MTA:A Merge-then-Adapt Framework for Personalized Large Language Models

Anonymous ACL submission

Abstract

Personalized Large Language Models (PLLMs) aim to align model outputs with individual user preferences, a crucial capability for user-centric applications. However, the prevalent approach of fine-tuning a separate module for each user faces two major limitations: (1) storage costs scale linearly with the number of users, rendering the method unscalable; and (2) fine-tuning a static model from scratch often yields suboptimal performance for users with sparse data. To address these challenges, we propose **MTA**, a Merge-then-Adapt framework for PLLMs. MTA comprises three key stages. First, we construct a shared *Meta-LoRA Bank* by selecting anchor users and pre-training meta-personalization traits within meta-LoRA modules. Second, to ensure scalability and enable dynamic personalization combination beyond static models, we introduce an *Adaptive LoRA Fusion* stage. This stage retrieves and dynamically merges the most relevant anchor meta-LoRAs to synthesize a user-specific adapter on the fly, thereby removing the need to maintain a dedicated, persistently stored per-user adapter for each user and enabling flexible personalization. Third, we propose a *LoRA Stacking for Few-Shot Personalization* stage, which optionally applies an additional ultra-low-rank, lightweight residual LoRA module on top of the merged LoRA. This stacked module captures user-specific residual signals under few-shot settings. Extensive experiments on the LaMP benchmark demonstrate that our approach outperforms existing SOTA methods across multiple tasks. Our code is available at `https://anonymous.4open.science/r/MTA-ACL`.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in semantic understanding and text generation across a wide range of tasks. Despite their general success, most current LLMs follow a "one-size-fits-all" paradigm during generation, which overlooks user-specific preferences and contextual alignment. This limitation significantly hinders their applicability in domains that inherently require personalized interactions, such as healthcare (Johnson et al., 2021), education (Pratama et al., 2023), personalized customer service (Ma et al., 2021; Li et al., 2025), and recommendation systems (Li et al., 2023; Gao et al., 2024; Zhao et al., 2025), etc. Personalized Large Language Models (PLLMs), which tailor their outputs to user-specific preferences, have emerged as a timely and critical research direction for bridging the gap between general-purpose models and individualized user needs.

Early research on PLLMs incorporates explicit user information through prompt-based methods, employing in-context learning (Liu et al., 2024), retrieval augmentation (Mysore et al., 2023), or profile augmentation (Richardson et al., 2023). However, these approaches are highly sensitive to input noise (Tan et al., 2024a) and pose potential risks of privacy leakage (Zhang et al., 2025). In contrast, recent research has increasingly focused on fine-tuning-based methods, particularly Parameter-Efficient Fine-Tuning (PEFT) methods such as LoRA (Hu et al., 2022), which implicitly encode personalization within model parameters. These approaches enable fine-grained alignment to user preferences while mitigating privacy exposure. OPPU (Tan et al., 2024b) is the first to introduce a one-LoRA-per-user paradigm, in which user preferences and behavioral patterns are encoded within personal parameters, resulting in superior generation quality. Building on this method, Per-Pcs (Tan et al., 2024a) decomposes complete LoRA modules into smaller components and introduces a dynamic routing mechanism that recombines these elements during inference, thereby improving both personalization adaptability and computational efficiency. PROPER (Woźniak et al., 2024) employs a hier-

archical modeling framework that operates across population, group, and user levels to enable fine-grained personalized alignment.

However, these approaches face two key limitations: *(1). Poor Scalability*: The one-LoRA-per-user approach necessitates the independent training and storage of LoRA modules for each user, resulting in parameter growth scaling linearly with the number of users, rendering it impractical for large-scale deployment. Although Per-Pcs partially alleviates this issue through component recombination, it still relies on a static assembly of pre-trained LoRAs and lacks the adaptability necessary to tailor representations to target users. *(2). Limited performance in sparse-data scenarios.* For OPPU and Per-Pcs, these methods require sufficient training data to achieve parameter convergence and to enable personalized alignment. However, their performance degrades in data-sparse settings, where alignment cannot be reliably established. While PROPER mitigates this issue by incorporating population and group-level training, its multi-level progressive learning pipeline increases procedural complexity and poses challenges for scalability.

To address these challenges, we propose **MTA**, a three-stage framework employing "Merge-then-Adapt" to enable scalable and data-efficient personalization. Firstly, we introduce the module of **Meta-LoRA Bank Construction**. We pre-train a set of meta-LoRAs by selecting anchor users with highly distinguishable personalization traits. Secondly, we introduce **Adaptive LoRA Merging**. Based on the similarity between the meta-LoRAs and the target user profile, multiple meta-LoRAs are retrieved. Their parameters are then dynamically merged according to their relevance to the target user. In this way, personalization goes beyond the static one-LoRA-per-user paradigm, achieving scalability through an infinite user-adaptive linear combination mechanism. Thirdly, we introduce **LoRA Stacking for Few-shot Personalization**. To enable personalization under sparse-data conditions, an ultra-low-rank LoRA is stacked atop the merged LoRA from the previous stage. During this phase, both the base LLM and the merged LoRA are kept frozen, only the newly added ultra-low-rank LoRA is updated. Owing to its minimal parameter size, this strategy enables efficient and robust adaptation even when training samples are extremely limited. We conduct comprehensive experiments across LaMP benchmarks (Salemi et al., 2023) under five different user personalization settings to validate the effectiveness, scalability, and efficiency of our proposed approach. Our contributions are summarized as follows:

- We present MTA, a novel framework that departs from the conventional static one-LoRA-per-user paradigm. MTA builds a meta-LoRA bank and employs a dynamic adapter-merging mechanism, delivering both scalability and effective personalized adaptation.
- To enhance robustness in sparse-data regimes, a LoRA-Stack training strategy is utilized to enable efficient and stable optimization under data-scarce conditions.
- Comprehensive experiments on the LAMP benchmark show that our approach achieves state-of-the-art performance, outperforming existing PEFT methods on PLLMs.

2 Preliminaries

Problem Formulation. Following previous studies on PLLMs (Tan et al., 2024b,a), our primary objective is to generate a user-specific response $\mathbf{r}_u$ for a given user u using a PLLM parameterized by Θ. The response generation is conditioned on the user's query $\mathbf{q}_u$ and their historical behavior data H_u.

