts strong LLMs like GPT-4 to act as judges and assess the quality of the models' responses. The maximum score is 10 points.
- **MMLU:** MMLU(Massive Multitask Language Understanding) is a benchmark that evaluates knowledge across 57 topics. In this paper, we used accuracy as the evaluation metric.

Korean

- **LogicKor:** LogicKor is a multi-turn benchmark dataset designed to measure reasoning ability in various domains for Korean language models, using an LLM-as-a-judge approach. The dataset consists of a total of 42 multi-turn prompts across six categories: reasoning, mathematics, writing, coding, comprehension, and Korean language. LogicKor prompts strong LLMs like GPT-4 to act as judges and assess the quality of the models' responses. The maximum score is 10 points.
- **KoBEST:** KoBEST is a Korean benchmark suite consists of 5 natural language understanding tasks that requires advanced knowledge in Korean. In this paper, we used F1-score as the evaluation metric.

Japanese

- **MT-Bench(ja):** MT-Bench(ja) is a benchmark released by Stability-AI, created using the MT-Bench. MT-bench(ja) prompts strong LLMs like GPT-4 to act as judges and assess the quality of the models' responses. The maximum score is 10 points.
- **MARC-ja:** MARC-ja is a dataset of the text classification task. This dataset is based on the Japanese portion of Multilingual Amazon Reviews Corpus. In this paper, we used accuracy_norm as the evaluation metric.

A.2 Details of the Pretraining Dataset

Table 4,5 displays the sources of the datasets used for pretraining, along with the size of each dataset. In this paper, we express data size in GB rather than in tokens to avoid disparities in the number of samples across languages, which would hinder fair data utilization. The number of tokens per GB varies by language, ranging from approximately 1 billion to 1.3 billion. Thus, 10GB of data contains roughly 10 to 13 billion tokens.

Language	Source	Content	Size(GB)
Korean	AI-Hub[1]	News, Books	19.15
	Modu-corpus[2]	Paper, News	19.20
	WIKI-ko[3]	Wikipedia	1.17
	uonlp/CulturaX[4]	Web	99.73
	cc100-ko[5]	Web	40.46
	Total		**179.71**
English	fineweb[6]	Web	**19.98**
Total			**199.69**

Table 4: Korean Pretraining Dataset Source

Language	Source	Content	Size(GB)
Japanese	uonlp/CulturaX[7]	Web	**9.00**
English	fineweb[8]	Web	**1.00**
Total			**10.00**

Table 5: Japanese Pretraining Dataset Source

A.3 Details of the Instruction Tuning Dataset

For a fair evaluation, we used the publicly available Instruction-Following dataset during the SFT (Supervised Fine-Tuning) phase, applying it uniformly across all models. The Tagengo dataset consists of over 70,000 prompt-response pairs in the ShareGPT format, covering 74 languages, formatted similarly to those used in Vicuna (Chiang et al., 2023). This dataset underwent human review and modification. We collected Korean, Japanese and English subsets from the Tagengo dataset, gathering 31K Instruction-Following pairs. Samples of the Korean data utilized can be found in Table 6.

B Experiment Environment

To ensure reproducibility and comparability across studies, we conducted evaluations using publicly available benchmarking tools (Gao et al., 2021).

GPUs Used. We used eight NVIDIA H100 GPUs for the training and evaluation of the model.

Hyperparameters. The hyperparameter settings used in this study can be found in Table 7. All models were trained for 1 epoch during the PT stage and 10 epochs during the SFT stage.

Experiment Reproduction. We are making code used for testing available to allow for exact reproduction of the experiments conducted in this study. The qualitative responses generated by the models during the experiments can be downloaded from the supplementary materials.

