🧩PERSONALIZED PIECES: Efficient Personalized Large Language Models through Collaborative Efforts

Anonymous ACL submission

Abstract

Personalized large language models (LLMs) aim to tailor interactions, content, and recommendations to individual user preferences. While parameter-efficient fine-tuning (PEFT) methods excel in performance and generalization, they are costly and limit communal benefits when used individually. To this end, we introduce PERSONALIZED PIECES (PER-PCS), a framework that allows users to safely share and assemble personalized PEFT efficiently with collaborative efforts. PER-PCS involves selecting sharers, breaking their PEFT into pieces, and training gates for each piece. These pieces are added to a pool, from which target users can select and assemble personalized PEFT using their history data. This approach preserves privacy and enables fine-grained user modeling without excessive storage and computation demands. Experimental results show PER-PCS outperforms non-personalized and PEFT retrieval baselines, offering performance comparable to OPPU with significantly lower resource use across six tasks. Further analysis highlights PER-PCS's robustness concerning sharer count and selection strategy, pieces sharing ratio, and scalability in computation time and storage space. PER-PCS's modularity promotes safe sharing, making LLM personalization more efficient, effective, and widely accessible through collaborative efforts.

1 Introduction

Personalization involves mining user's history data to tailor and customize a system's interaction, content, or recommendations to meet the specific needs, preferences, and characteristics, of individual users (Tan and Jiang, 2023; Chen et al., 2023; Kirk et al., 2024). By adapting to each user's unique preferences, personalization enhances the user experience and has become increasingly important in content recommendation (Li et al., 2023b; Wu et al., 2023; Baek et al., 2023), user

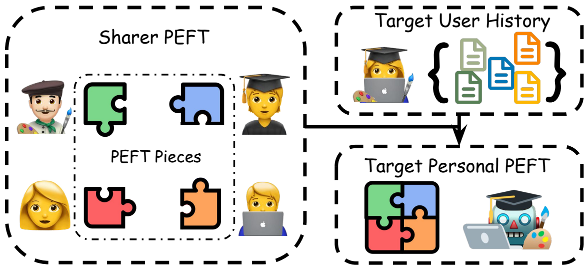

Figure 1: User personal PEFT parameter sharing framework: Sharers provide parts of their PEFT parameters (PEFT pieces). Using the target user's history data, we recycle the PEFT pieces shared by anchor users and assemble the target user's personal PEFT.

simulation (Dejescu et al., 2023; Zhang and Balog, 2020), personalized chatbot (Srivastava et al., 2020; Ma et al., 2021), user profiling (Gu et al., 2020; Gao et al., 2023), healthcare (Johnson et al., 2021; Goldenberg et al., 2021), and education (Alamri et al., 2021; Pratama et al., 2023).

Large language models (LLMs) are revolutionizing the research landscape with emergent abilities not observed in smaller models (Wei et al., 2022a; Lu et al., 2023), due to their training on massive textual corpora and billions of parameters. These abilities include step-by-step reasoning (Wei et al., 2022b), in-context learning (Min et al., 2022), and instruction following (Wei et al., 2021). Despite these capabilities, current LLMs adhere to a "one-size-fits-all" paradigm, being trained on broad, domain-agnostic data, which limits their effectiveness in adapting to individual user preferences (Chen et al., 2023). Consequently, personalizing LLMs to align with users' unique needs has become a crucial research focus (Li et al., 2023a).

Previous endeavors to personalize LLMs can be categorized into prompt-based and parameter-efficient fine-tuning (PEFT)-based methods. Prompt-based personalization involves designing prompt templates to help LLMs understand user preferences, using methods such as vanilla personalized prompting (Dai et al., 2023), retrieval-augmented prompting (Mysore et al., 2023), and profile-augmented prompting (Richardson et al., 2023). However, prompt-based methods expose user data to centralized LLM and can be easily distracted by irrelevant user history data, which retrieval can hardly avoid (Shi et al., 2023). *PEFT-based* personalization methods focus on storing users' preferences and behavior patterns in personal lightweight parameters. OPPU (Tan et al., 2024) is the pioneering work that stores users' preferences and behavior patterns in personal PEFT parameters, showing the superiority of model ownership and better user behavior pattern generalization compared to prompt-based methods. Despite their success, the "one-PEFT-per-user" paradigm is computationally and storage-intensive, especially for large user bases. For instance, using OPPU for personalized product rating prediction requires about 20 minutes of training on a single RTX A6000 GPU and 17 MB of storage per user, scaling linearly with the number of users. Additionally, individually owned PEFTs limit community value, as personal models cannot easily share knowledge or benefit from collaborative improvements.

Inspired by the exhaustiveness of human preferences (Lee et al., 2024), we propose the PERSONALIZED PIECES (PER-PCS) framework, which allows users to safely share a small fraction of their PEFT parameters and build personalized LLMs efficiently through collaborative efforts (Figure 1). Specifically, we first select representative users as sharers and train their PEFTs with their personal history data. We then break down the PEFT parameters into pieces, inject a routing gate for each piece, and update the gate parameters while keeping the other parameters frozen with a few steps. These pieces are added to a pieces pool along with their corresponding gates for selection. In the assembly stage, PER-PCS feeds the target user's history data and selects PEFT pieces from the pieces pool in an auto-regressive way, recycling the PEFT modules in the pieces pool. By processing all the history data through this pipeline, we determine the PEFT piece choices for all layers and obtain the target user's personal PEFT. PER-PCS is training-free and only requires the storage of sharer's index and corresponding composition weights, making it computation and storage efficient.

Experimental results show that PER-PCS outperforms non-personalized and PEFT retrieval baselines, delivering performance comparable to OPPU but with significantly reduced resource requirements across six personalization tasks in the LaMP benchmark (Salemi et al., 2023). Further studies highlight PER-PCS's robustness against sharer count and selection strategy. Even when sharers consent to share only a small portion of their pieces, PER-PCS maintains strong performance, comparable to scenarios where all pieces are shared. Time analysis reveals that PER-PCS is 38 times more efficient in storage and 7 times more efficient in computation costs compared to OPPU. These findings underscore the potential of personalizing general-purpose LLMs by integrating modular and collaborative parametric knowledge from personal PEFT pieces shared by users.

In summary, the contribution of PER-PCS is the pioneering framework that enables users to safely share personal PEFTs, facilitating efficient and fine-grained LLM personalization through collaborative efforts. Unlike OPPU, where personal PEFTs benefit only the individual user, PER-PCS allows users to share a limited portion of their PEFT parameters with others, ensuring user privacy. For target users, PER-PCS maintains model ownership and supports fine-grained user modeling comparable to OPPU, but with significantly reduced storage and computation resources. We envision PER-PCS as an initiative to encourage users to share their personal PEFT pieces, fostering collaboration in personalizing LLMs to create value for others. This approach preserves sharer privacy and reduces the carbon footprint of PEFT-based personalized LLM.

2 PERSONALIZED PIECES (PER-PCS)

We introduce PER-PCS, a novel framework to empower LLM personalization with modular and collaborative PEFT pieces within the community (Fig. 2). We first adapt non-personalized base LLMs to the task without incorporating personal preferences (§2.2). We then train personal PEFT and post-hoc gates for sharers and add them to the pool (§2.3). Finally, we assemble the target user's PEFT using their history and pieces from the pool (§2.5).