Our framework reformulates the task by decomposing parameter Θ into two components. The first component, Θ^{merged}, represents the collaboratively-informed base model constructed by merging modules from a Meta-LoRA Bank to provide a robust personalized foundation. The second component, Θ^{adapt}, is a lightweight, low-rank adapter trained on individual user data to capture fine-grained preferences, with the final personalized model integrating both components as follows:

$$\mathbf{r}_u = \text{LLM}(\mathbf{q}_u; \Theta^{\text{merged}} \oplus \Theta^{\text{adapt}}) \quad (1)$$

where $\oplus$ denotes the additive combination of parameters such that the collaborative knowledge from Θ^{merged} is augmented with user-specific adaptations from Θ^{adapt}, enabling both broad personalization capabilities and individual fine-tuning within a unified architecture.

3 Methodology

In this section, we provide an overview of our framework and detail each of its stages.

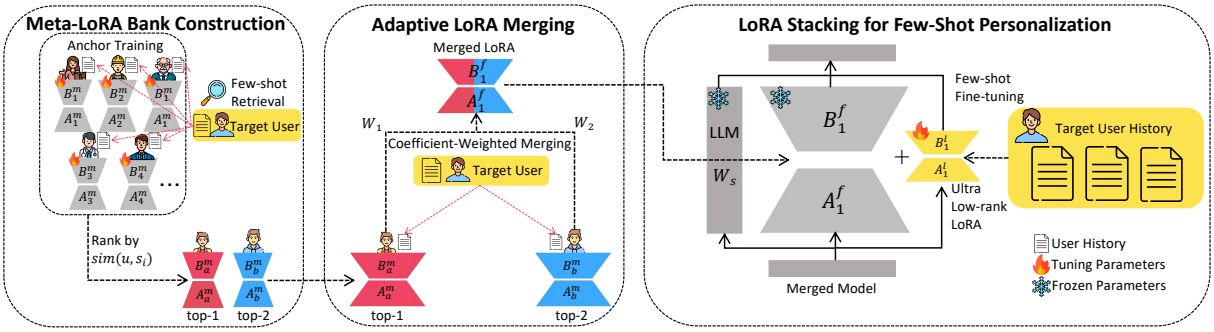

Figure 1: An overview of the **MTA** framework is as follows: The **left** panel, Meta-LoRA Bank Construction, shows the pre-training of a bank of anchor modules and the retrieval of the top two most relevant LoRAs for a target user. The **middle** panel, Adaptive LoRA Merging, shows the two retrieved LoRAs being combined through a coefficient-weighted merge to create a single, personalized LoRA. The **right** panel, LoRA Stacking for Few-Shot Personalization, depicts the freezing of the merged model and the fine-tuning of a new, ultra low-rank LoRA on top of it using the user's history for final adaptation.

3.1 Overall Framework

To overcome the critical limitations of poor scalability and ineffectiveness in data-sparse settings, we propose **MTA**, a Merge-then-Adapt framework, as shown in Figure 1. MTA consists of a three stages process:

- **Stage 1 (Meta-LoRA Bank Construction):** To build a bank of foundational modules with diverse personalization traits, we first generate embeddings for all users from their histories and partition them using K-Means clustering. The most active user from each cluster is selected as an "anchor user", and a unique LoRA module is fine-tuned on each one. This collection of anchor modules constitutes the Meta-LoRA Bank.

- **Stage 2 (Adaptive LoRA Merging):** To achieve scalability and move beyond the static one-LoRA-per-user paradigm, for a target user, we retrieve the most similar anchor LoRAs from the bank based on cosine similarity. Their corresponding modules are then combined using a weighted linear merge—with weights directly proportional to their similarity scores—to create a single, customized "Merged LoRA".

- **Stage 3 (LoRA Stacking for Few-Shot Personalization):** To enable robust and efficient adaptation in data-scarce settings, we follow a sequential **Merge-then-Adapt** strategy, the "Merged LoRA" is first irreversibly combined with the base model's weights. This entire model is then frozen. Finally, a new, extremely low-rank "stacked LoRA" is trained on top using the user's few-shot data, enabling a final, highly-efficient adaptation.

3.2 Meta-LoRA Bank Construction

The conventional one-LoRA-per-user paradigm exhibits $\mathcal{O}(n)$ computational complexity, necessitating linear scaling of both storage and training costs as user populations expand. To address this, we construct a fixed bank of $\mathcal{V}$ "experts" LoRAs with diverse yet representative personalization traits, distilled from the broader user population. This approach serves as the foundation for our MTA framework, where $\mathcal{V} \ll n$, effectively reducing computational complexity to $\mathcal{O}(\mathcal{V})$, while ensuring each expert represents a distinct preference profile. This first stage, therefore, focuses on constructing a compact yet representative Meta-LoRA Bank. This process begins by mapping each user's behavioral history, $H_u = \{h_1, ..., h_M\}$, into a dense vector space. We use a pre-trained DeBERTa-v3-Large (He et al., 2021) encoder to generate a corresponding embedding $\mathbf{e}_i$ for each history item h_i. The user's global preference embedding, $\boldsymbol{E}_u$, is then calculated as the average of all their item embeddings:

$$\boldsymbol{E}_u = \frac{1}{|H_u|} \sum_{\boldsymbol{h}_i \in H_u} \text{Encoder}(\boldsymbol{h}_i) \qquad (2)$$

After obtaining embeddings for all candidate users, we employ the K-Means algorithm to partition them into $\mathcal{V}$ disjoint clusters. To ensure each anchor user serves as a high-quality representative of their cluster's preferences, from each cluster $\mathcal{C}_v$, we select the most "active" user—the one with

the most extensive behavior history—as the anchor user u_v^*:

$$u_v^* = \arg\max_{u \in \mathcal{C}_v} |H_u| \tag{3}$$

Then, we pre-train a dedicated LoRA module for each selected anchor user using their historical data. This collection of modules, $\{\mathbf{\Theta}_{u_1^*}, ..., \mathbf{\Theta}_{u_V^*}\}$, constitutes our Meta-LoRA Bank to facilitate the following merging steps.

3.3 Adaptive LoRA Merging

After constructing the LoRA banks, maintaining the traditional rigid one-to-one mapping between users and LoRA modules would severely limit personalization capabilities and render the approach impractical for large-scale deployment. We propose **Adaptive LoRA Merging**, which dynamically combines parameters from multiple anchor LoRAs retrieved from the bank based on their relevance to the target user. This approach creates a personalized foundation that implements an infinite user-adaptive linear combination mechanism, enabling a small, fixed set of foundational modules to serve a virtually unlimited number of users.

The theoretical foundation for this approach rests on recent research demonstrating that skills and traits learned by different LoRA modules can be effectively combined through linear operations (Huang et al., 2024; Zhao et al., 2024; Prabhakar et al., 2024). These studies show that linear combinations facilitate generalization to new tasks while enabling the composition of novel abilities, providing the principled basis for our merging strategy.

