A Data-Efficient Path to Multilingual LLMs: Language Expansion via Post-training PARAMΔ Integration into Upcycled MoE

Hao Zhou[1]*, Tianhao Li[2], Zhijun Wang[1], Shuaijie She[1], Linjuan Wu[3], Hao-Ran Wei[2]†, Baosong Yang[2], Jiajun Chen[1], Shujian Huang[1]†
[1]National Key Laboratory for Novel Software Technology, Nanjing University
[2]Tongyi Lab, Alibaba Group [3]Zhejiang University
{zhouh,wangzj,shesj}@smail.nju.edu.cn, {chenjj,huangsj}@nju.edu.cn
{chongsheng.lth,funan.whr,yangbaosong.ybs}@alibaba-inc.com
wulinjuan525@zju.edu.cn

Abstract

Expanding Large Language Models (LLMs) to new languages is a costly endeavor, demanding extensive Continued Pre-Training (CPT) and data-intensive alignment. While recent data-free merging techniques attempt to bypass alignment by fusing a multilingual CPT-enhanced model with its instruct counterpart, they are plagued by a critical trade-off: mitigating parameter conflicts to preserve original abilities inevitably dilutes new language acquisition, and vice-versa. To resolve this conflict, we introduce DeltaMoE , which upcycles a dense model into a Mixture-of-Experts (MoE) architecture, allocating different experts to different languages. Alignment ability is then transferred by grafting a MoE-expanded parameter delta (Δ_{post}) to the CPT-enhanced base model, bypassing the complex alignment phase. Experiments demonstrate DeltaMoE 's superiority even against baselines with similar FLOPs or number of parameters; it improves performance on expanded languages while effectively preserving original capabilities. We further show our approach is highly applicable across different models and Post-training deltas.

1 Introduction

Large Language Models (LLMs) such as LongCat-Flash (Team et al., 2025b), Kimi k2 (Team et al., 2025a), Deepseek (Guo et al., 2025), have demonstrated remarkable capabilities on a variety of tasks (Jiang et al., 2024; Wang et al., 2025; Luo et al., 2025; Wang et al., 2024). However, they remain primarily optimized for English, given that the majority of the pre-training corpus is in English. As a result, the models yield inferior results in non-English languages.

The standard training pipeline of expanding LLMs to other languages involves Continued Pre-Training (CPT) followed by post-training. CPT requires massive data replay to prevent catastrophic forgetting, a costly prerequisite for the effective alignment step. The subsequent alignment stage is even more demanding, requiring not only immense computational power but also vast amounts of quality instruction data. The scale of this challenge is exemplified by the alignment of Qwen2.5 (Yang et al., 2025), where each step incurs substantial costs: (1) Supervised Fine-Tuning (SFT) on millions of meticulously curated examples and (2) a complex two-stage Reinforcement Learning (RL) process involving Offline RL (Rafailov et al., 2023) and Online RL (Shao et al., 2024). Together, the demands for high-quality data and large-scale computation make this pipeline prohibitively costly and difficult to achieve, highlighting the need for a more efficient alternative.

Therefore, recently, Yamaguchi et al. (2024); Cao et al. (2025) have explored gradient-free methods to transplant alignment capabilities from a well-aligned LLM to a continually pre-trained ver-

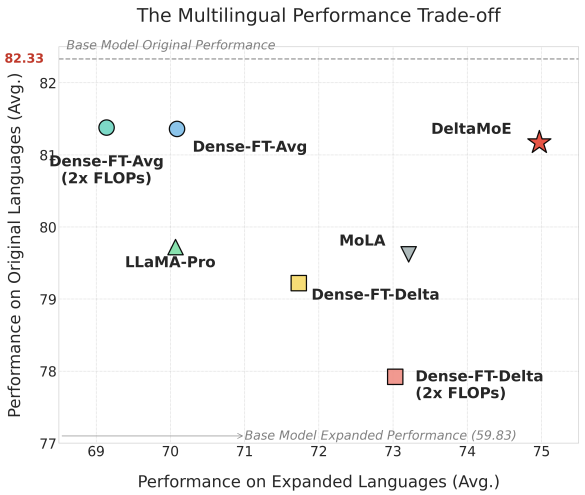

Figure 1: A visualization of the performance trade-off between expanded language capabilities (x-axis) and original language retention (y-axis). DeltaMoE resolves this conflict better than baseline methods.

*Work done during internship at Tongyi Lab.
†Corresponding author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 26888–26904
July 2-7, 2026

sion of its base model. The core idea of these methods is conducting parameter merging from the post-trained model and the CPT one. However, prevalent merging techniques exist a critical trade-off between original language and expanded language. Specifically, Yamaguchi et al. (2024) (similar to our DENSE-FT-AVG baseline) uses simple linear merge (Wortsman et al., 2022) to average the weights of the post-CPT model with the original instruct model. This operation inherently discounts the CPT updates, substantially diluting the newly acquired knowledge in expanded language and thus limiting gains on expanded languages. On the other hand, delta merging (Cao et al., 2025) (similar to our DENSE-FT-DELTA baseline) directly applies the full parameter changes from CPT, often inducing a drastic shift from the original weights, leading to catastrophic forgetting of its capabilities in the original language.

To address this critical trade-off, we introduce DeltaMoE , a novel approach that integrates the Mixture-of-Experts (MoE) architecture with a refined delta merging strategy. Specifically, we first upcycle the dense base model into an MoE structure. The original parameters are frozen to serve as a dedicated repository for the original knowledge during the CPT phase. Subsequently, the delta merging principle is applied to all experts, enhancing models with alignment capabilities. This two-stage strategy, illustrated in Figure 2, enables DeltaMoE to effectively inherit alignment capabilities on expanded languages from a well-trained open-source LLM, entirely bypassing the need for instruction data.

Experiments show that DeltaMoE outperforms strong baselines, improving average performance on expanded languages by 1.7 points and preservation capabilities by 1.5 points over other delta-based methods with comparable FLOPs or parameter counts, demonstrating our DeltaMoE effectiveness and efficiency.

Our contributions are described as follows:

- We propose a novel approach that effectively expand new languages for existing LLM and preserve original languages abilities, while acquiring alignment ability without a heavy post-training procedure.
- We conduct extensive experiments demonstrating that DeltaMoE resolves the key trade-off, substantially enhancing performance on expanded languages while mitigating catastrophic forgetting.
- We confirm the robustness and generality of this approach across various base models and post-training deltas.

2 Method

Our proposed method in Figure 2, DeltaMoE , is a two-stage process designed to efficiently build an expanded language-enhanced alignment model. The first stage, CPT, augments the model with new languages by selectively training only new experts and router, which preserves the original model's knowledge. The second stage, MoE Model Merging, then innovatively grafts a parameter delta from the original dense instruct model onto our sparse MoE architecture, effectively transferring alignment capabilities.

2.1 Continued Pre-training via Sparse Upcycling

The primary objective of this stage is to augment a pre-existing dense LLM with knowledge of new languages, while crucially preserving its original language capabilities. To achieve this, we employ a sparse upcycling strategy, transforming the dense model into an MoE Architecture.

MoE Architecture. Following the upcycling paradigm (Komatsuzaki et al., 2023), we initialize N experts by creating deep copies of the original feed-forward network (FFN) from the dense base model. As our base models employ SwiGLU-based FFNs (Grattafiori et al., 2024; Yang et al., 2025), the forward pass of a single expert E_i is defined as:

$$E(x)_i = (SiLU(xW^i_{\text{gate}} \odot xW^i_{\text{up}}))W^i_{\text{down}} \quad (1)$$

where $x \in \mathbb{R}^{\text{d}}$ is the input hidden state, $W^i_{\text{gate}}, W^i_{\text{up}} \in \mathbb{R}^{\text{d}\times\text{f}}, W^i_{\text{down}} \in \mathbb{R}^{\text{f}\times\text{d}}$ are the weight matrices of the i-th expert. d is model's hidden state dimension and f is the intermediate FFN dimension.