2.1 Preliminaries

Research Problem Formulation. For personalized LLM at time t, the model's output r_u for user u is conditioned on both query q_u and the user's behavior history $\mathcal{H}_u = \{h_u\}$ that includes all user behaviors occurred before query time t. Assuming users in set $\mathcal{U}$ have personal PEFT, while the target

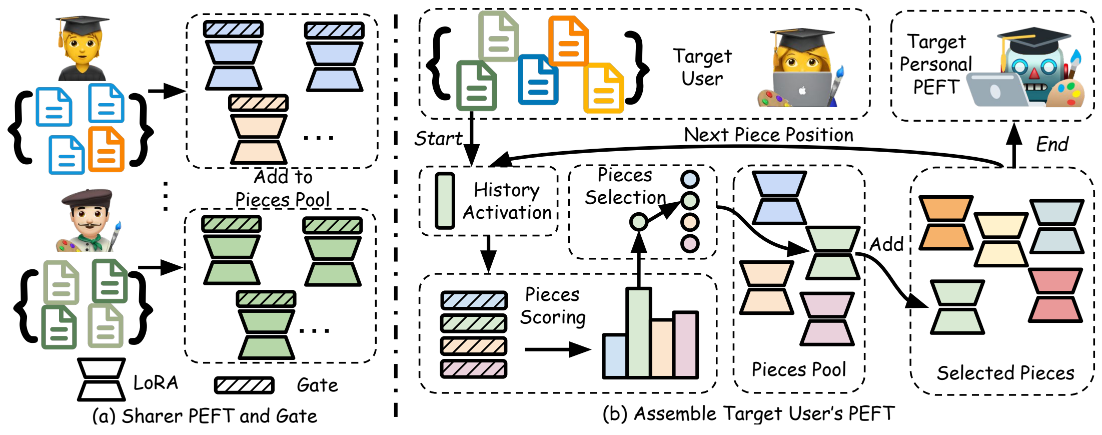

Figure 2: Overview of PER-PCS. First, we train PEFT and gate each piece for sharing. Next, we feed the target user's history, utilizing history activation and piece gates to score and select PEFT pieces from the pool. These selected pieces are then assembled to create a personalized PEFT for the target user.

user $\hat{u} \notin \mathcal{U}$ does not, our goal is to assemble the target user's PEFT $\Delta\Theta_{\hat{u}}$ from $\{\Delta\Theta_u, u \in \mathcal{U}\}$.

PEFT Pieces. We assume a PEFT method introduces modules throughout the whole model. For example, LoRA (Hu et al., 2021) introduces a low-rank update at every linear layer in the model. We refer to each of these updates as a "piece".

2.2 Base LLM Task Adaption

Since off-the-shelf LLMs do not inherently understand personalization tasks, we follow LaMP (Salemi et al., 2023) and Richardson et al. (2023) to fine-tune LLMs for fair comparison and task comprehension. In adapting the base LLM, we use data that excludes target users' and sharers' data to build an LLM that understands task-related capabilities rather than personal preferences. Specifically, the base LLM parameter Θ_o is optimized *w.r.t.* loss $L = \mathrm{CE}[\Theta_o(\phi(q_u, \mathcal{R}(q_u, \mathcal{H}_u, m))), r_u]$, where CE denotes the cross entropy loss function, $\mathcal{R}$ is the retriever, ϕ is the prompt construction function, m is the number of retrieval items, and $\mathcal{H}_u$ is the entire user behavior history. For computational efficiency, we adopted LoRA (Hu et al., 2021) for parameter-efficient fine-tuning and merged it into pretrained weights to obtain the base LLM.

2.3 Sharer Selection and PEFT Training

After adapting the base LLMs to the task without incorporating the target user's personal preferences, we select sharers who consent to share their PEFTs with the community and train their personal PEFTs. To select representative users, we first get embeddings for all candidate users by encoding their history with an encoder-only language model, `DeBERTa-v3-Large` (He et al., 2022). The user embedding $E_u = \sum_{h_u \in \mathcal{H}_u} \mathrm{Enc}(h_u)/|\mathcal{H}_u|$ by averaging all history items h_u from user u. We then cluster user embeddings with the k-means algorithm (K=50 by default) and select the most active users within the i-th cluster as sharer s_i $(i = 1, ..., K)$,[1]

$$s_i = \{\arg\max_{u \in \mathcal{C}_i} |\mathcal{H}_u|, \mathcal{C}_i \in \textit{k-means}(\mathcal{E}, K)\},$$

where $\mathcal{C}_i$ denotes the i-th user cluster, $\mathcal{E} = \{E_u, u \in \mathcal{U}\}$ denotes the embedding set of all sharer candidates. Following OPPU (Tan et al., 2024), we then train personal PEFT parameters Θ_{s_i} for sharer s_i using sharer's history data $\mathcal{H}_{s_i}$.

We then break the sharers' PEFT parameters Θ_{s_i} into pieces. For clarity, we consider the case where users perform personal PEFT using LoRA (Hu et al., 2021). It's worth noting that PER-PCS is compatible with all PEFT methods that introduce trainable modules throughout the model, such as Adapter (Houlsby et al., 2019), (IA)^3 (Liu et al., 2022), and prefix tuning (Li and Liang, 2021). We primarily focus on LoRA due to its popularity, widespread use, and superior performance demonstrated by OPPU. LoRA modifies the output of l-th linear layer from $z_t^l = W_o^l v_t^l$ using a low-rank decomposition to $z_t^l = W_o^l v_t^l + \Delta W^l v_t^l = W_o^l v_t^l + B^l A^l v_t^l$, where $v_t^l \in \mathbb{R}^n$ denotes the t-th input activation at layer l, $W_o^l \in \mathbb{R}^{d\times n}$ denotes the base model parameters which remain frozen during fine-tuning, $A^l \in \mathbb{R}^{r\times n}$ and $B^l \in \mathbb{R}^{d\times r}$ are trainable parameters. Therefore, a pair of (B^l, A^l) is defined as a "piece" and personal PEFT parameters Θ_{s_i} for sharer s_i can break down to pieces

[1]Please see more sharer selection strategies in Section 5.

set $\{(B^l_{s_i}, A^l_{s_i})\}^L_{l=1}$, L denotes the total number of layers in a LoRA module.

2.4 Post-Hoc Sharer Gating Training

We then add a piece selection gate for each sharer PEFT piece to determine which piece should be selected in the upcoming assembly step. For each sharer PEFT piece, after integrating the gate, a linear layer becomes:

$$z^l_t = W^l_o v^l_t + B^l_{s_i} A^l_{s_i} v^l_t \sigma(g^{l\top}_{s_i} v^l_t),$$

where $g^l_{s_i} \in \mathbb{R}^n$ is a trainable gate vector for sharer s_i at layer l and initialized to all zeros, σ is the sigmoid activation function, W^l_o, $B^l_{s_i}$, and $A^l_{s_i}$ are frozen. For sharer s_i, we optimize $\{g^l_{s_i}\}^L_{l=1}$ using sharer history $\mathcal{H}_{s_i}$. We then add the sharers' PEFT pieces and corresponding gates to the pieces pool for the upcoming selection and assembly. Gate training is limited to around 50 steps, making it computationally efficient. The post-hoc nature of gate learning also adds flexibility, facilitating easier deployment in real-world scenarios.