Our implementation follows a two-step process. First, we identify the most relevant anchor users through embedding-based similarity matching. We concatenate each user's complete text history H_u into a unified document and also encode it using the dense encoder to produce a comprehensive representation $\boldsymbol{E}_u^{\text{retrieval}}$. The relevance between target user u and anchor user s_i is quantified through cosine similarity:

$$\text{sim}(u, s_i) = \frac{\boldsymbol{E}_u^{\text{retrieval}} \cdot \boldsymbol{E}_{s_i}^{\text{retrieval}}}{\|\boldsymbol{E}_u^{\text{retrieval}}\| \|\boldsymbol{E}_{s_i}^{\text{retrieval}}\|} \tag{4}$$

Based on these scores, we select the top-K most similar anchors, denoted as $\mathcal{S}_u = \{s_1, \ldots, s_K\}$, and perform a weighted linear combination of their LoRA parameters to form the merged LoRA $\mathbf{\Theta}_u^{\text{merged}}$:

$$\mathbf{\Theta}_u^{\text{merged}} = \sum_{i=1}^{K} \alpha_u^i \cdot \mathbf{\Theta}_{s_i} \tag{5}$$

where α_u^i represents the normalized weighting coefficient for the i-th anchor, ensuring that the contribution of each anchor is directly proportional to its relevance to the target user:

$$\alpha_u^i = \frac{\text{sim}(u, s_i)}{\sum_{j=1}^{K} \text{sim}(u, s_j)} \tag{6}$$

This weighting scheme guarantees that the most similar anchors contribute more significantly to the final merged LoRA, creating a knowledge-rich and highly relevant starting point for the final personalization stage. In our main experiments, we set $K = 2$ as the default configuration, as it strikes an optimal balance between personalization performance and computational efficiency (further analysis on K is provided in §4.5).

3.4 LoRA Stacking for Few-Shot Personalization

Following the merging step, although the merged module achieves personalization gains, it still lacks fine-grained training on target users. Moreover, sparse data distribution across different users often prevents reliable alignment due to insufficient data for parameter convergence. This necessitates achieving fine-grained training even in data-sparse scenarios. To address this, we introduce **LoRA Stacking**, a sequential Merge-then-Adapt strategy designed for data efficiency. Instead of continuing to train the larger merged LoRA, we freeze it and train *only* a new, ultra low-rank adaptation adapter on the user's few-shot data. Given its minimal parameter size, this approach enables efficient and robust adaptation even when training samples are extremely limited, ensuring a strong final alignment.

The first step involves integrating the merged LoRA $\mathbf{\Theta}_u^{\text{merged}}$ into the base model weights. This operation creates a new, intermediate model with updated weights $\boldsymbol{W}_{\text{merged}} = \boldsymbol{W}_0 + \boldsymbol{W}_{\text{LoRA}}^{\text{merged}}$, where the collaborative knowledge from anchor users becomes embedded within the model architecture. The resulting merged model is then frozen to serve as an enhanced foundation for subsequent personalization.

The second step introduces a user-specific adaptation layer designed for efficient fine-tuning on

limited data. We deploy an ultra low-rank LoRA module, termed the adaptation adapter $\mathbf{\Theta}_u^{\text{adapt}}$, which operates on top of the frozen merged foundation. The deliberately minimal parameter count of this adapter enables rapid convergence while maintaining personalization effectiveness. The forward pass for this stacked architecture follows:

$$\begin{aligned} \boldsymbol{h} &= \boldsymbol{W}_{\text{merged}}\boldsymbol{x} + \boldsymbol{W}_u^{\text{adapt}}\boldsymbol{x} \\ &= (\boldsymbol{W}_0 + \boldsymbol{W}_{\text{LoRA}}^{\text{merged}})\boldsymbol{x} + \boldsymbol{B}_u^{\text{adapt}}\boldsymbol{A}_u^{\text{adapt}}\boldsymbol{x} \end{aligned} \tag{7}$$

During training, the optimization process focuses exclusively on the parameters of Θ_u^{adapt} while minimizing loss on the user's limited historical data H_u. The adaptation adapter employs an ultra-low-rank configuration (e.g., $r = 4$), which substantially reduces trainable parameters and ensures computationally efficient convergence. This design enables effective personalization even when user data is severely limited, making the approach particularly suitable for real-world deployment scenarios where extensive user histories are unavailable.

4 Experiment

4.1 Experimental Setting

Datasets. Following previous work (Tan et al., 2024b,a; Zhang et al., 2024a), we conduct experiments on the Large Language Model Personalization (LaMP) benchmark (Salemi et al., 2023), which features seven public tasks spanning four classification and three generation scenarios(see Appendix §A for task details)[1]. A key focus of our work is to address the challenge of personalization in data-scarce scenarios. Therefore, following previous works (Tan et al., 2024b,a), for the test set of each task, we randomly select 100 users from the benchmark who possess a notably limited behavioral history. The remaining users are utilized for the construction of our Meta-LoRA Bank. This strict separation ensures that the users in the test set and those contributing to the bank are completely disjoint. Further details regarding the dataset statistics are available in the Appendix §B.

Baselines. We compare our proposed framework against two main categories of personalization baselines: prompt-based and fine-tuning-based methods. For the prompt-based baseline, we adopt the Retrieval-Augmented Personalization (RAG) method proposed in LaMP (Salemi et al., 2023), which augments the user's query with the top-k most relevant items retrieved from their personal history corpus. For the more advanced fine-tuning-based methods, we compare against several state-of-the-art frameworks, including OPPU (Tan et al., 2024b), PER-PCS (Tan et al., 2024a), and PROPER (Woźniak et al., 2024). Further details on the implementation of each baseline can be found in the Appendix §C.

Evaluation Metrics. Our evaluation protocol strictly adheres to the standards established by the LaMP benchmark (Salemi et al., 2023). For the text classification tasks (LaMP-1, LaMP-2), we assess performance using **accuracy** and **F1-score**. The personalized product rating task (LaMP-3), which can be treated as a regression problem, is evaluated using **Mean Absolute Error (MAE)** and **Root Mean Squared Error (RMSE)**. For all text generation tasks (LaMP-4, LaMP-5), we measure output quality using **ROUGE-1** and **ROUGE-L** (Lin, 2004).

Experimental Details. Detailed hyperparameter settings and the specific prompt templates used in our experiments are provided in Appendix §D and §E. To ensure a fair and powerful comparison, we use META-LLAMA-3-8B-INSTRUCT (Llama Team, 2024) as the foundational backbone LLM for all evaluated approaches, including our own. All experiments are conducted under three different random seeds and are conducted on a server equipped with a single NVIDIA L20 (48GB) GPU and 20 vCPUs of an Intel(R) Xeon(R) Platinum 8457C processor.