Top-k Gating. A router, parameterized by W_{router} determines the contribution of each expert for a given token. The gating weights $P \in \mathbb{R}^N$ are computed via a softmax over the router's logits:

$$p = \text{softmax}\,(xW_{\text{router}}) \quad (2)$$

We employ top-k routing, which activates only the k experts with the highest gating weights for each

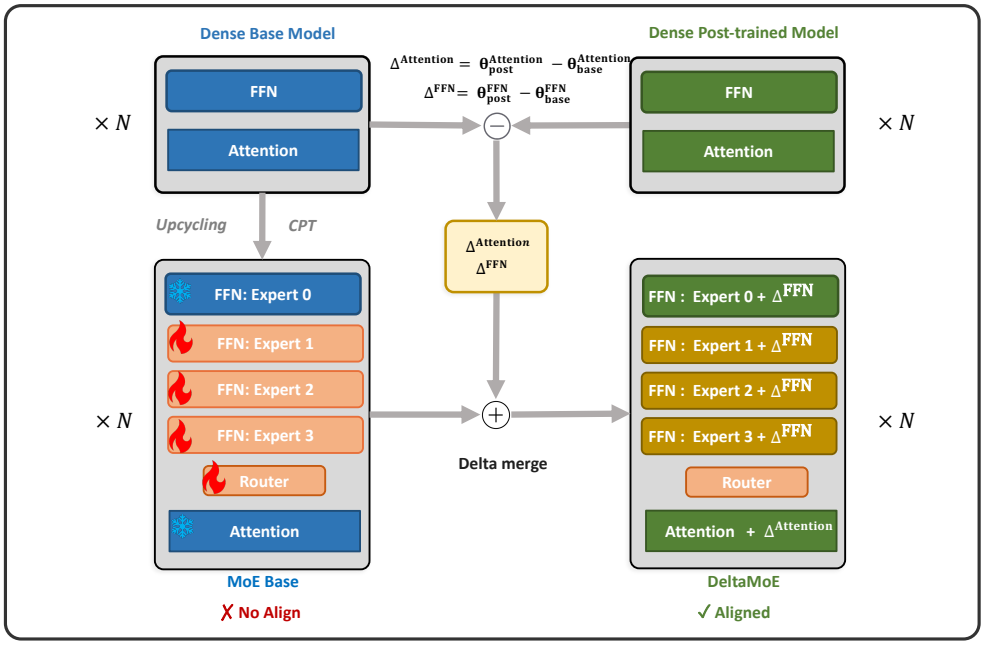

Figure 2: The two-stage DeltaMoE pipeline: 1) CPT via sparse upcycling with a frozen expert to preserve knowledge, followed by 2) MoE model merging to transfer alignment abilities.

token. Let $\mathcal{T} = \text{TopK}(p, k)$ be the set of indices for the k selected experts. The weights for these activated experts are re-normalized:

$$w_i = \begin{cases} \frac{p_i}{\sum_{j \in \mathcal{T}} p_j} & \text{if } i \in \mathcal{T} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The final output of the MoE layer is a weighted combination of the expert outputs, combined with a residual connection:

$$y = \sum_{i=1}^{N} w_i \cdot E_i(x) + x \tag{4}$$

Training Objective. To preserve the original language knowledge, we freeze all parameters of the dense base model, including the 0-th expert, which serves as a knowledge anchor. Consequently, only the newly added expansion experts and the router are updated during CPT (Zhou et al., 2025). The primary objective is the Next Token Prediction (NTP) loss, computed as follows:

$$\mathcal{L}_{NTP} = -\frac{1}{|\mathcal{D}|} \sum_{\mathbf{y} \in \mathcal{D}} \frac{1}{|\mathbf{y}|} \sum_{t=1}^{|\mathbf{y}|} \log P\left(y_t \mid y_{<t}; \theta_{\text{tr}}\right) \tag{5}$$

Here, D is the training corpus and θ_{tr} represents the set of trainable parameters, which includes only the expansion experts and the router: $\theta_{\text{tr}} = \{\theta^{(k)}_{\text{exp}}\}_{k=1}^{N} \cup \{\theta_{\text{router}}\}$

To mitigate the issue of unbalanced expert allocation, we use $\mathcal{L}_{LB}$:

$$\mathcal{L}_{LB} = N \cdot \sum_{i=1}^{N} f_i \cdot P_i \tag{6}$$

where N is the total number of experts, f_i is the fraction of tokens in a batch dispatched to expert i, P_i is the average router probability for expert i across the batch, and α is a scalar hyperparameter. The final training objective is the combination of these two losses:

$$\mathcal{L} = \mathcal{L}_{NTP} + \alpha \mathcal{L}_{LB} \tag{7}$$

where α is the hyper-parameter.

2.2 MoE Model Merging

Upon completion of the CPT stage, we obtain an expanded language-enhanced MoE base model, denoted as $M_{\text{MoE-base}}$. While this model possesses broad multilingual knowledge, it lacks the alignment capability. To instill these abilities without costly post-training, we propose a novel MoE Delta Merging strategy. This strategy adapts the delta parameterization concept (Cao et al., 2025) to our unique MoE architecture. The core idea is to compute a delta weight $\Delta_{\text{post}} = \theta_{\text{post-trained}} - \theta_{\text{base}}$ representing the knowledge gained during the post-training of a public, dense LLM. We then graft

this Δ_{post} onto our $M_{\text{MoE-base}}$ to create the final alignment model, $M_{\text{MoE post-trained}}$

Given that our $M_{\text{MoE-base}}$ is a sparse MoE model while the delta is derived from dense models, a direct application is not feasible. We therefore devise a component-wise merging strategy:

Shared Parameters: For parameters that are common to both the dense and MoE architectures (e.g: attention and embedding block), we directly apply the corresponding delta weight. Let $\theta^{\text{MoE}}_{\text{shared}}$ be such a parameter in our model, and $\Delta^{\text{MoE}}_{\text{shared}}$ be the corresponding delta from the dense models. The merged parameter is:

$$\hat{\theta}^{\text{MoE}}_{\text{shared}} = \theta^{\text{MoE}}_{\text{shared}} + \Delta^{\text{shared}}_{\text{post}} \tag{8}$$

Expert Parameters: For the expert FFN layers, which do not have a direct counterpart in the dense model's delta, we leverage the fact that they were initialized from the dense model's FFN. We compute a single FFN-specific delta, $\Delta^{\text{FFN}}_{\text{post}} = \theta^{\text{FFN}}_{\text{post}} - \theta^{\text{FFN}}_{\text{base}}$. This $\Delta^{\text{FFN}}_{\text{post}}$ is then applied uniformly to all expert parameters within our MoE base model. For the i-th expert's weights $\{W^i_{\text{gate}}, W^i_{\text{up}}, W^i_{\text{down}}\}$ the merging process is:

$$\begin{aligned} \hat{W}^i_{\text{gate}} &= W^i_{\text{gate}} + \Delta^{\text{gate}}_{\text{post}} \\ \hat{W}^i_{\text{up}} &= W^i_{\text{up}} + \Delta^{\text{up}}_{\text{post}} \\ \hat{W}^i_{\text{down}} &= W^i_{\text{down}} + \Delta^{\text{down}}_{\text{post}} \end{aligned} \tag{9}$$

In essence, we treat all experts as having inherited the same foundational structure, and thus they should all benefit from the same post-training update derived from the dense FFN. This strategy elegantly resolves the architectural mismatch and allows the post-training knowledge to be broadcast across all experts.

3 Experiment

3.1 Setup

Models. Our primary experiments use the Qwen2.5-7B (Yang et al., 2025) series as the backbone, and we validate the generalizability of our approach on the LLaMA-3.1-8B (Grattafiori et al., 2024) family in Section 5.2. These models were selected for their strong English performance and vocabularies well-suited for multilingual CPT. For our MoE architecture, we upcycle the dense model into a 4-expert MoE [1] with a top-2 gating strategy.