2.5 Assemble Target Personal PEFT

Motivated by the exhaustiveness of human preferences (Lee et al., 2024), we assemble PEFT modules for target users using the PEFT modules and gate vectors from sharers. Using the target user's history $\mathcal{H}_{\hat{u}}$ as input, we perform auto-regressive PEFT piece selection to assemble the target user's PEFT from input to output. For each layer l in LoRA, we feed the user history in LLM and compute the score for the input activation v^l_t and candidate pieces using cosine similarity, then aggregate these scores from the token level to obtain the piece-level score $\alpha^l_{s_i}$ for the piece from sharer s_i:

$$\alpha^l_{s_i} = \sum^{e}_{t=b} (\overline{g}^{l\top}_{s_i} \overline{v}^l_t),$$

where $\overline{g}^l_{s_i}$ and $\overline{v}^l_t$ are the normalized gate vector and activation. For user history $(x_{\hat{u}}, y_{\hat{u}}) \in \mathcal{H}_{\hat{u}}$ aligned with the task format, we set begin position $b = |x_{\hat{u}}| + 1$ and end position $e = |x_{\hat{u}}| + |y_{\hat{u}}| + 1$, where $|\cdot|$ denotes sequence length. Otherwise, for history $x_{\hat{u}} \in \mathcal{H}_{\hat{u}}$, we set $b = 1$ and $e = |x_{\hat{u}}| + 1$.

We then select the top-k PEFT pieces at l-th layer to select sharer set $\mathcal{S}^l$ for target user PEFT assemble $\mathcal{S}^l = \{s_i, \text{keep top-}k \text{ ranked by } \alpha^l_{s_i}\}$. Next, we normalize the selected weights with the $1/\sqrt{n}$ scaling factor to avoid saturation (Vaswani et al., 2017), which can be expressed as

$$w^l_s = \mathit{softmax}(\{\alpha^l_s/\sqrt{n},\ s \in \mathcal{S}^l\}),$$

where n denotes the embedding dimension, s denotes the index of selected pieces. We then aggregate the selected PEFT pieces with weight to assemble the target user's PEFT $\Delta W^l_{\hat{u}}$ at layer l:

$$\Delta W^l_{\hat{u}} = \sum_{s \in \mathcal{S}^l} (w^l_s B^l_s A^l_s)$$

where A^l_s and B^l_s are PEFT piece parameters from s-th sharer. Using the assembled personal PEFT parameters for target user $\hat{u}$, the feed forward function for a linear layer becomes

$$z^l_t = W^l_o v^l_t + \Delta W^l_{\hat{u}} v^l_t.$$

After detailing the assembly process for a single piece in the target user's PEFT, we extend it to the entire model that contains L layers. Once the piece at l-th layer parameter assembly is complete, the output z^l_t is used as the input activation for the l+1 layer selection. After computing parameter selection for all history items and layers, we average the composed parameters to obtain the final PEFT parameters for the target user $\Delta\Theta_{\hat{u}} = \{\Delta W^l_{\hat{u}}\}^L_{l=1}$, which is a set of assembled parameters across all layers sourced from sharers' piece parameters.

Overall, the assembly process does not involve model training or optimization, making it computationally efficient compared to training personal PEFT for each target user from scratch. For storage, instead of storing the entire set of matrices in LoRA for each target user, PER-PCS only needs to store the selected PEFT piece index $\mathcal{S}^l$ and corresponding weights w^l_s across all layer positions, ensuring PER-PCS storage efficient.

3 Experiment Settings

Datasets We adopt the Large Language Model Personalization (LaMP) benchmark (Salemi et al., 2023) for our experiments, which consists of six public language model personalization tasks, including three text classification tasks (personalized citation identification, movie tagging, and producing rating) and three text generation tasks (personalized news headline generation, scholarly title generation, and tweet paraphrasing).[2] We randomly

[2]Task details can be found in Appendix F. We exclude the LaMP-6: Email subject generation task since it involves private data that we cannot access.

Table 1: Main experiment results on the LaMP benchmark. R-1 and R-L denote ROUGE-1 and ROUGE-L. k refers to the number of retrieved items, with k=0 indicating no retrieval; k=1 is the default. ↑ means higher values are better, and ↓ means lower values are better. The best score for each task is in **bold**, and the second best is underlined. '*' indicates significant improvement against counterparts without PER-PCS.

Task	Metric	Non-Personalized		RAG	PAG	PEFT Retrieval			PER-PCS (Ours)			OPPU		
		k=0	Random	k=1	k=1	Base	+RAG	+PAG	Base	+RAG	+PAG	Base	+RAG	+PAG
LAMP-1: PERSONALIZED	Acc ↑	.536	.576	.584	.656	.480	.576	.656	.592*	.579	**.672***	.560	.584	.664
CITATION IDENTIFICATION	F1 ↑	.532	.568	.567	.654	.361	.556	.653	.589*	.564	**.664***	.553	.567	.658
LAMP-2: PERSONALIZED	Acc ↑	.340	.301	.417	.469	.336	.419	.477	.410*	.452*	.499*	.463	.467	**.507**
MOVIE TAGGING	F1 ↑	.268	.255	.331	.375	.265	.326	.380	.301*	.343*	.383*	.320	.349	**.385**
LAMP-3: PERSONALIZED	MAE ↓	.645	.336	.301	.301	.431	.305	.299	**.262***	.272*	**.262***	**.262**	.272	.266
PRODUCT RATING	RMSE ↓	1.277	.662	.639	.618	.897	.622	.608	**.558***	.580*	.561*	**.558**	.580	.561
LAMP-4: PERSONALIZED	R-1 ↑	.175	.179	.201	.204	.189	.193	.197	.193*	.205*	.205	.193	.205	**.209**
NEWS HEADLINE GEN.	R-L ↑	.158	.162	.183	.184	.171	.175	.178	.174*	.186*	.186	.173	.185	**.190**
LAMP-5: PERSONALIZED	R-1 ↑	.485	.486	.501	.505	.488	.508	.509	.488	.510*	**.515***	.490	.509	.512
SCHOLARLY TITLE GEN.	R-L ↑	.436	.439	.450	.453	.432	.448	.456	.439	.458*	**.460***	.439	.457	.459
LAMP-7: PERSONALIZED	R-1 ↑	.516	.514	.552	**.565**	.522	.552	.559	.528*	.563*	**.565**	.529	.559	.561
TWEET PARAPHRASING	R-L ↑	.463	.457	.511	.517	.475	.512	.517	.482*	**.521***	.519	.480	.515	.519

select 25% of users to train the base model for task adaptation. From the remaining users, we randomly sample 100 to serve as test users for efficient and fair comparison with OPPU (Tan et al., 2024). The rest of the users are used as sharer candidates who consent to share their PEFT parameters.[3]

Baselines We compare our proposed PER-PCS with the non-personalized baseline, prompt-based methods (retrieval-augmented (Salemi et al., 2023) and profile-augmented personalization (Richardson et al., 2023)), and PEFT-based personalization methods (PEFT retrieval (Zhao et al., 2024) and OPPU (Tan et al., 2024)). Although PEFT retrieval has not been applied to personalization before, we employ it as a PEFT-level composition baseline. compared with PER-PCS, OPPU requires significantly more resources, which can be seen as the upper bound for sharer personal PEFT composition. We provide more baseline details in Appendix G. For all baselines and PER-PCS, we use `Llama-2-7B` (Touvron et al., 2023) as the base LLM and BM25 (Trotman et al., 2014) for retrieval operations to ensure efficient and fair comparisons.