4.2 Overall Performance

Table 1 shows the performance of our proposed framework and baselines on the curated test sets across five tasks from the LaMP benchmark(Finally, we present a detailed case study in Appendix §G). We have the following observations:

MTA vs. RAG. As shown in Table 1, our proposed framework, MTA, shows a clear performance advantage over the Retrieval-Augmented Generation (RAG) baseline in the vast majority of tasks and metrics. For instance, MTA achieves relative gains of 6.67% in accuracy and 9.07% in F1-score for the citation identification task (LaMP-1) when compared against RAG's best performance

[1]We exclude the LaMP-6 task due to restricted access to its private dataset. Additionally, we exclude LaMP-7 task as its user history lacks the query-answer correspondence required to construct a training dataset for our framework.

Task	Metric	Prompt-based (RAG)			Fine-tuning-based			
		k=1	k=3	k=5	OPPU	PER-PCS	PROPER	**MTA(Ours)**
LaMP-1: PERSONALIZED CITATION IDENTIFICATION	Acc ↑	.4915	.5085	.4407	.5085	<u>.5339</u>	.5169	**.5424***
	F1 ↑	.4821	.4962	.4087	.5037	<u>.5289</u>	.5110	**.5412***
LaMP-2: PERSONALIZED MOVIE TAGGING	Acc ↑	.2917	.3472	.4028	.4167	.3611	<u>.4306</u>	**.4444***
	F1 ↑	.2054	.2510	<u>.3131</u>	.2368	.1911	**.3161***	.2592
LaMP-3: PERSONALIZED PRODUCT RATING	MAE ↓	.3964	.4144	.4144	.2432	.2162	<u>.2072</u>	**.1982***
	RMSE ↓	.7229	.8383	.8490	.5452	.5199	<u>.4746</u>	**.4350***
LaMP-4: PERSONALIZED NEWS HEADLINE GEN.	R-1 ↑	.1664	.1928	.1890	.2144	.1891	<u>.2152</u>	**.2399***
	R-L ↑	.1332	.1611	.1621	<u>.1885</u>	.1821	.1849	**.2134***
LaMP-5: PERSONALIZED SCHOLARLY TITLE GEN.	R-1 ↑	.4663	.4926	.4887	.4836	.4958	<u>.5122</u>	**.5291***
	R-L ↑	.3839	.4184	.4141	.4254	.4370	<u>.4487</u>	**.4659***

Table 1: Overall performance on the LaMP benchmark. For the RAG method, k denotes the number of retrieved user history items. The metrics R-1/-L refer to ROUGE-1/-L. The ↑ indicates the higher the better, while the ↓ indicates that lower values are preferable. The best results are highlighted in **bold** and the second-best is <u>underlined</u>. $(*)$ indicates statistically significant improvements (i.e., a two-sided t-test with $p < 0.05$) over the best baseline.

across different k-values. This trend is even more pronounced in the product rating prediction task (LaMP-3), where our method yields substantial improvements of 50.00% in MAE and 39.83% in RMSE. For news headline generation (LaMP-4), our framework also shows relative improvements of 24.43% in ROUGE-1 and 31.65% in ROUGE-L. These results highlight the benefits of our hybrid merging and stacking approach over relying solely on retrieval for LLM personalization.

MTA vs. OPPU. Our framework's "merge-then-adapt" strategy shows a distinct advantage over the OPPU baseline, which trains a LoRA module from scratch and is less effective in data-sparse settings. The empirical results in Table 1 confirm this. For instance, in the movie tagging task (LaMP-2), our method achieves relative gains of 6.65% in accuracy and 9.46% in F1-score. This performance improvement is also evident in the news headline generation task (LaMP-4), where our framework secures relative improvements of 11.89% in ROUGE-1 and 13.21% in ROUGE-L. These results underscore the superiority of our approach, which leverages collaborative knowledge to create a more robust and data-efficient starting point for personalization.

MTA vs. PER-PCS. While both our framework and **PER-PCS** leverage collaborative knowledge, our "merge-then-adapt" strategy demonstrates superior performance by incorporating a final, adaptive training stage. PER-PCS assembles a personalized model from pre-existing LoRA "pieces" without retraining, which can limit its ability to capture highly unique user profiles. The results in Table 1 validate this advantage. For example, in the movie tagging task (LaMP-2), MTA achieves relative improvements of 23.07% in accuracy and 35.64% in F1-score over PER-PCS. This performance gap is also substantial in the product rating task (LaMP-3), where our method obtains relative gains of 8.33% in MAE and 16.33% in RMSE. These results show that while assembling from shared components is effective, the final low-rank training step in our MTA is crucial to learn the fine-grained, residual preferences of individual users.

MTA vs. PROPER. Our framework also compares favorably to PROPER, a more complex, progressive training baseline. While PROPER utilizes a sophisticated Mixture-of-Experts and dual-router system, our simpler "merge-then-adapt" approach achieves competitive or superior results with significantly less architectural complexity, as demonstrated in Table 1. For instance, in the scholarly title generation task (LaMP-5), MTA surpasses PROPER with relative gains of 3.30% in ROUGE-1 and 3.83% in ROUGE-L. A similar advantage is observed in the product rating task (LaMP-3), where our method achieves relative improvements of 4.34% in MAE and 8.34% in RMSE. These findings suggest that our direct fusion and targeted residual adaptation strategy offers a more efficient path to strong personalization, avoiding the overhead associated with training complex routing and expert mechanisms.

4.3 Ablation Experiment

To validate the effectiveness and synergy of each key component within our proposed framework, we conduct a detailed ablation study with the results presented in Table 2. This study compares our

Task	Metric	Adapt-Only LoRA	Merged-Only LoRA	MTA (Ours)
LaMP-1	Acc ↑	.5254	.5339	**.5424**
	F1 ↑	.5182	.5323	**.5412**
LaMP-2	Acc ↑	.2917	.3611	**.4444**
	F1 ↑	.2054	.2031	**.2592**
LaMP-3	MAE ↓	.3514	.2342	**.1982**
	RMSE ↓	.8329	.4840	**.4350**
LaMP-4	R-1 ↑	.1757	.2374	**.2399**
	R-L ↑	.1467	.2127	**.2134**
LaMP-5	R-1 ↑	.4885	.5043	**.5291**
	R-L ↑	.4058	.4525	**.4659**

Table 2: Ablation study comparing our complete framework against its core components. The best performance for each metric is in **bold**, and the second-best is underlined.