[1]We find that 3 trainable experts provide sufficient capacity to accommodate multiple expanded languages. See Appendix H for detailed experiments.

Training Details. All of our experiments are implemented using the LLaMA-Factory (Zheng et al., 2024) and are optimized for large-scale training with DeepSpeed ZeRO-3 (Rajbhandari et al., 2020). During the CPT stage, we train for 1 epoch. We set learn rate to $5e-5$ with a cosine learning rate scheduler. The global batch size is set to 512, with a maximum sequence length of 2048 tokens. All training is performed using BF16 mixed-precision. The load-balancing loss coefficient α is set to 0.01.

Datasets. We designate Hungarian (Hu), Serbian (Sr), and Bengali (Bn) as our expanded languages, selected due to the poor performance of the LLM on them. We also include high-resource original languages: English (En), Chinese (Zh), Spanish (Es), and French (Fr).

For each of the three expanded languages, we sample 3 billion tokens of unlabeled, monolingual text data. The data for Hungarian and Bengali are sourced from the FineWeb2 dataset (Penedo et al., 2025). As the volume of Serbian data in FineWeb2 is insufficient for our needs, we sourced the Serbian corpus from CulturaX (Nguyen et al., 2023).

Evaluation Benchmarks. To comprehensively assess zero-shot multilingual capabilities, we evaluate our models on a diverse suite of benchmarks. This includes tests for mathematical reasoning (MGSM; Huang et al., 2025),instruction following (MIFEVAL; Huang et al., 2025), reading comprehension (BELEBELE; Bandarkar et al., 2023), general knowledge (M_MMLU; Institute, 2025), and machine translation (FLORES-200; Costa-Jussà et al., 2022). Detailed descriptions of each benchmark, including specific prompting strategies and evaluation metrics, are provided in Appendix C.

Baselines. To rigorously evaluate the effectiveness of our proposed method, DeltaMoE , we compare it against baselines grouped into three categories for fair comparison: 1) methods with identical training data, 2) a matched computational budget (FLOPs), and 3) a comparable number of parameters.

- **DENSE-FT-AVG**: This baseline is a variant of the method proposed by Yamaguchi et al. (2024). It performs CPT on the public instruct model, then restores alignment capabilities by linearly averaging its weights with the original instruct model.

Model	Parameters (B)		Zero-shot Performance						
							Flores-200		
	Active	**Total**	**MGSM**	**MIFEVAL**	**BELEBELE**	**M_MMLU**	**En-XX**	**XX-En**	**Avg.**
Part 1: Performance on Expanded Languages (hu, sr, bn)									
Qwen2.5-7B-Instruct	7.6	7.6	60.40	59.27	71.89	47.29	40.31	79.84	59.83
Baselines w/ Same CPT Data									
Dense-FT-Avg	7.6	7.6	66.13	58.82	76.85	52.01	76.71	90.01	70.09
Dense-FT-Delta	7.6	7.6	61.87	60.80	80.00	54.48	84.14	89.11	71.73
Baselines w/ Matched FLOPs (2x CPT Data)									
Dense-FT-Avg-2FLOPs	7.6	7.6	62.13	56.74	76.48	50.66	78.45	90.38	69.14
Dense-FT-Delta-2FLOPs	7.6	7.6	64.67	62.49	81.44	54.64	85.18	89.73	73.03
Baselines w/ Matched Parameters									
LLaMA-Pro	10.8	10.8	60.80	58.72	81.70	55.01	82.56	86.82	70.94
MoLA	13.2	21.7	65.47	61.44	80.74	54.52	87.36	89.74	73.21
DeltaMoE	13.3	24.7	**67.60**	**64.95**	**82.22**	**56.11**	**88.13**	**90.82**	**74.97**
Part 2: Performance on Original Languages (en, zh, es, fr)									
Qwen2.5-7B-Instruct	7.6	7.6	72.50	77.27	90.47	69.50	89.94	94.33	82.33
Baselines w/ Same CPT Data									
Dense-FT-Avg	7.6	7.6	71.20	72.94	**89.56**	**68.68**	**90.87**	**94.95**	81.36
Dense-FT-Delta	7.6	7.6	68.20	73.08	88.78	65.69	86.66	92.89	79.22
Baselines w/ Matched FLOPs (2x CPT Data)									
Dense-FT-Avg-2FLOPs	7.6	7.6	**74.30**	72.58	88.17	67.96	90.41	94.86	**81.38**
Dense-FT-Delta-2FLOPs	7.6	7.6	67.00	72.28	87.47	63.67	84.55	92.57	77.92
Baselines w/ Matched Parameters									
LLaMA-Pro	10.8	10.8	69.30	68.13	88.06	64.86	89.04	92.68	78.68
MoLA	13.2	21.7	68.80	**73.77**	88.50	64.43	89.09	93.15	79.62
DeltaMoE	13.3	24.7	73.00	73.47	89.42	67.24	89.88	94.04	81.17

Table 1: Main results on expanded and original languages. "Total" denotes the total number of model parameters, while "Active" refers to the activated parameters during inference. The best results are in **bold**, and second-best are underlined. Detailed performance breakdowns are available in Appendix A.

- **DENSE-FT-DELTA**: Adopting the approach from Cao et al. (2025), this baseline starts with the base dense model, performs CPT, and then adds the pre-computed alignment delta, Δ_{instruct}, to instill alignment abilities.
- **DENSE-FT-AVG-2FLOPS & DENSE-FT-DELTA-2FLOPS**: To match the training FLOPs of our top-2 MoE architecture, these dense baselines are trained on twice the amount of CPT data (18B tokens total). Note that our top-2 MoE introduces approximately $1.8\times$ the training and inference FLOPs of a dense model; however, since the MoE optimizer updates all parameters, we conservatively use $2\times$ the CPT data to ensure a fair comparison.
- **LLAMA-PRO**: This baseline implements the block expansion strategy from Wu et al. (2024). We select the strongest configuration in Appendix E. After CPT, the alignment delta is applied only to the original dense parameters.
- **MOLA**: We implement MoLA (Gao et al., 2024) by adding LoRA (Hu et al., 2022) experts (rank=1120) to each linear layer. The number of experts per layer increases with model depth.[2] Similar to LLaMA-Pro, the alignment delta is added only to the original dense model weights.

A detailed summary of the CPT hyperparameters for all baselines is provided in Appendix B.

3.2 Main Results

As presented in Table 1, DeltaMoE demonstrates two key advantages. Firstly, it establishes state-of-the-art (SOTA) performance on the expanded languages. Secondly, it exhibits strong performance preservation on the original languages, offering a far more effective resolution to the inherent trade-off between expanding and retaining knowledge.

[2] Specifically, we divide the model's layers into four blocks and assign 2, 4, 6, and 8 LoRA experts to the linear layers within each respective block, from shallow to deep.

Dense Merging Reveals a Performance Trade-off. The dense model baselines trained on the same data reveal a stark performance trade-off. While Dense-FT-Delta surpasses Dense-FT-Avg by 1.64 points in expanded languages, it incurs a significant drop of 2.14 points in original languages. This trade-off originates from their underlying mechanics. The linear interpolation of Dense-FT-Avg ($\theta_{\text{merged}} = \frac{1}{2}(\theta_{\text{post-trained}} + \theta_{\text{post-trained cpt}})$) dilutes the newly learned knowledge, limiting its gains. Conversely, the parameter shift in Dense-FT-Delta leads to severe catastrophic forgetting of the original abilities. In contrast, DeltaMoE circumvents this issue. DeltaMoE 's MoE architecture decouples knowledge acquisition from retention, thus achieving a superior performance balance.