Evaluation Metrics Following LaMP (Salemi et al., 2023), we use accuracy and F1-score for personalized text classification tasks (LaMP-1 and LaMP-2), and MAE and RMSE for LaMP-3: personalized product rating. For personalized text generation tasks (LaMP-4, LaMP-5, and LaMP-7), we adopt ROUGE-1 and ROUGE-L (Lin, 2004). Higher scores indicate better performance for all metrics except RMSE and MAE used in LaMP-3.

[3]Statistics are presented in Table 4.

4 Results

Table 1 shows the performance on the curated test set of six public tasks in the LaMP benchmark. We have observations as follows.

Performance with PER-PCS. Models equipped with PER-PCS outperform non-personalized, RAG, and PAG counterparts across all six tasks. In personalized text classification, PER-PCS achieves 11.79% and 6.02% relative gains in accuracy and F1-score for movie tagging, and 27.32% and 24.92% improvements in MAE and RMSE for product ratings. For personalized text generation, PER-PCS shows 4.25% and 4.28% relative improvements in ROUGE-1 and ROUGE-L scores for news headline generation. These results demonstrate PER-PCS's effectiveness in enhancing LLM personalization.

PER-PCS *vs.* PEFT Retrieval. Compared to the PEFT retrieval method, PER-PCS shows clear superiority. For instance, PER-PCS achieves 8.76% and 22.09% performance gains in accuracy and F1-score for citation identification. Significant improvements are also seen in movie tagging, product rating prediction, and news headline generation tasks, highlighting the benefits of fine-grained PEFT piece composition over PEFT-level composition, which may risk user data leakage.

PER-PCS *vs.* OPPU. Compared to OPPU, which trains personal PEFT from scratch and requires more computational and storage resources, PER-PCS achieves comparable or slightly better results. Specifically, PER-PCS achieves 99.28% of OPPU's performance on average with 7 times less computation and 38 times less storage in personalized text

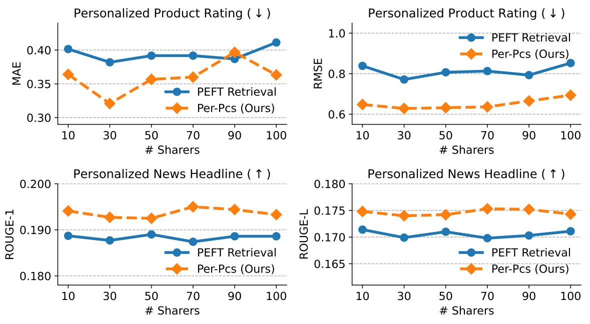

Figure 3: Model performance with different numbers of sharers in the product rating task (lower values indicate better performance). Our piece-level composition Per-Pcs is stable and consistently outperforms the PEFT-level composition baseline.

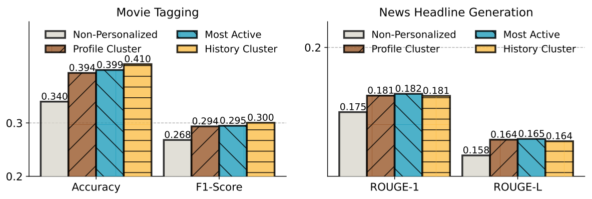

Figure 4: Performance of Per-Pcs on movie tagging and news headline generation tasks with different sharer selection strategies. We find Per-Pcs is robust to the choice of sharers.

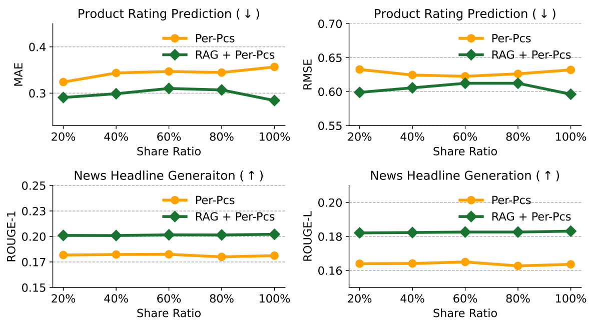

Figure 5: The Per-Pcs performance with different PEFT parameter sharing ratios. Per-Pcs maintains stable performance with a small sharing ratio, while non-parametric user knowledge via RAG enhances stability and performance.

classification. In personalized text generation, Per-Pcs shows comparable or better results in scholarly title generation and tweet paraphrasing.

Per-Pcs with Non-Parametric Knowledge. Integrating both parametric user knowledge in personal PEFT and non-parametric in retrieval and user profile leads to notable performance gain. Averaging all tasks, RAG and PAG bring 1.67% and 12.4% performance gain in text classification tasks, as well as 6.11% and 6.35% enhancement in text generation tasks.

Note that introducing RAG and PAG means users would expose their historical data or profiles to a centralized LLM, raising concerns about how user data are stored, used, and protected, and potentially affecting model ownership. For users prioritizing privacy and ownership, pure Per-Pcs without retrieval avoids revealing user data to centralized LLM, and our experiments show it significantly outperforms non-personalized baselines. Conversely, those seeking optimal performance and consent to reveal data to centralized LLMs should opt for Per-Pcs+RAG/PAG.

5 Analysis

Robustness against Sharer Count In real-world deployment, the number of users who consent to share their personal PEFT can vary, and computational resources may constrain the number of sharers, making the sharer count a crucial factor in Per-Pcs. In this experiment, we alter the number of sharers in two representative tasks from the text classification and generation categories to test the model's robustness. As shown in Figure 3, Per-Pcs exhibits relatively stable performance despite changes in the number of sharers and achieves the best performance with just 30 sharers in the personalized product rating prediction task, demonstrating its strong efficiency. Compared with the PEFT-level composition baseline (PEFT Retrieval), Per-Pcs consistently shows better performance, highlighting the effectiveness of fine-grained piece-level composition in user modeling.

On Sharer Selection Strategy In the main results, we present findings based on selecting sharers by clustering user history embeddings. However, users can have diverse distributions, and those who consent to share PEFT parameters may be biased in their distribution. Therefore, we tested Per-Pcs with different sharer selection strategies to demonstrate its robustness against sharer selection. Specifically, we tested three strategies by restricting the sharer number to 50 users: "Most Active," which selects the 50 most active users; "Profile Cluster," which uses a `DeBERTa-v3-Large` encoder to obtain user embeddings for k-means clustering; and "History Cluster," the default setting, which averages user history embeddings to obtain user embeddings for clustering. As shown in Figure 4, all sharer selection strategies lead to better performance than the non-personalized baseline. Furthermore, Per-

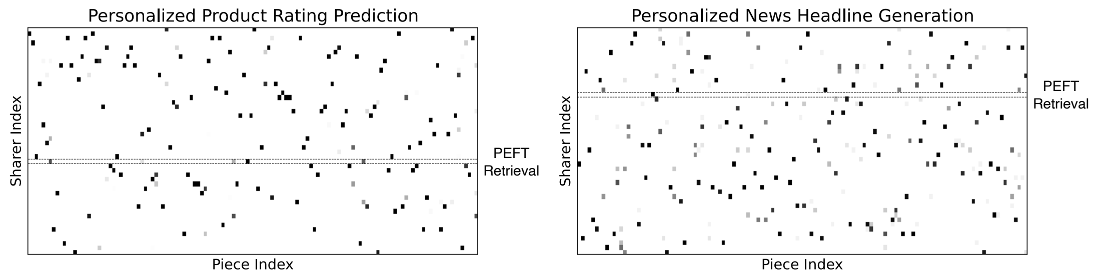

Figure 6: Case study on a specific user's PEFT assembled from sharers and corresponding piece weights in PER-PCS, compared with the PEFT retrieval choice. Unlike PEFT-level retrieval, PER-PCS models user history data in a more fine-grained manner while ensuring the privacy of sharers.