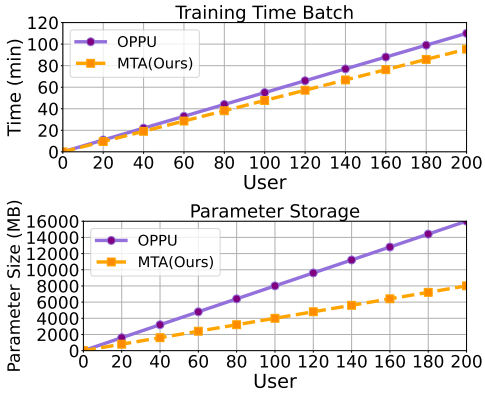

Figure 2: Efficiency comparison between our MTA framework and the OPPU baseline. The top plot shows total training time versus the number of users; the bottom plot shows total parameter storage.

complete framework, MTA, against the following two ablated variants:

- **Adapt-Only LoRA**: This variant bypasses the merging stage entirely. It directly fine-tunes a new, extremely low-rank LoRA on the base LLM using only the user's few-shot data. This isolates the effect of the final "adapt" step without the benefit of our collaborative warm-start.
- **Merged-Only LoRA**: This variant performs only the adaptive LoRA merging, omitting the final adaptation step. It isolates the effect of the collaborative warm-start.

The results in Table 2 clearly demonstrate that our framework MTA consistently and significantly outperforms both ablated versions. The performance of "Merged-Only LoRA" is substantially better than that of "Adapt-Only LoRA", which confirms that starting from a collaboratively-informed, merged base is far more effective than fine-tuning from a generic one. Furthermore, the significant performance jump from "Merged-Only LoRA" to our full MTA framework across all tasks highlights that the final, low-rank adaptation step is crucial for capturing the fine-grained, specific preferences of the user. In conclusion, both the initial merge and the final adaptation are integral and synergistic components of our framework's success.

4.4 Efficiency Experiment

We evaluate the efficiency of our framework in terms of average total training time and parameter storage across five benchmark tasks, comparing MTA against the OPPU baseline. As shown in Figure 2, MTA demonstrates significant advantages in both metrics. The efficiency stems from only needing to store a single, ultra-low-rank adapter per user and leveraging a strong merged foundation for faster convergence. While a training-free method like PER-PCS provides superior inference speed, it lacks a crucial adaptation stage for the target user, leading to degraded performance as validated in our experiments. Conversely, the hierarchical PROPER method incurs substantial costs: its initial stages demand exceptionally high parameter storage, and its multi-stage training paradigm incurs significant computational costs. These comparisons confirm that MTA offers a more balanced and scalable solution for large-scale deployment.

4.5 Parameter Experiment

In this section, we explore four key parameters that affect our results: the Merging Coefficient, which determines the proportional influence of each selected anchor LoRA; the Number of Merged Anchors, which sets how many modules are combined; the Adaptation LoRA Rank, which controls the parameter budget for the final personalization stage; and the Meta-LoRA Bank Size ($\mathcal{V}$), which defines the total number of candidate experts available for retrieval (the latter two are detailed in Appendix §F).

Analysis of the Merging Coefficient α_u. In our framework, the merging coefficient α_u is adaptively determined based on user similarity. To validate the effectiveness of this adaptive approach, we conduct an ablation study comparing it against several fixed merging coefficients. To demonstrate this, we select one representative task from both classification and generation domains: LaMP-3

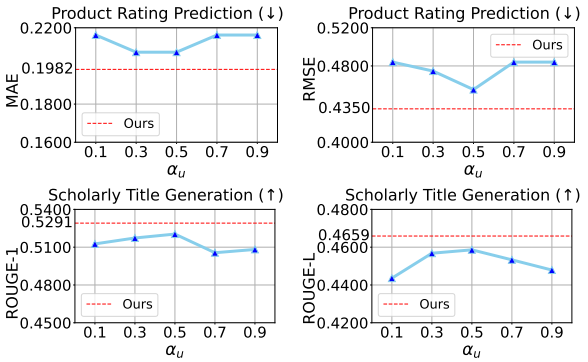

Figure 3: Performance comparison on Personalized Product Rating Prediction (LaMP-3) (top row, MAE/RMSE, lower is better) and Personalized Scholarly Title Generation (LaMP-5) (bottom row, ROUGE-1/L, higher is better) tasks with different fixed merging coefficients (α_u) versus our adaptive method (red dashed line). The blue solid line shows the performance of fixed-alpha variants.

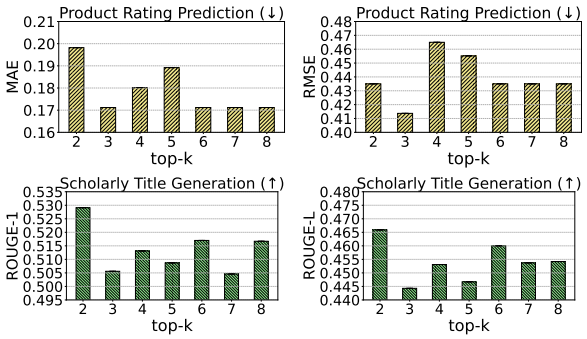

Figure 4: Performance on Personalized Product Rating Prediction (LaMP-3) (top row) and Personalized Scholarly Title Generation (LaMP-5) (bottom row) as a function of the number of merged top-K anchor users.

(Personalized Product Rating) and LaMP-5 (Personalized Scholarly Title Generation), respectively. We test several fixed values for the merging coefficient, which are sampled at equal intervals between 0 and 1.

As illustrated in Figure 3, our adaptive method consistently outperforms all fixed-alpha settings across every tested metric. This demonstrates the clear superiority of our similarity-based adaptive merging approach.

Impact of the Number of Merged Anchors (Top-K). To examine the impact of the number of merged anchors on performance, we conduct an analysis where we vary this number K from 2 to 8. For each K value, the top-K most similar anchor users are retrieved, and their LoRA modules are merged based on the normalized similarity weights. We evaluate this on two representative tasks: LaMP-3 (Product Rating) and LaMP-5 (Scholarly Title Generation). The results are illustrated in Figure 4.

For the Product Rating task (LaMP-3), the best performance is achieved at K=3. While a larger K can be beneficial for certain tasks, it also adds extra computing cost as the cost of merging scales linearly with the number of anchors. Conversely, for the Scholarly Title Generation task (LaMP-5), performance is optimal at K=2, and including more anchors beyond this point leads to performance degradation, likely due to noise from less-relevant peers.