Superiority under Matched FLOPs. Even when dense baselines are given twice the CPT data to match DeltaMoE training FLOPs, DeltaMoE 's superiority holds. Although Dense-FT-Delta-2FLOPs improves on expanded languages by about 3 points to its 1x-data counterpart, it still lags behind DeltaMoE by a significant margin of 1.94 points. More critically, its catastrophic forgetting worsens by 1.3 points compared to its 1x-data counterpart. This suggests that simply scaling data is an ineffective strategy for delta-based methods. Meanwhile, Dense-Avg-2FLOPs performs even worse on expanded languages than its 1x-data counterpart, confirming that its averaging mechanism severely dilutes new knowledge regardless of data volume.

Architectural Advantage of DeltaMoE . Among parameter-expansion architectures, DeltaMoE proves superior. The vertical expansion of LLaMA-Pro is suboptimal, as it lags behind DeltaMoE by a substantial 4 points on expanded languages. While the stronger MoLA baseline is more competitive, it still lags behind our method by 1.76 points on expanded and 1.55 points on original languages. We attribute this performance gap to a fundamental architectural incompatibility: the dense alignment delta cannot be applied to MoLA's LoRA experts. In contrast, our method's architectural design enables the direct application of the alignment delta to our experts.

4 Analysis and Ablation

4.1 Analysis of Expert Routing Allocation

As shown in Figure 3, the router demonstrates clear language-based specialization. For English, it routes nearly all tokens to the frozen 0-th expert, preserving original-language capabilities. Conversely, expert selection for Hungarian shows a dynamic division of labor: trainable expansion experts are active in the upper layers, while the frozen 0-th expert handles middle-layer computations. This suggests the model performs core reasoning in English (Wendler et al., 2024).

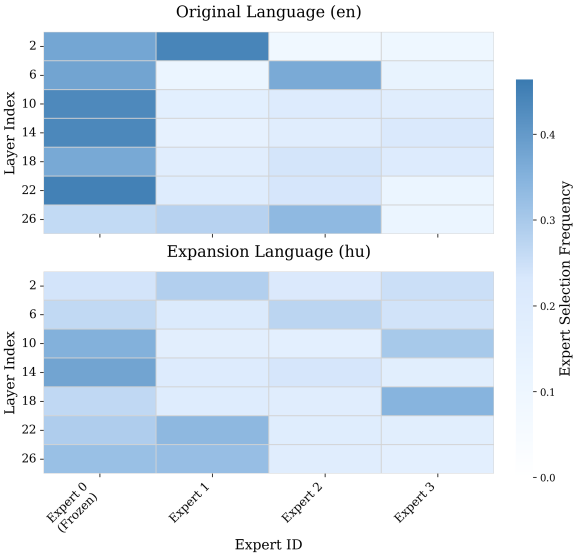

Figure 3: Average expert selection frequency across layer for English and Hungarian inputs in the ifeval benchmark.

4.2 Effective and Efficient Knowledge Retention

The retention scores in our main results (Table 1) motivate this targeted analysis. While DeltaMoE slightly trails Dense-FT-Avg in retention, it significantly outperforms Dense-FT-Delta. We therefore investigate two aspects: 1) whether minimal data replay can close the retention gap with Dense-FT-Avg, and 2) the architectural efficiency of DeltaMoE over its dense delta counterpart.

To investigate this, we use a minimal 0.15B token in-domain replay budget (en, es, zh) and hold out French as an out-of-domain (OOD) test[3]. This data is applied via two distinct strategies: standard data mixing during CPT for dense models, versus a targeted post-CPT router-tuning phase(Zhou et al., 2025) for DeltaMoE (Appendix D).

Figure 4 shows our approach is both highly effective and efficient. With this minimal replay bud-

[3]Sourced from DCLM (Li et al., 2024) (English), FineWeb2 (Penedo et al., 2025) (Spanish), and SkyPile-150B (Wei et al., 2023) (Chinese), containing 50k documents per language.

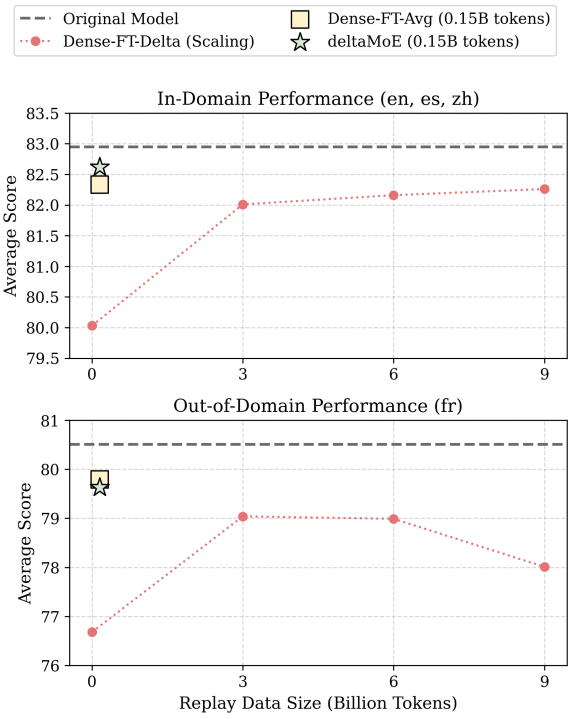

Figure 4: Knowledge retention performance with data replay. The dashed line indicates the original model's performance.

get, the router-tuning enables DeltaMoE to surpass Dense-FT-Avg's retention on in-domain languages. For the OOD language, it achieves highly competitive performance, closely matching the retention of Dense-FT-Avg. Simultaneously, our method demonstrates superior architectural efficiency, outperforming a Dense-FT-Delta model even when the latter is scaled with 60× more replay data in both in-domain and OOD languages. This confirms that our anchored MoE architecture, combined with router tuning, provides a more efficient solution for knowledge preservation.

4.3 Ablation on MoE Merging Strategy

In order to validate the necessity of our delta merging strategy when combined with the MoE architecture, we compare against a baseline, MoE-CPT-Avg, which starts from the dense instruct model for CPT, but then applies linear averaging by merging all post-CPT MoE parameters with their counterparts in the original instruct model.

The results in Table 2 show that the averaging strategy fails to resolve the fundamental performance trade-off, even within the MoE architecture. MoE-CPT-Avg nearly perfectly preserves original capabilities, but at the cost of severely reduced gains in expanded languages, lagging more than 4 points behind DeltaMoE . This confirms that averaging inherently dilutes the crucial knowledge gained during CPT. In contrast, our delta-based approach shows a powerful synergy: the frozen expert anchors original knowledge, while delta merging transfers alignment abilities without dilution, yielding a far superior overall balance.

Merging Strategy	Expanded	Original	Avg.
Qwen2.5-7B-Instruct	59.83	82.33	71.08
MoE-CPT-Avg	70.93	**81.35**	76.14
DeltaMoE (Ours)	**74.97**	81.17	**78.07**

Table 2: Performance comparison of different merging strategies on the MoE architecture.

5 Generalization Analysis

To demonstrate the generalizability of DeltaMoE , we conduct experiments across two dimensions: using an alignment delta (Δ_{instruct}) from a different source, and applying our framework to the LLaMA-3.1-8B model family. These results confirm that our approach is not limited to a specific post-training pipeline or base architecture.

5.1 Generalization to a Different Alignment Delta

To verify that DeltaMoE is not dependent on a specific post-training pipeline, we expanded our generalization analysis. We created a new delta Tulu-Delta by performing SFT on the Qwen2.5-7B base model using the Tulu3 dataset mixture (Lambert et al., 2024)[4]. This delta was then applied to DeltaMoE and a comprehensive set of baselines.

The results, summarized in Table 3, reaffirm the robustness of our framework. DeltaMoE once again delivers the best overall performance, achieving SOTA performance on the expanded languages and competitive results that rival the top-performing baseline on the original languages. This confirms that our framework is agnostic to the source of the alignment delta and robustly transfers alignment abilities more effectively than alternative merging strategies.