PCS's performance remains stable across different sharer selection strategies, demonstrating its robustness. We hypothesize that fine-grained piece-level parameter composition can decompose complex user preferences from diverse dimensions, facilitating robustness against different sharer distributions.

Shared Pieces Ratio Study We designed PER-PCS to enable sharers to share a small portion of their PEFT parameters, preserving user privacy while maintaining strong performance. In this experiment, we varied the PEFT parameter sharing ratio and assessed its impact on model performance. Shown in Figure 5, using two representative tasks from text classification and generation, we found that PER-PCS is highly robust to the sharing ratio, achieving comparable performance with just 20% of the sharers' PEFT parameters compared to full parameter sharing. Additionally, with non-parametric user knowledge from RAG, PER-PCS demonstrates greater stability and performance. These results show that PER-PCS effectively balances privacy preservation and model performance.

Case Study To better understand the mechanism of piece-level composition in PER-PCS, we conducted a case study on piece selection and corresponding composition weights in product rating prediction and news headline generation, representing text classification and generation categories, respectively. As illustrated in Figure 6, we observe that in both text classification and generation tasks, the selected pieces are diverse. Additionally, the weight distribution in generation tasks is more uniform, likely due to the intrinsic complexity of personality in text generation tasks. Compared with PEFT-level retrieval, we find that PER-PCS almost never selects the same PEFT chosen by retrieval, yet it outperforms PEFT retrieval by 19.11% and 4.15% in product rating prediction and news headline generation tasks, respectively. We speculate that this is due to PER-PCS's ability to effectively decompose and combine sharer PEFT pieces in a fine-grained manner, leveraging multiple sharers' parameters to enhance generalization.

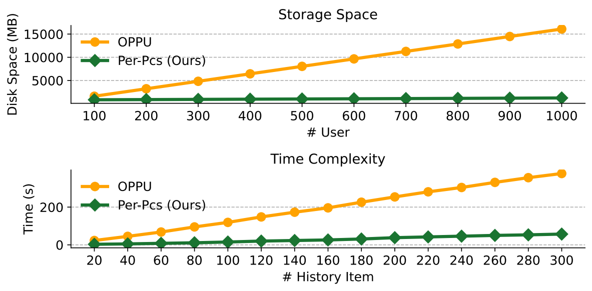

Figure 7: Comparison of storage and time complexity between our PER-PCS and OPPU, demonstrating that PER-PCS requires significantly less time to assemble personal PEFTs and less storage space to save them.

Time and Space Complexity Analysis Scalability and efficiency are crucial for large-scale deployment of personalization methods. We compared PER-PCS and OPPU in terms of storage and assembly time. For storage efficiency, we used the product rating prediction task, observing requirements as the user count increased. For time efficiency, we examined a single user in the movie tagging task by varying the number of user history items. As shown in Figure 7, PER-PCS is significantly more efficient than OPPU in both storage and time. With increasing numbers of users and history items, PER-PCS's efficiency advantage becomes even more pronounced, being approximately 38 times more efficient in storage and 7 times more efficient in time. Moreover, as the number of users and history items grows, the efficiency advantage of PER-PCS becomes even more pronounced, being approximately 38 times more efficient in storage and 7 times more efficient in time.

6 Related Work

6.1 Personalization of LLMs

Existing LLM personalization methods can be categorized into prompt-based and Parameter Efficient Fine-tuning (PEFT)-based methods.

Prompt-based personalization method focuses on designing prompts that incorporate user-generated content and behavior to help LLMs understand user preferences, which can be further categorized into vanilla personalized prompting, retrieval-augmented personalized prompting, and profile-augmented personalized prompting. Vanilla personalized prompting leverages LLMs' in-context learning and few-shot learning abilities by encoding either complete or randomly sampled user history behaviors as contextual examples (Dai et al., 2023; Wang et al., 2023; Kang et al., 2023). To manage the rapidly growing user behavior and LLMs' limited context window, researchers have proposed retrieval-augmented methods for personalized LLMs (Salemi et al., 2023), and enhance the calibration (Mysore et al., 2023) and optimize retrieval (Salemi et al., 2024). Moving beyond simple retrieval, some researchers have proposed profile-augmented personalization prompting, summarizing natural language user preferences and behavior patterns to augment user queries (Richardson et al., 2023), and constructing hierarchy personalized retrieval databases (Sun et al., 2024).

PEFT-based personalization methods store user preferences and behavior patterns in parameters. OPPU (Tan et al., 2024) equips each user with a personal PEFT module, storing preferences in PEFT parameters and offering better generalization of user behavior patterns compared to prompt-based methods. Another line of work focuses on designing personalized alignment methods via parameter merging (Jang et al., 2023a), personalized RLHF (Li et al., 2024; Park et al., 2024), personalized reward models (Cheng et al., 2023), and black-box LLM personalization (Zhuang et al., 2024).

6.2 Model Parameter Composition

Existing work has shown that performing weighted linear interpolation of model parameters leads to the composition of each model ability (Li et al., 2022; Tam et al., 2023). This approach recycles efforts and computational resources used to create specialized models. These methods can be divided into model-, PEFT-, and piece-level compositions.

Model-level composition methods treat the entire model parameter as the minimum composition unit (Wortsman et al., 2022; Choshen et al., 2022; Ramé et al., 2023; Jin et al., 2022). Ilharco et al. (2022) propose the task vector, which subtracts the weights of a fine-tuned model from the pre-trained weights and conducts task vector arithmetic to enable generalization across tasks and domains. PEFT offers lightweight alternatives for fine-tuning LLMs by updating small, plug-in parameters while keeping the pre-trained weights frozen to save computational resources (He et al., 2021). In *PEFT-level composition*, the entire PEFT module is treated as the minimum unit. By composing PEFT parameters, models can achieve task and domain generalization (Shah et al., 2023; Gou et al., 2023; Zhang et al., 2023). LoRAHub (Huang et al., 2023) uses a black-box optimizer to integrate specialized LoRAs, facilitating generalization to unseen tasks. Another line of work focuses on retrieving PEFT (Jang et al., 2023b). LoRARetriever (Zhao et al., 2024) retrieves and composes multiple LoRAs based on the given input. In *piece-level composition*, the minimum composition unit is a plug-in sub-component of PEFT within a specific layer. For instance, in LoRA, each low-rank update at a linear layer constitutes a "piece." Muqeeth et al. (2024) focuses on task generalization and proposes recycling PEFT pieces by employing per-token and per-piece composition under zero-shot settings.

In this work, we propose PER-PCS, a personal PEFT sharing framework that takes advantage of piece-wise parameter composition, enabling users to share partial parameters. This approach ensures the sharer's privacy while maintaining model ownership, fine-grained user modeling, and strong efficiency for personalized LLM democratization.

7 Conclusion

We proposed PER-PCS, a novel framework that enables users to share their personal PEFTs, creating community value while preserving privacy. For target users, PER-PCS maintained model ownership, efficiency, and fine-grained personalization by employing piece-level composition based on user history data. Extensive experiments showed that PER-PCS outperforms non-personalized and PEFT retrieval methods, and performs close to OPPU with significantly lower computational and storage resources. We envisioned PER-PCS as a community-driven effort to advance personalized LLM, making it more modular, effective, and widely accessible.