5 Related Work

LLMs excel at general knowledge tasks but struggle with personalization tasks. Current personalized LLM research follows two main categories. (1). Prompt-based methods (Mysore et al., 2023; Richardson et al., 2023; Li et al., 2024) directly incorporate personalized information into prompts, offering flexibility and cost-effectiveness but suffering from noise susceptibility and privacy risks. (2). Fine-tuning-based methods typically employ PEFT for fine-tuning. Some works train a shared LoRA for cross-user generalization (e.g., PLoRA (Zhang et al., 2024b), LM-P (Woźniak et al., 2024)), while others follow a one-LoRA-per-user paradigm for enhanced personalization, such as OPPU (Tan et al., 2024b), PerPCS (Tan et al., 2024a), and PROPER (Zhang et al., 2025). Our MTA advances beyond static one-LoRA-per-user designs by constructing a meta-LoRA bank with dynamic merging, supplemented by LoRA stacking for finer-grained alignment, achieving accurate personalized adaptation even with limited samples.

6 Conclusion

In this paper, we propose MTA, a three-stage MTA framework for scalable and adaptable PLLMs. MTA first constructs a Meta-LoRA Bank from diverse anchor users. It then employs an adaptive merging technique to create a customized foundation for target users, followed by an ultra-low-rank LoRA stacking adaptation step to capture fine-grained preferences. Experiments on LaMP show consistent gains over prompt-based and strong fine-tuning baselines. MTA significantly reduced training and parameter storage costs, enabling efficient personalization at scale.

Limitations

Despite the effectiveness of our MTA framework, several limitations remain. First, our method relies on retrieving relevant experts; for "distinct users" whose styles significantly deviate from the pre-trained bank, the retrieval mechanism may fail to find suitable anchors, potentially limiting performance gains. Second, relying on a fixed Meta-LoRA Bank poses maintenance challenges in dynamic environments, where evolving user interests may render the expert pool outdated without regular updates.

References

Jingtong Gao, Bo Chen, Menghui Zhu, Xiangyu Zhao, Xiaopeng Li, Yuhao Wang, Yichao Wang, Huifeng Guo, and Ruiming Tang. 2024. Hierrec: Scenario-aware hierarchical modeling for multi-scenario recommendations. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 653–662.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *Preprint*, arXiv:2111.09543.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. 2024. Lorahub: Efficient cross-task generalization via dynamic lora composition. *Preprint*, arXiv:2307.13269.

Kevin B Johnson, Wei-Qi Wei, Dilhan Weeraratne, Mark E Frisse, Karl Misulis, Kyu Rhee, Juan Zhao, and Jane L Snowdon. 2021. Precision medicine, ai, and the future of personalized health care. *Clinical and translational science*, 14(1):86–93.

Changhao Li, Yuchen Zhuang, Rushi Qiang, Haotian Sun, Hanjun Dai, Chao Zhang, and Bo Dai. 2024. Matryoshka: Learning to drive black-box llms with llms. *arXiv preprint arXiv:2410.20749*.

Xiaopeng Li, Pengyue Jia, Derong Xu, Yi Wen, Yingyi Zhang, Wenlin Zhang, Wanyu Wang, Yichao Wang, Zhaocheng Du, Xiangyang Li, and 1 others. 2025. A survey of personalization: From rag to agent. *arXiv preprint arXiv:2504.10147*.

Xiaopeng Li, Fan Yan, Xiangyu Zhao, Yichao Wang, Bo Chen, Huifeng Guo, and Ruiming Tang. 2023. Hamur: Hyper adapter for multi-domain recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 1268–1277.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. 2024. Once: Boosting content-based recommendation with both open-and closed-source large language models. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 452–461.

Llama Team. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Zhengyi Ma, Zhicheng Dou, Yutao Zhu, Hanxun Zhong, and Ji-Rong Wen. 2021. One chatbot per person: Creating personalized chatbots based on implicit user profiles. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 555–564.

Sheshera Mysore, Zhuoran Lu, Mengting Wan, Longqi Yang, Steve Menezes, Tina Baghaee, Emmanuel Barajas Gonzalez, Jennifer Neville, and Tara Safavi. 2023. Pearl: Personalizing large language model writing assistants with generation-calibrated retrievers. *arXiv preprint arXiv:2311.09180*.

Akshara Prabhakar, Yuanzhi Li, Karthik Narasimhan, Sham Kakade, Eran Malach, and Samy Jelassi. 2024. Lora soups: Merging loras for practical skill composition tasks. *Preprint*, arXiv:2410.13025.

Muh Putra Pratama, Rigel Sampelolo, and Hans Lura. 2023. Revolutionizing education: harnessing the power of artificial intelligence for personalized learning. *Klasikal: Journal of education, language teaching and science*, 5(2):350–357.

Chris Richardson, Yao Zhang, Kellen Gillespie, Sudipta Kar, Arshdeep Singh, Zeynab Raeesy, Omar Zia Khan, and Abhinav Sethy. 2023. Integrating summarization and retrieval for enhanced personalization via large language models. *arXiv preprint arXiv:2310.20081*.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2023. LaMP: When large language models meet personalization. *Preprint*, arXiv:2304.11406.

Zhaoxuan Tan, Zheyuan Liu, and Meng Jiang. 2024a. Personalized pieces: Efficient personalized large language models through collaborative efforts. *arXiv preprint arXiv:2406.10471*.

Zhaoxuan Tan, Qingkai Zeng, Yijun Tian, Zheyuan Liu, Bing Yin, and Meng Jiang. 2024b. Democratizing large language models via personalized parameter-efficient fine-tuning. *arXiv preprint arXiv:2402.04401*.

Stanisław Woźniak, Bartłomiej Koptyra, Arkadiusz Janz, Przemysław Kazienko, and Jan Kocoń. 2024. Personalized large language models. In *2024 IEEE*

International Conference on Data Mining Workshops (ICDMW), pages 511–520. IEEE.

Linhai Zhang, Jialong Wu, Deyu Zhou, and Yulan He. 2025. Proper: A progressive learning framework for personalized large language models with group-level adaptation. *arXiv preprint arXiv:2503.01303*.

Linhai Zhang, Jialong Wu, Deyu Zhou, and Guoqiang Xu. 2024a. STAR: Constraint LoRA with dynamic active learning for data-efficient fine-tuning of large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3519–3532, Bangkok, Thailand. Association for Computational Linguistics.

You Zhang, Jin Wang, Liang-Chih Yu, Dan Xu, and Xuejie Zhang. 2024b. Personalized lora for human-centered text understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19588–19596.

Xiangyu Zhao, Yichao Wang, Bo Chen, Jingtong Gao, Yuhao Wang, Xiaopeng Li, Pengyue Jia, Qidong Liu, Huifeng Guo, and Ruiming Tang. 2025. Joint modeling in recommendations: A survey. *arXiv preprint arXiv:2502.21195*.

Ziyu Zhao, Tao Shen, Didi Zhu, Zexi Li, Jing Su, Xuwu Wang, Kun Kuang, and Fei Wu. 2024. Merging loras like playing lego: Pushing the modularity of lora to extremes through rank-wise clustering. *Preprint*, arXiv:2409.16167.