5.2 Generalization on different model

To further assess the generality of our approach across model architecture, we replicate the entire experimental pipeline on a different, widely-used

[4] https://huggingface.co/datasets/allenai/tulu-3-sft-mixture

Model	Expanded	Original
Base Model: Qwen2.5-7B, Delta: Tulu3		
Qwen-7B + Tulu-Delta	53.85	79.90
Baselines w/ Same CPT Data		
Dense-FT-Avg	61.55	**76.46**
Dense-FT-Delta	62.25	72.60
Baselines w/ Matched FLOPs (2x CPT Data)		
Dense-FT-Avg-2FLOPs	58.32	75.27
Dense-FT-Delta-2FLOPs	59.64	68.92
Baselines w/ Matched Parameters		
LLaMA-Pro	64.97	75.49
MoLA	64.25	74.59
DeltaMoE	**65.77**	76.14

Table 3: Generalization results using a delta derived from the Tulu3 SFT dataset. For LLaMA-Pro, we use its best variant in Appendix F.

Model	Expanded	Original
Base Model: LLaMA-3.1-8B		
LLaMA-3.1-8B-Instruct	64.91	80.37
Baselines w/ Same CPT Data		
Dense-FT-Avg	66.39	76.54
Dense-FT-Delta	66.88	74.37
Baselines w/ Matched FLOPs		
Dense-FT-Avg-2FLOPs	65.81	75.78
Dense-FT-Delta-2FLOPs	64.38	71.62
Baseline w/ Matched Parameters		
LLaMA-Pro	58.92	65.78
MoLA	68.39	74.21
DeltaMoE	**69.37**	**77.32**

Table 4: Generalization results on the LLaMA-3.1-8B model family. For LLaMA-Pro, we use its strongest variant as determined by Appendix G

family of models LLaMA-3.1-8B (Grattafiori et al., 2024).

As shown in Table 4, the findings are consistent with our main results. DeltaMoE delivers the most favorable trade-off, achieving SOTA performance on both expanded and original languages. This result strongly indicates that DeltaMoE serves as a general and robust strategy for enhancing performance on expanded languages while preserving strong capabilities in the original ones across various foundational models.

6 Related Work

6.1 Mixture of Experts

The MoE architecture enables efficient scaling of LLMs by activating only a subset of parameters per token (Du et al., 2022; Lepikhin et al., 2021; Zoph et al., 2022). This paradigm allows for models with massive parameter counts while maintaining a fixed inference budget. Recent advancements have further refined MoE through techniques like shared and fine-grained experts (Guo et al., 2025), zero-expert (Jin et al., 2025) which dynamiclly control the activated parameters, as well as shortcut-connected mechanisms that optimize inference speed (Cai et al., 2025).

The MoE architecture is increasingly being adopted for multilingual CPT. Early approaches in this area, such as MoE-LPR (Zhou et al., 2025), established a two-stage training process to balance performance across original and expansion languages. More recent methods, including DMoE (Li et al., 2025) and LayerMoE (Zhang et al., 2025), have refined this concept by dynamically allocating experts based on linguistic similarity. However, these methods focus solely on expanding the base model, while our work presents a more holistic solution that integrates the subsequent transfer of alignment abilities.

6.2 Model Merging

Model merging is a data-free method for combining capabilities from multiple specialized models(Yu et al., 2024; Yadav et al., 2023). Commonly, it is used to resolve task conflicts in post-training, such as merging specialized skills (Yadav et al., 2023; Ma et al., 2025; Wu et al., 2025; Dang et al., 2024).

In the context of CPT, merging has been repurposed to transfer alignment capabilities from a public model to a continually trained multilingual base model. This application, however, introduces a critical trade-off: simple averaging dilutes newly learned knowledge (Yamaguchi et al., 2024), while delta-based methods can cause catastrophic forgetting of original abilities (Cao et al., 2025). Our work introduces a novel merging strategy tailored for MoE models that directly resolves this trade-off.

7 Conclusion

This paper presented DeltaMoE , a framework that resolves trade-off by first creating new experts in an MoE architecture while freezing all original parameters during CPT, then grafting an alignment delta (Δ_{post}). Experiments confirm DeltaMoE significantly enhances expansion language performance while mitigating catastrophic forgetting, proving effective across diverse models and alignment deltas.

Ultimately, DeltaMoE offers a practical and scalable pathway for extending the multilingual alignment of existing LLMs.

Limitations

While DeltaMoE effectively resolves the core trade-off between acquiring new languages and retaining original capabilities, this work has two primary limitations. First, the evaluated languages and benchmarks, while substantial, are insufficient to fully represent the global linguistic diversity and task spectrum. Second, the MoE architecture introduces non-trivial computational overhead in both training and inference compared to dense models.

Acknowledgments

We would like to thank the anonymous reviewers for their insightful comments. Shujian Huang and Hao-Ran Wei are the co-corresponding authors. This work is supported by National Science Foundation of China (No. 62376116), research project of Nanjing University-China Mobile Joint Institute (NJ20250038), the Fundamental Research Funds for the Central Universities (No. 2024300507).

References

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2023. The belebele benchmark: a parallel reading comprehension dataset in 122 language variants. *arXiv preprint arXiv:2308.16884*.

Weilin Cai, Juyong Jiang, Le Qin, junweicui, Sunghun Kim, and Jiayi Huang. 2025. Shortcut-connected expert parallelism for accelerating mixture of experts. In *Forty-second International Conference on Machine Learning*.

Sheng Cao, Mingrui Wu, Karthik Prasad, Yuandong Tian, and Zechun Liu. 2025. Paramδ for direct mixing: Post-train large language model at zero cost. In *The Thirteenth International Conference on Learning Representations*.

Marta R Costa-Jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, and 1 others. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

John Dang, Shivalika Singh, Daniel D'souza, Arash Ahmadian, Alejandro Salamanca, Madeline Smith, Aidan Peppin, Sungjin Hong, Manoj Govindassamy, Terrence Zhao, and 1 others. 2024. Aya expanse: Combining research breakthroughs for a new multilingual frontier. *arXiv preprint arXiv:2412.04261*.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, and 1 others. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In *International conference on machine learning*, pages 5547–5569. PMLR.

Chongyang Gao, Kezhen Chen, Jinmeng Rao, Baochen Sun, Ruibo Liu, Daiyi Peng, Yawen Zhang, Xiaoyuan Guo, Jie Yang, and VS Subrahmanian. 2024. Higher layers need more lora experts. *arXiv preprint arXiv:2402.08562*.

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

Nuno M Guerreiro, Ricardo Rei, Daan van Stigt, Luisa Coheur, Pierre Colombo, and André FT Martins. 2024. xcomet: Transparent machine translation evaluation through fine-grained error detection. *Transactions of the Association for Computational Linguistics*, 12:979–995.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Xu Huang, Wenhao Zhu, Hanxu Hu, Conghui He, Lei Li, Shujian Huang, and Fei Yuan. 2025. Benchmax: A comprehensive multilingual evaluation suite for large language models. *arXiv preprint arXiv:2502.07346*.

Alexandra Institute. 2025. m_mmlu (revision 18e6c8e).

Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. 2024. A survey on large language models for code generation. *arXiv preprint arXiv:2406.00515*.

Peng Jin, Bo Zhu, Li Yuan, and Shuicheng YAN. 2025. Moe++: Accelerating mixture-of-experts methods with zero-computation experts. In *The Thirteenth International Conference on Learning Representations*.

Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay, Mostafa Dehghani, and Neil Houlsby. 2023. Sparse upcycling: Training mixture-of-experts from dense checkpoints. In *The Eleventh International Conference on Learning Representations*.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. {GS}hard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*.

Chong Li, Yingzhuo Deng, Jiajun Zhang, and Chengqing Zong. 2025. Group then scale: Dynamic mixture-of-experts multilingual language model. *arXiv preprint arXiv:2506.12388*.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Guha, Sedrick Scott Keh, Kushal Arora, and 1 others. 2024. Datacomp-lm: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37:14200–14282.

Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, and 1 others. 2025. Large language model agent: A survey on methodology, applications and challenges. *arXiv preprint arXiv:2503.21460*.

Qianli Ma, Dongrui Liu, Qian Chen, Linfeng Zhang, and Jing Shao. 2025. Led-merging: Mitigating safety-utility conflicts in model merging with location-election-disjoint. *arXiv preprint arXiv:2502.16770*.

Thuat Nguyen, Chien Van Nguyen, Viet Dac Lai, Hieu Man, Nghia Trung Ngo, Franck Dernoncourt, Ryan A Rossi, and Thien Huu Nguyen. 2023. Culturax: A cleaned, enormous, and multilingual dataset for large language models in 167 languages. *arXiv preprint arXiv:2309.09400*.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. 2025. Fineweb2: One pipeline to scale them all–adapting pre-training data processing to every language. *arXiv preprint arXiv:2506.20920*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025a. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Meituan LongCat Team, Bei Li, Bingye Lei, Bo Wang, Bolin Rong, Chao Wang, Chao Zhang, Chen Gao, Chen Zhang, Cheng Sun, and 1 others. 2025b. Longcat-flash technical report. *arXiv preprint arXiv:2509.01322*.

Peng-Yuan Wang, Tian-Shuo Liu, Chenyang Wang, Yi-Di Wang, Shu Yan, Cheng-Xing Jia, Xu-Hui Liu, Xin-Wei Chen, Jia-Cheng Xu, Ziniu Li, and 1 others. 2025. A survey on large language models for mathematical reasoning. *arXiv preprint arXiv:2506.08446*.

Yuxia Wang, Minghan Wang, Muhammad Arslan Manzoor, Fei Liu, Georgi Georgiev, Rocktim Das, and Preslav Nakov. 2024. Factuality of large language models: A survey. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 19519–19529.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, Chenxia Li, Liu Yang, Xilin Luo, Xuejie Wu, Lunan Liu, Wenjun Cheng, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, and 11 others. 2023. Skywork: A more open bilingual foundation model. *Preprint*, arXiv:2310.19341.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. Do llamas work in english? on the latent language of multilingual transformers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15366–15394.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and 1 others. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR.

Chengyue Wu, Yukang Gan, Yixiao Ge, Zeyu Lu, Jiahao Wang, Ye Feng, Ying Shan, and Ping Luo. 2024. Llama pro: Progressive llama with block expansion. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6518–6537.

Han Wu, Yuxuan Yao, Shuqi Liu, Zehua Liu, Xiaojin Fu, Xiongwei Han, Xing Li, Hui-Ling Zhen, Tao Zhong, and Mingxuan Yuan. 2025. Unlocking efficient long-to-short llm reasoning with model merging. *arXiv preprint arXiv:2503.20641*.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2023. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36:7093–7115.

Atsuki Yamaguchi, Terufumi Morishita, Aline Villavicencio, and Nikolaos Aletras. 2024. Elchat: Adapting chat language models using only target unlabeled language data. *arXiv preprint arXiv:2412.11704*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*.

Xue Zhang, Yunlong Liang, Fandong Meng, Songming Zhang, Yufeng Chen, Jinan Xu, and Jie Zhou. 2025. Less, but better: Efficient multilingual expansion for llms via layer-wise mixture-of-experts. *arXiv preprint arXiv:2505.22582*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.

Hao Zhou, Zhijun Wang, Shujian Huang, Xin Huang, Xue Han, Junlan Feng, Chao Deng, Weihua Luo, and Jiajun Chen. 2025. Moe-lpr: Multilingual extension of large language models through mixture-of-experts with language priors routing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 26092–26100.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. 2022. St-moe: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*.

A Detailed Benchmark Results

This section provides a comprehensive breakdown of the zero-shot performance for all models on each benchmark. Tables 5 through 10 detail the scores.

B Baselines CPT Hyperparameter Settings

Table 11 provides a detailed summary of the key hyperparameters used during the CPT stage for all baseline models presented in the main experiments.

C Evaluation Benchmark and Prompting Details

We provide a detailed description of the benchmarks and prompting strategies used to evaluate our models in a zero-shot setting.

C.1 Benchmark Descriptions

- **MGSM** (Huang et al., 2025): A multilingual benchmark for grade-school mathematical reasoning. We adopt the standard zero-shot chain-of-thought prompting strategy from the original paper and report accuracy.
- **MIFEVAL** (Huang et al., 2025): A benchmark designed to test a model's adherence to complex and nuanced instructions in a multilingual context. We use the prompts and evaluation scripts provided by the authors. Following standard practice (Grattafiori et al., 2024), we report the overall score, which averages four sub-metrics (prompt-strict, prompt-loose, instruction-strict, and instruction-loose).
- **FLORES-200** (Costa-Jussà et al., 2022): A large-scale benchmark for machine translation. We use a custom prompt format to ensure the model directly outputs the translated text. The template is as follows in Figure 5. Performance is measured using the reference-free XCOMET-XXL metric (Guerreiro et al., 2024)[5] for both English-to-target (En-XX) and target-to-English (XX-En) directions.
- **BELEBELE** (Bandarkar et al., 2023): A massively multilingual reading comprehension dataset. We report zero-shot accuracy using the chain-of-thought prompt detailed in Section C.2.
- **M_MMLU** (Institute, 2025): A multilingual version of the MMLU benchmark for general knowledge. We report zero-shot accuracy using the chain-of-thought prompt detailed in Section C.2. To maintain representativeness while reducing computational overhead, we evaluate on a stratified subset created by sampling 10% of questions from each subject category.

```
Translate the following text from
{src_lang} to {tgt_lang}. Only output the
translation without any additional text.

{src_lang} source:{src_sentence}

{tgt_lang} translation:
```

Figure 5: The prompt for flores evaluation

```
Answer the following multiple choice question.
The last line of your response should be of the
following format: 'Answer: $LETTER' (without
quotes) where LETTER is one of ABCD. Think step
by step before answering.

{question}

A) {A}
B) {B}
C) {C}
D) {D}
```

Figure 6: The prompt for mmlu evaluation

C.2 Multiple-Choice Question Prompting and Extraction

For the multiple-choice question (MCQ) benchmarks (M_MMLU and BELEBELE), we employ a unified chain-of-thought prompting strategy to encourage step-by-step reasoning in zero-shot setting.

Prompt Translation and Structure. We adapted the English prompt template from the OpenAI simple-evals repository[6]. To create multilingual versions, the English template was translated into each target language using the DeepSeek-V3 (Guo et al., 2025) model. An example of the English MMLU prompt is shown in Figure 6:

The BELEBELE prompt is similar but includes a passage field for the context. Multilingual versions follow the same structure, translated accordingly.

Robust Answer Extraction. We implement a two-stage process for robust answer extraction from

[5] https://huggingface.co/Unbabel/XCOMET-XXL

[6] https://github.com/openai/simple-evals

	Original Languages					Expanded Languages			
Model	**en**	**es**	**zh**	**fr**	**Avg.**	**hu**	**bn**	**sr**	**Avg.**
MGSM									
Qwen2.5-7B-Instruct	80.80	72.00	73.60	63.60	72.50	61.20	58.40	61.60	60.40
Baselines w/ Same CPT Data									
Dense-FT-Avg	80.40	64.40	75.60	64.40	71.20	64.40	64.40	69.60	66.13
Dense-FT-Delta	78.00	67.20	73.60	54.00	68.2	56.80	63.20	65.60	61.87
Baselines w/ Matched FLOPs (2x CPT Data)									
Dense-FT-Avg-2FLOPs	80.00	70.80	78.00	68.40	74.30	56.80	62.00	67.60	62.13
Dense-FT-Delta-2FLOPs	76.00	61.60	72.40	58.00	67.00	62.00	61.20	70.80	64.67
Baselines w/ Matched Parameters									
LLaMA-Pro	77.20	72.00	69.20	58.80	69.30	60.40	53.60	68.40	60.80
MoLA	79.60	65.20	70.80	59.60	68.80	63.20	66.00	67.20	65.47
DeltaMoE	82.40	71.60	74.40	63.60	73.00	65.20	65.60	72.00	67.60

Table 5: Detailed per-language results on the MGSM benchmark.