[1]https://www.aihub.or.kr
[2]https://kli.korean.go.kr/corpus
[3]https://github.com/lovit/kowikitext
[4]https://huggingface.co/datasets/uonlp/CulturaX
[5]https://huggingface.co/datasets/lcw99/cc100-ko-only
[6]https://huggingface.co/datasets/HuggingFaceFW/fineweb
[7]https://huggingface.co/datasets/uonlp/CulturaX
[8]https://huggingface.co/datasets/HuggingFaceFW/fineweb

Sample of lightblue/tagengo-gpt4 data

[Korean Instruction]
금융 전문가가 되어 주식이 무엇인지 설명하시오.
(Please explain what stocks are as if you were a financial expert.)
[OUTPUT]
주식(stock)은 기업의 소유권을 나타내는 금융 자산입니다. 한 기업이 자본을 조달하기 위해 발행하는 주식을 구매함으로써, 구매자는 그 기업의 일부 소유권을 갖게 됩니다. 이는 구매자가 해당 기업의 주주가 되며, 기업의 이익과 손실, 성장과 위축에 따라 주식의 가치가 변동한다는 것을 의미합니다.
(A stock represents a financial asset that signifies ownership in a company. By purchasing shares issued by a company to raise capital, the buyer acquires partial ownership of that company. This means the buyer becomes a shareholder and the value of the stock fluctuates with the company's profits, losses, growth, and contraction.)

주식은 크게 두 가지 유형으로 나뉩니다: 보통주와 우선주입니다.
(Stocks are primarily divided into two types: common stock and preferred stock.)

1. **보통주(Common Stock)**: 보통주는 주주에게 투표권을 부여하며, 기업의 이익에 따라 배당금을 받을 수 있는 권리를 줍니다. 그러나 회사가 파산할 경우, 채권자, 우선주 주주, 그리고 다른 의무가 충족된 후에야 보통주 주주들이 자산에서 몫을 받을 수 있습니다.
(1. **Common Stock**: Common stock grants shareholders voting rights and the right to receive dividends based on the company's profits. However, in the event of bankruptcy, common stockholders receive their share of assets only after creditors, preferred stockholders, and other obligations have been satisfied.)

2. **우선주(Preferred Stock)**: 우선주는 보통주보다 배당금 지급에 있어 우선권을 갖습니다. 즉, 회사가 배당금을 지급할 때 우선주 주주들이 먼저 배당을 받습니다. 그러나 대부분의 우선주는 투표권이 없어 회사의 중요한 결정에 참여할 수 없습니다.
(2. **Preferred Stock**: Preferred stock has priority over common stock in terms of dividend payments. This means preferred stockholders receive dividends first when the company pays out dividends. However, most preferred stocks do not grant voting rights, so holders cannot participate in major company decisions.)

주식은 주식시장에서 거래됩니다. 이 시장은 크게 두 가지로 나뉘는데, 하나는 주식을 처음으로 공개하는 기업과 투자자 간의 거래가 이루어지는 **기업공개(IPO, Initial Public Offering)** 시장이고, 다른 하나는 이미 공개된 주식이 투자자들 사이에서 거래되는 **2차 시장**입니다.
(Stocks are traded on the stock market, which is divided into two main types: the **Initial Public Offering (IPO)** market, where shares are first issued and traded between the company and investors, and the **secondary market**, where already issued shares are traded among investors.)

주식 투자는 기업의 성장에 참여하고 그 이익을 공유할 수 있는 기회를 제공하지만, 기업의 성과에 직접적으로 영향을 받기 때문에 위험도 내포하고 있습니다. 따라서 투자자는 주식 투자에 앞서 기업 분석, 시장 분석, 자신의 투자 목표와 위험 감수 능력을 면밀히 고려해야 합니다.
(Stock investing provides an opportunity to participate in a company's growth and share in its profits, but it also involves risk due to the direct impact of the company's performance. Therefore, investors should carefully consider company analysis, market analysis, their investment goals, and risk tolerance before investing in stocks.)

Table 6: Example of the instruction data for ShareGPT data

	value
learning_rate	5.0e-06
Optimizer	AdamW_bnb_8bit
lr_scheduler	cosine
Epoch for PT	1
Epoch for SFT	10
sequence_len	8192
Batch size for ELO PT	4
Batch size for ELO align	1
Batch size for sft	1
Random Seed	42

Table 7: Applied hyperparameter settings.