8 Limitations

We identify two key limitations in PER-PCS. First, constrained by the dataset, our focus is primarily on one specific task per user rather than examining user behaviors across multiple tasks and domains. For instance, in the movie tagging task, users are solely engaged in that specific activity, without the inclusion of behaviors from other domains or platforms. Despite this, the PER-PCS framework is inherently adaptable to any text sequence generation task and is compatible with diverse user instructions across various tasks and domains. Personalizing LLM across a broader range of tasks and domains is left as future work. Second, despite our proposed PER-PCS is compatible with all PEFT methods that introduce trainable modules throughout the model, such as Adapter (Houlsby et al., 2019), $(\mathrm{IA})^3$ (Liu et al., 2022), and prefix tuning (Li and Liang, 2021), we primarily focus on LoRA in this work. This is due to LoRA's popularity, widespread use, and superior performance demonstrated by OPPU (Tan et al., 2024), while we expect to expand our experiment and analysis to more PEFT methods in future work.

9 Ethical Considerations

Data Bias Personalizing LLMs relies heavily on personal data input into the system. If this data is biased or unrepresentative, the model's outputs could perpetuate these biases, leading to unfair or prejudiced responses. It is crucial to monitor and mitigate such biases in personal data and personalized models to ensure fair, unbiased, and safe responses from personalized LLMs. In PER-PCS, where users build personal PEFTs through collaborative efforts, bias in user data could spread within the community, amplifying negative effects. Future work could focus on preventing harmful biases in user data at both the personal and community levels.

Accessibility While advancing personalized LLMs aims to enhance user interactions with AI systems, their complexity and resource-intensive nature can pose accessibility challenges. Smaller entities or individual researchers with limited computational power and budgetary constraints may struggle to engage with advanced personalized LLMs, potentially widening the gap in AI research and application. Efforts should be made to make these technologies more accessible to a broader audience to ensure equitable advancement in AI research.

References

Hamdan A Alamri, Sunnie Watson, and William Watson. 2021. Learning technology models that support personalization within blended learning environments in higher education. *TechTrends*, 65:62–78.

Jinheon Baek, Nirupama Chandrasekaran, Silviu Cucerzan, Sujay Kumar Jauhar, et al. 2023. Knowledge-augmented large language models for personalized contextual query suggestion. *arXiv preprint arXiv:2311.06318*.

Jin Chen, Zheng Liu, Xu Huang, Chenwang Wu, Qi Liu, Gangwei Jiang, Yuanhao Pu, Yuxuan Lei, Xiaolong Chen, Xingmei Wang, et al. 2023. When large language models meet personalization: Perspectives of challenges and opportunities. *arXiv preprint arXiv:2307.16376*.

Pengyu Cheng, Jiawen Xie, Ke Bai, Yong Dai, and Nan Du. 2023. Everyone deserves a reward: Learning customized human preferences. *Preprint*, arXiv:2309.03126.

Leshem Choshen, Elad Venezian, Noam Slonim, and Yoav Katz. 2022. Fusing finetuned models for better pretraining. *arXiv preprint arXiv:2204.03044*.

Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering chatgpt's capabilities in recommender systems. *arXiv preprint arXiv:2305.02182*.

Cosmina Andreea Dejescu, Lucia V Bel, Iulia Melega, Stefana Maria Cristina Muresan, and Liviu Ioan Oana. 2023. Approaches to laparoscopic training in veterinary medicine: A review of personalized simulators. *Animals*, 13(24):3781.

Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-rec: Towards interactive and explainable llms-augmented recommender system. *arXiv preprint arXiv:2303.14524*.

Dmitri Goldenberg, Kostia Kofman, Javier Albert, Sarai Mizrachi, Adam Horowitz, and Irene Teinemaa. 2021. Personalization in practice: Methods and applications. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 1123–1126.

Yunhao Gou, Zhili Liu, Kai Chen, Lanqing Hong, Hang Xu, Aoxue Li, Dit-Yan Yeung, James T Kwok, and Yu Zhang. 2023. Mixture of cluster-conditional lora experts for vision-language instruction tuning. *arXiv preprint arXiv:2312.12379*.

Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, and Dawei Yin. 2020. Hierarchical user profiling for e-commerce recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 223–231.

Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. 2020. Array programming with numpy. *Nature*, 585(7825):357–362.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2022. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. In *The Eleventh International Conference on Learning Representations*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

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

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. 2023. Lorahub: Efficient cross-task generalization via dynamic lora composition. *arXiv preprint arXiv:2307.13269*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*.

Joel Jang, Seungone Kim, Bill Yuchen Lin, Yizhong Wang, Jack Hessel, Luke Zettlemoyer, Hannaneh Hajishirzi, Yejin Choi, and Prithviraj Ammanabrolu. 2023a. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*.

Joel Jang, Seungone Kim, Seonghyeon Ye, Doyoung Kim, Lajanugen Logeswaran, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2023b. Exploring the benefits of training expert language models over instruction tuning. In *International Conference on Machine Learning*, pages 14702–14729. PMLR.

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

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2022. Dataless knowledge fusion by merging weights of language models. In *The Eleventh International Conference on Learning Representations*.

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.

Wang-Cheng Kang, Jianmo Ni, Nikhil Mehta, Maheswaran Sathiamoorthy, Lichan Hong, Ed Chi, and Derek Zhiyuan Cheng. 2023. Do llms understand user preferences? evaluating llms on user rating prediction. *Preprint*, arXiv:2305.06474.

Hannah Rose Kirk, Bertie Vidgen, Paul Röttger, and Scott A Hale. 2024. The benefits, risks and bounds of personalizing the alignment of large language models to individuals. *Nature Machine Intelligence*, pages 1–10.

Seongyun Lee, Sue Hyun Park, Seungone Kim, and Minjoon Seo. 2024. Aligning to thousands of preferences via system message generalization. *arXiv preprint arXiv:2405.17977*.

Cheng Li, Mingyang Zhang, Qiaozhu Mei, Yaqing Wang, Spurthi Amba Hombaiah, Yi Liang, and Michael Bendersky. 2023a. Teach llms to personalize–an approach inspired by writing education. *arXiv preprint arXiv:2308.07968*.

Jiacheng Li, Ming Wang, Jin Li, Jinmiao Fu, Xin Shen, Jingbo Shang, and Julian McAuley. 2023b. Text is all you need: Learning language representations for sequential recommendation. *arXiv preprint arXiv:2305.13731*.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A Smith, and Luke Zettlemoyer. 2022. Branch-train-merge: Embarrassingly parallel training of expert language models. *arXiv preprint arXiv:2208.03306*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Xinyu Li, Zachary C Lipton, and Liu Leqi. 2024. Personalized language modeling from personalized human feedback. *arXiv preprint arXiv:2402.05133*.

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.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965.

Sheng Lu, Irina Bigoulaeva, Rachneet Sachdeva, Harish Tayyar Madabushi, and Iryna Gurevych. 2023. Are emergent abilities in large language models just in-context learning? *arXiv preprint arXiv:2309.01809*.

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.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064.

Mohammed Muqeeth, Haokun Liu, Yufan Liu, and Colin Raffel. 2024. Learning to route among specialized experts for zero-shot generalization. *arXiv preprint arXiv:2402.05859*.

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*.

Chanwoo Park, Mingyang Liu, Kaiqing Zhang, and Asuman Ozdaglar. 2024. Principled rlhf from heterogeneous feedback via personalization and preference aggregation. *arXiv preprint arXiv:2405.00254*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

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.