	Original Languages					Expanded Languages			
Model	**en**	**es**	**zh**	**fr**	**Avg.**	**hu**	**bn**	**sr**	**Avg.**
MIFEval									
Qwen2.5-7B-Instruct	79.40	77.46	74.72	77.50	77.27	58.91	57.78	61.12	59.27
Baselines w/ Same CPT Data									
Dense-FT-Avg	77.76	73.22	67.40	73.37	72.94	61.48	51.45	63.53	58.82
Dense-FT-Delta	77.87	69.93	69.89	74.62	73.08	60.98	53.01	68.40	60.80
Baselines w/ Matched FLOPs (2x CPT Data)									
Dense-FT-Avg-2FLOPs	78.25	72.53	67.05	72.49	72.58	55.47	52.80	61.95	56.74
Dense-FT-Delta-2FLOPs	77.13	69.34	69.89	72.77	72.28	64.67	56.87	65.94	62.49
Baselines w/ Matched Parameters									
LLaMA-Pro	71.73	66.62	64.64	69.53	68.13	60.20	50.40	65.58	58.72
MoLA	78.01	72.79	71.69	72.59	73.77	62.98	52.85	68.50	61.44
DeltaMoE	76.37	73.84	69.93	73.72	73.47	68.54	55.90	70.40	64.95

Table 6: Detailed per-language results on the MIFEval benchmark.

	Original Languages					Expanded Languages			
Model	**en**	**es**	**zh**	**fr**	**Avg.**	**hu**	**bn**	**sr**	**Avg.**
Belebele									
Qwen2.5-7B-Instruct	93.11	89.44	89.00	90.33	90.47	70.33	67.67	77.67	71.89
Baselines w/ Same CPT Data									
Dense-FT-Avg	91.78	88.00	88.44	90.00	89.56	76.33	72.44	81.78	76.85
Dense-FT-Delta	91.67	87.22	88.67	87.56	88.78	84.22	71.22	84.56	80.00
Baselines w/ Matched FLOPs (2x CPT Data)									
Dense-FT-Avg-2FLOPs	90.89	86.56	86.78	88.44	88.17	74.11	72.67	82.67	76.48
Dense-FT-Delta-2FLOPs	92.00	84.89	87.11	85.89	87.47	83.89	75.33	85.11	81.44
Baselines w/ Matched Parameters									
LLaMA-Pro	91.11	85.78	87.89	87.44	88.06	85.00	74.78	85.33	81.70
MoLA	92.33	85.89	87.78	88.00	88.50	83.89	72.89	85.44	80.74
DeltaMoE	93.44	88.33	88.44	87.44	89.42	85.00	74.78	86.89	82.22

Table 7: Detailed per-language results on the BELEBELE benchmark.

model outputs.

1. **Regex Extraction:** We first apply a regular expression to parse the final line of the model's generation, searching for the pattern 'Answer: [A-D]'.

2. **Model-based Extraction Fallback:** For outputs where regex parsing fails, we employ Qwen3-4B-Instruct[7] as a fallback extractor. Prompted as detailed in Figure 7, it identifies the chosen option (A-D) or reports ambiguity ('Z'), ensuring robust answer parsing across varied response formats.

[7] https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507

Model	Original Languages					Expanded Languages			
	en	es	zh	fr	Avg.	hu	bn	sr	Avg.
M_MMLU									
Qwen2.5-7B-Instruct	73.80	68.37	68.11	67.71	69.50	49.00	39.82	53.03	47.29
Baselines w/ Same CPT Data									
Dense-FT-Avg	73.34	68.75	66.44	66.18	68.68	56.00	41.54	58.50	52.01
Dense-FT-Delta	71.61	63.71	64.24	63.20	65.69	58.00	44.07	61.38	54.48
Baselines w/ Matched FLOPs (2x CPT Data)									
Dense-FT-Avg-2FLOPs	73.12	66.12	66.36	66.26	67.96	51.15	42.11	58.73	50.66
Dense-FT-Delta-2FLOPs	68.45	62.28	60.98	62.97	63.67	57.46	45.38	61.08	54.64
Baselines w/ Matched Parameters									
LLaMA-Pro	70.41	63.41	62.80	62.82	64.86	59.69	44.56	60.77	55.01
MoLA	70.18	61.16	63.79	62.59	64.43	57.46	44.73	61.38	54.52
DeltaMoE	73.04	64.91	66.29	64.73	67.24	59.31	46.28	62.75	56.11

Table 8: Detailed per-language results on the M_MMLU benchmark.

Model	Original Languages				Expanded Languages			
	en-es	en-zh	en-fr	Avg.	en-hu	en-bn	en-sr	Avg.
Flores-200 (En-XX)								
Qwen2.5-7B-Instruct	91.93	88.89	89.00	89.94	34.04	40.44	46.44	40.31
Baselines w/ Same CPT Data								
Dense-FT-Avg	92.68	89.70	90.22	90.87	79.45	70.66	80.02	76.71
Dense-FT-Delta	90.08	83.00	86.89	86.66	88.31	79.00	85.10	84.14
Baselines w/ Matched FLOPs (2x CPT Data)								
Dense-FT-Avg-2FLOPs	92.54	89.04	89.65	90.41	80.14	73.27	81.95	78.45
Dense-FT-Delta-2FLOPs	88.53	80.05	85.05	84.55	89.89	80.72	84.92	85.18
Baselines w/ Matched Parameters								
LLaMA-Pro	91.53	87.49	88.12	89.04	91.01	82.11	74.57	82.56
MoLA	91.10	88.20	87.98	89.09	90.56	82.54	88.97	87.36
DeltaMoE	92.22	88.49	88.92	89.88	91.95	84.21	88.24	88.13

Table 9: Detailed per-language results for Flores-200 (En-XX) translation.

Model	Original Languages				Expanded Languages			
	es-en	zh-en	fr-en	Avg.	hu-en	bn-en	sr-en	Avg.
Flores-200 (XX-En)								
Qwen2.5-7B-Instruct	94.09	93.99	94.91	94.33	77.72	77.17	84.62	79.84
Baselines w/ Same CPT Data								
Dense-FT-Avg	94.14	95.74	94.98	94.95	91.44	86.25	92.35	90.01
Dense-FT-Delta	92.78	92.06	93.83	92.89	90.92	84.65	91.76	89.11
Baselines w/ Matched FLOPs (2x CPT Data)								
Dense-FT-Avg-2FLOPs	94.09	95.47	95.02	94.86	92.02	86.41	92.72	90.38
Dense-FT-Delta-2FLOPs	92.87	91.29	93.54	92.57	91.64	85.82	91.75	89.73
Baselines w/ Matched Parameters								
LLaMA-Pro	92.05	93.47	92.53	92.68	88.91	84.13	87.42	86.82
MoLA	92.99	92.63	93.81	93.15	91.83	85.31	92.08	89.74
DeltaMoE	93.64	94.05	94.43	94.04	92.53	87.25	92.69	90.82

Table 10: Detailed per-language results for Flores-200 (XX-En) translation.

D Router-Tuning Replay Strategy

To enhance knowledge retention for DeltaMoE , we implemented a brief, router-tuning phase. In this stage, only the router parameters of the MoE layers were trained. The training data consisted of a 0.15B original languages token corpus, with a 1:2 ratio of original language data to expansion language data. This process allows the router to refine its ability to correctly allocate tokens from original languages to the frozen expert.