Appendix

A Task Details

Here we present a detailed breakdown of the seven personalization tasks. The objective of these descriptions is to provide a clear understanding of the format and requirements for each task.

- **Personalized Citation Identification:** This is a binary text classification task. Given a paper with title x authored by user u, and two candidate papers for citation, the model must predict which of the two candidates user u is more likely to cite. The prediction is based on the user's history, which consists of the titles and abstracts of their previous publications.

- **Personalized Movie Tagging:** As a 15-way text classification task, the goal here is to predict a tag for a movie that aligns with a user's personal tagging preferences. The input to the model is a movie description x. The model must output a suitable tag, chosen from a set of fifteen, based on the user's historical movie-tag associations.

- **Personalized Product Rating:** This task can be framed as either a 5-way text classification or a regression problem. The model receives a product review x written by user u. Its objective is to predict the integer rating (from 1 to 5) that user u would assign to the product. This prediction is guided by the user's past review and rating pairs.

- **Personalized News Headline Generation:** This is a text generation task designed to assess a model's capacity to emulate an author's distinct style. The input query x is the body of a news article. The model is required to generate a headline y for this article that reflects the stylistic patterns observed in the user's historical article-headline pairs.

- **Personalized Scholarly Title Generation:** Similar to the news headline task, this is a text generation challenge set in an academic context. Given an abstract of a research paper x, the model's task is to generate a title y. The generated title should be stylistically consistent with the author's previous work, as evidenced by their historical abstract-title pairs.

We exclude the **LaMP-6: Email Subject Generation** task due to restricted access to its private dataset.

Furthermore, we choose to exclude **LaMP-7: Personalized Tweet Paraphrasing** because its data organization presents a fundamental conflict with our framework's design. In LaMP-7, the provided user history is a holistic corpus of a user's past tweets, rather than a collection of structured query-answer pairs. Specifically, the history contains an author's original tweets, while the task is to paraphrase a new, normalized sentence in that author's style. This format prevents us from constructing a training dataset directly from the user's history.

Benchmark methods such as **OPPU** and **PROPER** adopt a multi-stage process to handle such scenarios. Their process begins with a **population-level adaptation** stage. In this initial step, the model is fine-tuned on a general task dataset using query data from a pool of non-target users, with the goal of learning broad, task-related capabilities before any personalization occurs. To simplify the overall process and improve efficiency, our framework is designed to circumvent this large-scale, preparatory pre-training. Therefore, we omit the LaMP-7 task from our experiments.

B Datasets Details

Table 3 presents the statistics for the number of user history items across the different LaMP tasks (Salemi et al., 2023). As stated in the main text, our work focuses on evaluating personalization in data-scarce scenarios. Consistent with this focus, we curate our test sets by selecting users who possess a limited number of historical interactions. Specifically, for each task, we randomly select 100 users for the test set based on the length of their history. For the LaMP-2 and LaMP-4 tasks, we choose users with 10 history items each. For LaMP-1 and LaMP-5, users with 50 history items are selected. Finally, for the LaMP-3 task, we select users with 100 history items.

Table 3: Statistics on the number of history items per user for each LaMP task.

Task	Max Items	Min Items	Avg. Items
LaMP-1	607	40	88.4
LaMP-2	1289	4	204.46
LaMP-3	1021	97	202.52
LaMP-4	945	7	31.2
LaMP-5	911	47	94.34

C Baseline Details

We compare our proposed framework against two main categories of personalization baselines: prompt-based and fine-tuning-based methods. Further details on each are provided below.

- **RAG**: This is a prompt-based personalization method introduced in the LaMP benchmark (Salemi et al., 2023). For a given user query, this approach retrieves the top-k most relevant items from that user's historical data corpus. These retrieved items are then concatenated with the original query to form an augmented prompt, which is subsequently fed to the LLM to generate a personalized response. In our experiments, we evaluate RAG with the number of retrieved items k set to 1, 3, and 5, and compare our framework against its strongest performance across these settings.

- **OPPU** (Tan et al., 2024b): This fine-tuning-based framework, proposed by , pioneers the "one-PEFT-per-user" paradigm. It addresses challenges of model ownership and generalization by training a dedicated, lightweight PEFT module (specifically LoRA) for each individual user from scratch, using that user's entire behavior history. The resulting personal PEFT module encapsulates the user's specific preferences and can be "plugged into" a base LLM to create a personalized model. This approach requires separate training and storage for each user, leading to computation and storage costs that scale linearly with the user base.

- **PER-PCS** (Tan et al., 2024a): This framework introduces a more efficient, collaborative approach to PEFT-based personalization. Instead of training a new PEFT module for every target user, PER-PCS first trains PEFTs for a select group of representative "sharer" users. These PEFTs are then broken down into smaller modules, or "pieces". For a new target user, PER-PCS assembles a personalized PEFT in a training-free manner by selecting and combining the most relevant pieces from the shared pool, guided by the target user's own history data. This method achieves performance comparable to OPPU but with significantly lower computation and storage requirements.

- **PROPER** (Zhang et al., 2025): This framework addresses the data scarcity issue in personalization through a progressive, three-stage learning process. It introduces a meso-level, group-based adaptation between the population and user levels. The process is as follows: (1) **Population-Level Adaptation**, where a base LoRA is trained on a general user pool for task alignment; (2) **Group-Level Adaptation**, which uses a Mixture-of-Experts (MoE) structure to learn shared preferences for automatically clustered user groups; and (3) **User-Level Adaptation**, where a final, user-specific LoRA is trained to capture the remaining individual preferences. This hierarchical approach aims to provide a better starting point for personalizing data-sparse users.

D Hyperparameter Details

The specific hyperparameter settings used for the final adaptation stage of our framework are detailed in Table 4. We note that the LoRA rank, per device train batch size, and gradient accumulation steps are kept consistent across all tasks, while the number of training epochs and the learning rate are varied to optimize performance for each specific task.

Task	Rank	Epochs	LR	BS	Accum.
LaMP-1	4	2	5×10^{-5}	2	4
LaMP-2	4	3	1×10^{-4}	2	4
LaMP-3	4	1	5×10^{-5}	2	4
LaMP-4	4	3	1×10^{-4}	2	4
LaMP-5	4	2	5×10^{-5}	2	4

Table 4: Hyperparameter settings for our framework across the evaluated LaMP tasks.

E Prompt Details

We present the prompt templates used for each task in our experiments. The text in curly braces, such as `{USER HISTORY}`, is replaced with content specific to different users and queries.