Alexandre Ramé, Kartik Ahuja, Jianyu Zhang, Matthieu Cord, Léon Bottou, and David Lopez-Paz. 2023. Model ratatouille: Recycling diverse models for out-of-distribution generalization. In *International Conference on Machine Learning*, pages 28656–28679. PMLR.

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, Surya Kallumadi, and Hamed Zamani. 2024. Optimization methods for personalizing large language models through retrieval augmentation. *arXiv preprint arXiv:2404.05970*.

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

Viraj Shah, Nataniel Ruiz, Forrester Cole, Erika Lu, Svetlana Lazebnik, Yuanzhen Li, and Varun Jampani. 2023. Ziplora: Any subject in any style by effectively merging loras. *arXiv preprint arXiv:2311.13600*.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR.

Biplav Srivastava, Francesca Rossi, Sheema Usmani, and Mariana Bernagozzi. 2020. Personalized chatbot trustworthiness ratings. *IEEE Transactions on Technology and Society*, 1(4):184–192.

Chenkai Sun, Ke Yang, Revanth Gangi Reddy, Yi R Fung, Hou Pong Chan, ChengXiang Zhai, and Heng Ji. 2024. Persona-db: Efficient large language model personalization for response prediction with collaborative data refinement. *arXiv preprint arXiv:2402.11060*.

Derek Tam, Mohit Bansal, and Colin Raffel. 2023. Merging by matching models in task subspaces. *arXiv preprint arXiv:2312.04339*.

Zhaoxuan Tan and Meng Jiang. 2023. User modeling in the era of large language models: Current research and future directions. *arXiv preprint arXiv:2312.11518*.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to bm25 and language models examined. In *Proceedings of the 19th Australasian Document Computing Symposium*, pages 58–65.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Danqing Wang, Kevin Yang, Hanlin Zhu, Xiaomeng Yang, Andrew Cohen, Lei Li, and Yuandong Tian. 2023. Learning personalized story evaluation. *arXiv preprint arXiv:2310.03304*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. 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.

Chuhan Wu, Fangzhao Wu, Yongfeng Huang, and Xing Xie. 2023. Personalized news recommendation: Methods and challenges. *ACM Transactions on Information Systems*, 41(1):1–50.

Jinghan Zhang, Junteng Liu, Junxian He, et al. 2023. Composing parameter-efficient modules with arithmetic operation. *Advances in Neural Information Processing Systems*, 36:12589–12610.

Shuo Zhang and Krisztian Balog. 2020. Evaluating conversational recommender systems via user simulation. In *Proceedings of the 26th acm sigkdd international conference on knowledge discovery & data mining*, pages 1512–1520.

Ziyu Zhao, Leilei Gan, Guoyin Wang, Wangchunshu Zhou, Hongxia Yang, Kun Kuang, and Fei Wu. 2024. Loraretriever: Input-aware lora retrieval and composition for mixed tasks in the wild. *arXiv preprint arXiv:2402.09997*.

Yuchen Zhuang, Haotian Sun, Yue Yu, Qifan Wang, Chao Zhang, and Bo Dai. 2024. Hydra: Model factorization framework for black-box llm personalization. *arXiv preprint arXiv:2406.02888*.

Table 2: Performance of PER-PCS across different ablated versions: Top-p refers to setting a cumulative probability threshold p and aggregating all pieces that first reach this threshold. Topk-Sampling denotes sampling one piece from the top k pieces with normalized scores as probabilities.

Ablation Settings	**LaMP-2**		**LaMP-4**	
	Acc	F1	R-1	R-L
full model	0.410	0.301	0.193	0.174
w/o attention	0.340	0.266	0.176	0.158
replace Topk-Agg. w/ Topp-Agg.	0.390	0.288	0.169	0.153
replace Topk-Agg. w/ Topk-Sampling	0.383	0.297	0.172	0.155

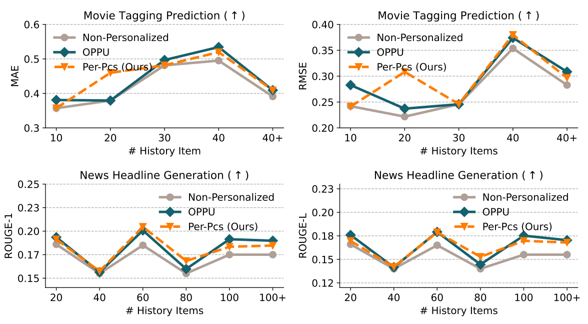

Figure 8: Model performance on personalized movie tagging and news headline generation for users with different numbers of history items.

A Ablation Study

As PER-PCS outperforms various baselines in personalization tasks, we investigate the impact of each design choice in PER-PCS to verify their effectiveness. More specifically, we perform ablation on the assembling process in both attention aggregation and piece selection steps. As is shown in Table 2, the full PER-PCS outperforms all ablated models, proving our design choice's effectiveness. Moreover, the weighted aggregation of PEFT pieces has a significant impact on performance and is essential for model generalization for target users. We also find that TopP and TopK sampling strategies for pieces strategy would involve randomness and noises and eventually hurt the model performance.

B Computation Resources Details

All experiments are implemented on a server with 3 NVIDIA A6000 GPU and Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz with 20 CPU cores.

C Hyperparameters

The hyperparameters of PER-PCS are presented in Table 3 to facilitate further research.

D Modeling Users with Different Active Levels

Users can exhibit different levels of activity, resulting in varying lengths of user history items for user modeling and personalization. To investigate the impact of user activity levels, quantified by the number of historical behavior items, on model performance, we randomly sampled 10 users from each range of activity levels. As shown in Figure 8, we observe that (*i*) PER-PCS generally shows stronger relative performance when user behavior items are fewer than 20, likely due to the collaborative signals captured during the assembling process that help the model understand user preferences. (*ii*) PER-PCS generally performs similarly to OPPU, which requires training and maintaining personal PEFT from scratch and significantly more resources, and (*iii*) both OPPU and PER-PCS outperform the non-personalized baseline at almost all activity levels. Overall, these results demonstrate the strong performance and robustness of PER-PCS across all user activity levels.

E Scientific Artifacts

PER-PCS is built with the help of many existing scientific artifacts, including PyTorch (Paszke et al., 2019), Numpy (Harris et al., 2020), huggingface, and transformers (Wolf et al., 2020). We will make the PER-PCS implementation publicly available to facilitate further research.

F Task Details

We present the task details as follows to help readers gain a better understanding of the task format.

- **Personalized Citation Identification** is a binary text classification task. Specifically, given user u writes a paper x, the task aims to make the model determine which of the two candidate papers u will cite in paper x based on the user's history data, which contains the publications of user u.

- **Personalized News Categorization** is a 15-way text classification task to classify news articles written by a user u. Formally, given a news article x written by user u, the language model is required to predict its category from the set of categories based on the user's history data, which contains the user's past article and corresponding category.

Table 3: Hyperparameter settings of PER-PCS across six tasks on LaMP data.