E Ablation on LLaMA-Pro Configuration

To ensure a fair and robust comparison against the LLaMA-Pro baseline, we conducted an ablation

Model / Group	Learning Rate	Global Batch Size	CPT Data (Tokens)
Dense Merging Baselines			
Dense-FT (Avg & Delta)	2e-5	512	9B
Dense-FT-2FLOPs (Avg & Delta)	2e-5	512	18B
Parameter-Expansion Baselines			
LLaMA-Pro	2e-4	512	9B
MoLA	5e-5	512	9B

Table 11: Hyperparameter settings for the CPT stage. All baselines were trained for 1 epoch using the AdamW optimizer and a cosine learning rate scheduler.

```
You are an AI assistant designed to analyze the output of another AI model and extract the multiple-choice option (A, B, C, or D)
it selects. Follow these steps:

1. **Input:** You will receive a text output from a model (possibly in different languages) regarding a multiple-choice question.
2. **Task:** Determine whether the model explicitly or implicitly chose an option (A, B, C, or D). If no option can be clearly
determined, output `<answer>Z</answer>`.
3. **Requirements:**
- Focus solely on the first occurrence of a valid option (Latin letters A, B, C, or D) in the text.
- Ignore any non-Latin characters or symbols (e.g., checkmarks, brackets). Only Latin letters A, B, C, D are valid.
- If the text references an option indirectly (e.g., "the correct answer is A"), extract the option.
- If the output is very short (e.g., just "B" or "选项是B"), extract the option directly.
- If the choice is ambiguous, unclear, or no option is detected, output `<answer>Z</answer>`.
4. **Output Format:** After a brief reasoning, output exactly in the format:
`<answer>X</answer>`
where `X` is the extracted option (e.g., A, B, C, D) or "Z" if no clear option can be determined.

**Examples:**
- Input: "A szondát felbocsátó keringő műholdnak nem volt legénysége, ezért az A válasz a helyes." Reasoning: The text explicitly
mentions "A válasz" (meaning "A answer"). Output: `<answer>A</answer>`
- Input: "The passage states that the Three Kingdoms era was one of the bloodiest eras in Ancient China’s history. Therefore, the
answer is D." Reasoning: The phrase "the answer is D" indicates option D. Output: `<answer>D</answer>`
- Input: "B) সম্ভাবনা মূলক নমুনা না নেওয়া" Reasoning: The text starts with "B)" which directly indicates option B. Output:
`<answer>B</answer>`
- Input: "This is a general comment without any option specified." Reasoning: No option (A, B, C, D) is mentioned or can be clearly
determined. Output: `<answer>Z</answer>`
- Input: "I think both A and C could be correct." Reasoning: The response is ambiguous and no single clear option is chosen. Output:
`<answer>Z</answer>`
**Now process the following input:**
{}
Begin with a brief reasoning (1-2 sentences), then output strictly as `<answer>X</answer>`.
"""
```

Figure 7: The prompt used for model-based answer extraction

study to determine the optimal number of newly added Transformer blocks. We experimented with adding 7, 14, and 28 blocks to the base 28-layer Qwen2.5-7B architecture. This resulted in models with total parameter counts of 9.2B, 10.8B, and 12.4B, respectively.

As shown in Table 12, the model with 14 added layers (10.8B total parameters) achieved the best overall performance, maximizing gains on expanded languages while maintaining strong performance on the original languages.

F Ablation on LLaMA-Pro with Tulu Delta

Similar to our main experiments, we performed an ablation study to identify the strongest LLaMA-Pro configuration when using the Tulu alignment delta. We tested models with 7, 14, and 28 added layers.

The results, presented in Table 13, show a different trend compared to our primary experiments. In this scenario, adding only 7 layers yielded the best performance. Consequently, we selected the LLaMA-Pro (+7 layers) variant as the baseline for our generalization analysis in Section 5.1.

G Ablation on LLaMA-Pro for LLaMA-3.1

To identify the strongest LLaMA-Pro baseline for the LLaMA-3.1-8B model family, we performed an ablation study on the number of added Transformer blocks. We tested variants with 8, 16, 32 added layers.

The results, presented in Table 14, show a different trend compared to our primary experiments. In this scenario, adding only 8 layers yielded the best performance.

Model Configuration	Total Params (B)	Avg. (Expanded)	Avg. (Original)
Qwen2.5-7B-Instruct	*7.6*	*59.83*	*82.33*
LLaMA-Pro (+7 layers)	9.2	66.81	75.86
LLaMA-Pro (+14 layers)	**10.8**	**70.07**	**79.72**
LLaMA-Pro (+28 layers)	12.4	24.60	8.66

Table 12: Ablation on the number of added layers for the LLaMA-Pro baseline in Qwen2.5-7B.

Model Configuration	Total Params (B)	Avg. (Expanded)	Avg. (Original)
Qwen-7B + Tulu-Delta	*7.6*	*53.85*	*79.90*
LLaMA-Pro (+7 layers)	**9.2**	**64.97**	**75.49**
LLaMA-Pro (+14 layers)	10.8	32.71	38.96
LLaMA-Pro (+28 layers)	12.4	22.36	8.34

Table 13: Ablation on the number of added layers for the LLaMA-Pro baseline using the Tulu delta.

Model Configuration	Total Params (B)	Avg. (Expanded)	Avg. (Original)
LLaMA-3.1-8B-Instruct	*8.0*	*64.91*	*80.37*
LLaMA-Pro (+8 layers)	9.7	**58.92**	65.78
LLaMA-Pro (+16 layers)	11.5	44.44	**66.49**
LLaMA-Pro (+24 layers)	15	14.80	13.87

Table 14: Ablation on the number of added layers for the LLaMA-Pro baseline in LLaMA3.1-8B.

H Expert Capacity and Language Scalability

In this section, we provides additional experiments to demonstrate that a small number of newly added trainable experts are sufficient to handle multiple expanded languages.

Expert count does not need to scale with language count. To investigate whether more experts are needed as additional languages are introduced, we fix the number of expanded languages to 5 (Hungarian, Bengali, Serbian, Telugu, and Icelandic, with 1B tokens each) and compare our standard 4-expert setup (1 frozen + 3 trainable) against a 6-expert setup (1 frozen + 5 trainable). As shown in Table 15, increasing the number of experts yields negligible performance improvement, indicating that 3 trainable experts already provide sufficient capacity for this setting.

Scaling languages does not cause interference. We further examine whether expanding to more languages causes interference among learned languages. Keeping the expert count fixed at 4, we increased the number of expanded languages from three (Hungarian, Bengali, Serbian; totaling 9 billion tokens) to eight (adding Czech, Telugu, Icelandic, Greek, and Turkish; 2 billion tokens each, amounting to 19 billion tokens in total). We evaluate on the original 3 expanded languages (hu, bn, sr) to directly measure interference, and compare against the Dense-FT-Delta baseline trained on the same data.

As shown in Table 16, performance on the original 3 languages remains highly stable when scaling to 8 languages, and DeltaMoE continues to substantially outperform the Dense-FT-Delta baseline. These results demonstrate that the trainable experts can accommodate an increasing number of languages without causing significant interference.

Model	**Experts**	Original	Expanded (5-lang)	Avg.
Qwen2.5-7B-Instruct	—	82.33	51.81	67.07
DeltaMoE	4 (1+3)	81.26	70.07	75.67
DeltaMoE	6 (1+5)	81.03	70.66	75.84

Table 15: Performance comparison when scaling the number of experts for 5 expanded languages. The marginal gain from increasing expert count confirms that 3 trainable experts are sufficient.

Model	**Training Setup**	Original	Expanded (Initial 3)	Avg.
Qwen2.5-7B-Instruct	—	82.33	59.83	71.08
DeltaMoE	3 langs (9B)	81.17	74.97	78.07
DeltaMoE	8 langs (19B)	81.04	74.43	77.74
Dense-FT-Delta	8 langs (19B)	78.21	70.00	74.10

Table 16: Performance on the original 3 expanded languages when scaling to 8 languages with a fixed 4-expert setup. DeltaMoE retains strong performance on previously learned languages and outperforms Dense-FT-Delta under identical data conditions.