LaMP-1: Personalized Citation Identification

User History:
`{USER HISTORY}`

User Instruction:
For an author who has written the paper with the title "`{QUERY PAPER TITLE}`", which reference is related? Just answer with [1] or [2] without explanation.
Reference: [1] - `{OPTION 1}` [2] - `{OPTION 2}`
Answer:

LaMP-2: Personalized Movie Tagging

User History:
`{USER HISTORY}`

User Instruction:
Which tag does this movie relate to among the following tags? Just answer with the tag name without further explanation.
tags: [sci-fi, based on a book, comedy, action, twist ending, dystopia, dark comedy, classic, psychology, fantasy, romance, thought-provoking, social commentary, violence, true story]
description: `{QUERY MOVIE DESCRIPTION}`
Tag:

LaMP-3: Personalized Product Rating

User History:
`{USER HISTORY}`

User Instruction:
What is the score of the following review on a scale of 1 to 5? just answer with 1, 2, 3, 4, or 5 without further explanation.
review: `{QUERY REVIEW}`
Score:

LaMP-4: Personalized News Headline Generation

User History:
`{USER HISTORY}`

User Instruction:
Generate a headline for the following article.
Article: `{QUERY ARTICLE}`
Headline:

LaMP-5: Personalized Scholarly Title Generation

User History:
`{USER HISTORY}`

User Instruction:
Generate a title for the following abstract of a paper.
Abstract: `{QUERY ABSTRACT}`
Title:

F Supplementary Experiment

Impact of Adaptation LoRA Rank. To analyze the impact of the rank (r) of the final adaptation LoRA on performance, we conduct a supplementary experiment. We evaluate the performance of our full framework on two representative tasks (LAMP-3 Product Rating and LAMP-5 Scholarly Title Generation) across a range of different ranks ($r \in \{2, 4, 6, 8, 16\}$).

The results are shown in Figure 5. We observe that the optimal rank is task-dependent. For the Product Rating task, the model achieves the best performance on both MAE and RMSE metrics at a rank of 6 or 8. In contrast, the optimal rank for the Scholarly Title Generation task is 4. Notably, the relationship between performance and rank is non-monotonic; for both tasks, performance degrades when the rank is increased to 16.

This finding confirms that using an extremely low rank is an efficient and effective strategy. Our analysis shows that selecting a small rank (such as $r = 4$) achieves optimal or near-optimal perfor-

mance while minimizing the parameter count. This validates our choice of $r = 4$ as the default setting in our main experiments, striking an excellent balance between performance and efficiency.

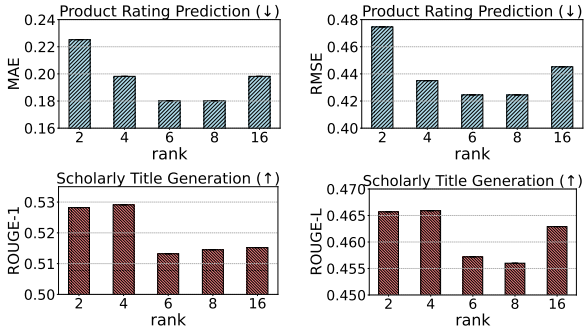

Figure 5: Performance analysis of the final adaptation LoRA with varying ranks (r). The top row shows Product Rating (LAMP-3) results; the bottom row shows Scholarly Title Generation (LAMP-5) results.

Impact of Meta-LoRA Bank Size ($\mathcal{V}$). To investigate the influence of the Meta-LoRA Bank size on model performance, we conduct an experiment on the LaMP-4 (Personalized News Headline Generation) task by varying $\mathcal{V}$ from 50 to 100. As illustrated in Figure 6, both ROUGE-1 and ROUGE-L scores exhibit a consistent upward trend as the bank size expands. We attribute this improvement to the enriched diversity of the expert pool: a larger bank covers a broader spectrum of latent styles, which increases the probability of retrieving highly relevant anchors for the target user. This results in a more effective merged initialization, ultimately boosting generation quality.

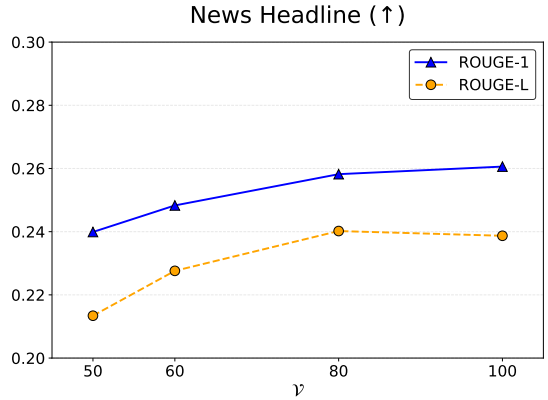

Figure 6: Performance analysis of Meta-LoRA Bank size ($\mathcal{V}$) on the LaMP-4 task. Increasing the number of candidate LoRAs in the bank consistently improves generation quality (ROUGE-1 and ROUGE-L).

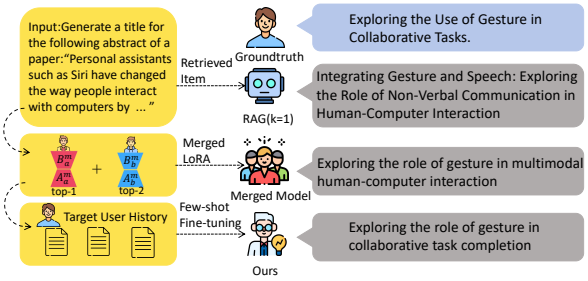

Figure 7: Case study on personalized title generation (LaMP-5) for a user specializing in Computer Vision. Our full framework generates the most accurate title by capturing the user's task-oriented focus.

G Case Study

Figure 7 presents a case study on the LaMP-5 task to demonstrate our framework's effectiveness. The target user ('user_id: 11001393') is an expert in Computer Vision, with a publication history focused on geometric matching, image segmentation, and object recognition. The query asks for a title for an abstract about using gestures in collaborative tasks for human-computer interaction. The RAG (k=1) baseline produces a verbose title focused on "Speech-Human-Computer Interaction", missing the key concept of "Collaborative Tasks". The Merged-Only Model, benefiting from the merged LoRAs of similar users, shows improvement by identifying the general "multimodal human-computer interaction" theme but still lacks specificity.

In contrast, our complete framework (Ours) generates a highly precise title, "Exploring the role of gesture in collaborative task completion", which is closely match the ground truth, demonstrating the critical importance of our final adaptation stage. Our framework successfully distills the user's specific, task-oriented research focus (evident from their history on "model matching" and "task completion") and then effectively applies it to the new domain. This targeted approach enables the system to correctly identify and emphasize the "collaborative task" component that other models overlook, resulting in superior performance.