Task	Sharer PEFT			Sharer Gate			PER-PCS Assemble	
	batch size	epoch	lr	batch size	step	lr	top-k	batch size
LAMP-1: PERSONALIZED CITATION IDENTIFICATION	16	1	1e-5	6	100	1e-5	1	16
LAMP-2: PERSONALIZED MOVIE TAGGING	6	3	2e-5	6	100	2e-5	3	16
LAMP-3: PERSONALIZED PRODUCT RATING	2	2	1e-5	4	100	1e-5	1	6
LAMP-4: PERSONALIZED NEWS HEADLINE GEN.	10	3	2e-5	6	50	2e-5	1	16
LAMP-5: PERSONALIZED SCHOLARLY TITLE GEN.	3	2	2e-5	6	50	2e-5	1	10
LAMP-7: PERSONALIZED TWEET PARAPHRASING	16	2	1e-5	6	50	2e-5	2	16

- **Personalized Movie Tagging** is a 15-way text classification task to make tag assignments aligned with the user's history tagging preference. Specifically, given a movie description x, the model needs to predict one of the tags for the movie x based on the user's historical movie-tag pairs.

- **Personalized Product Rating** is a 5-way text classification task and can also be understood as a regression task. Given the user u's historical review and rating pairs and the input review x, the model needs to predict the rating corresponding to x selected from 1 to 5 in integer.

- **Personalized News Headline Generation** is a text generation task to test the model's ability to capture the stylistic patterns in personal data. Given a query x that requests to generate a news headline for an article, as well as the user profile that contains the author's historical article-title pairs, the model is required to generate a news headline specifically for the given user.

- **Personalized Scholarly Title Generation** is a text generation task to test personalized text generation tasks in different domains. In this task, we require language models to generate titles for an input article x, given a user profile of historical article-title pairs for an author.

- **Personalized Tweet Paraphrasing** is also a text generation task that tests the model's capabilities in capturing the stylistic patterns of authors. Given a user input text x and the user profile of historical tweets, the model is required to paraphrase x into y that follows the given user's tweet pattern.

Table 4: Dataset statistics: We report average sequence length in terms of number of tokens. #Q is the number of queries, L_{in} and L_{out} are the average length of input and output sequence respectively, and #History is the number of user history items. To save space, task names can be found in Table 1.

Task in LaMP	Sharer Candidates				Target Users			
	#Q	#History	L_{in}	L_{out}	#Q	#History	L_{in}	L_{out}
1	5,334	88.5	51.4	1.0	125	147.2	50.8	1.0
2	2,385	12.3	92.5	1.7	2,228	37.3	92.3	2.0
3	15,034	202.5	132.1	1.0	614	360.6	160.9	1.0
4	7,568	31.3	30.1	10.1	3,949	155.9	26.5	10.7
5	10,821	94.3	162.7	9.7	608	144.0	158.9	9.7
7	9,978	15.7	299.6	16.9	114	77.2	30.3	17.0

G Baseline Details

- **Non-Personalized baseline**: We present two approaches under the non-personalized setting: non-retrieval and random history. *Non-retrieval method* (k=0) refers to only feeding the user's query without revealing the user's behavior history to the LLMs. *Random history* baseline means augmenting the user's query with random history behavior from all user history corpus.

- **Retreival-Augmented Personalization (RAG)**: We follow the retrieval-augmented personalization method presented in LaMP (Salemi et al., 2023), where the user's query is augmented with top k retrieved items from the corresponding user's history corpus. We take k=1 by default in this work.

- **Profile-Augmented Personalization (PAG)**: This method is taken from Richardson et al. (2023), in which the user's input sequence would concatenate the user's profile summarizing the user's preference and behavior patterns. In our experiments, we generate user profiles using the `Mistral-7B` (Jiang et al., 2023) model. Moreover, the profile-augmented method could be combined with the retrieval augmentation. In this case, we take the number of retrieval items k=1 following the setting of Richardson et al. (2023).

- **PEFT Retrieval**: Similar to Jang et al. (2023b); Zhao et al. (2024), when a target user comes, we compute the cosine similarity between embeddings of target users and sharers and find the top-k similar users and conduct weighted aggregation to obtain target user's PEFT. The PEFT retrieval method has not been applied to LLM personalization before and we select it as a PEFT-level composition baseline to compare with our proposed fine-grained piece-level composition method.

- **OPPU**: This method was proposed by Tan et al. (2024), which trains a PEFT for each user from scratch and can be integrated with prompt-based personalization methods. Compared to our PER-PCS, OPPU requests significantly more computation and storage.

H Dataset Statistics

The dataset statistics are presented in Table 4.

I Prompt Details

We present the prompt used in our experiments in this section, where the text in `{BRACES}` can be replaced with content specific to different users and queries. Prompts for user profile generation are presented in Table 5, prompts for personalization tasks are presented in Table 6

Table 5: Prompt for user profile generation.

Task	Prompt
LaMP-1: Personalized Citation Identification	Write a summary, in English, of the research interests and topics of a researcher who has published the following papers. Only generate the summary, no other text. User History: `{USER HISTORY}` Answer:
LaMP-2: Personalized Movie Tagging	Look at the following past movies this user has watched and determine the most popular tag they labeled. Answer in the following form: most popular tag: <tag>. User History: `{USER HISTORY}` Answer:
LaMP-3: Personalized Product Rating	Based on this userś past reviews, what are the most common scores they give for positive and negative reviews? Answer in the following form: most common positive score: <most common positive score>, most common negative score: <most common negative score>. User History: `{USER HISTORY}` Answer:
LaMP-4: Personalized News Headline Generation	Given this authorś previous articles, try to describe a template for their headlines. I want to be able to accurately predict the headline gives one of their articles. Be specific about their style and wording, doní tell me anything generic. User History: `{USER HISTORY}` Answer:
LaMP-5: Personalized Scholarly Title Generation	Given this authorś previous publications, try to describe a template for their titles. I want to be able to accurately predict the title of one of the papers from the abstract. Only generate the template description, nothing else. User History: `{USER HISTORY}` Answer:
LaMP-7: Personalized Tweet Paraphrasing	Given this personś previous tweets, try to describe a template for their tweets. I want to take a generic sentence and rephrase it to sound like one of their tweets, with the same style/punctuation/capitalization/wording/tone/etc. as them. Only give me the template description, nothing else. User History: `{USER HISTORY}` Answer:

Table 6: Prompt for personalization tasks.

Task	Prompt

LaMP-1: Personalized Citation Identification	### User Profile:
	`{USER PROFILE}`
	### User History:
	`{USER HISTORY}`
	### User Instruction:
	Identify the most relevant reference for the listed publication by the researcher. Select the reference paper that is most closely related to the researcherś work. Please respond with only the number that corresponds to the reference.
	Paper Title: `{QUERY PAPER TITLE}` Reference: [1] - `{OPTION1}` [2] - `{OPTION2}`
	Answer:

LaMP-2: Personalized Movie Tagging	### User Profile:
	`{USER PROFILE}`
	### 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 Profile:
	`{USER PROFILE}`
	### 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 Profile:
	`{USER PROFILE}`
	### User History:
	`{USER HISTORY}`
	### User Instruction:
	Generate a headline for the following article.
	Article: `{QUERY ARTICLE}` Headline:

LaMP-5: Personalized Scholarly Title Generation	### User Profile:
	`{USER PROFILE}`
	### User History:
	`{USER HISTORY}`
	### User Instruction:
	Generate a title for the following abstract of a paper.
	Abstract: `{QUERY ABSTRACT}` Title:

LaMP-7: Personalized Tweet Paraphrasing	### User Profile:
	`{USER PROFILE}`
	### User History:
	`{USER HISTORY}`
	### User Instruction:
	Paraphrase the following text into tweet without any explanation before or after it.
	Text: `{QUERY TEXT}` Tweet: