Exploring Personalization Shifts in Representation Space of LLMs

Anonymous ACL submission

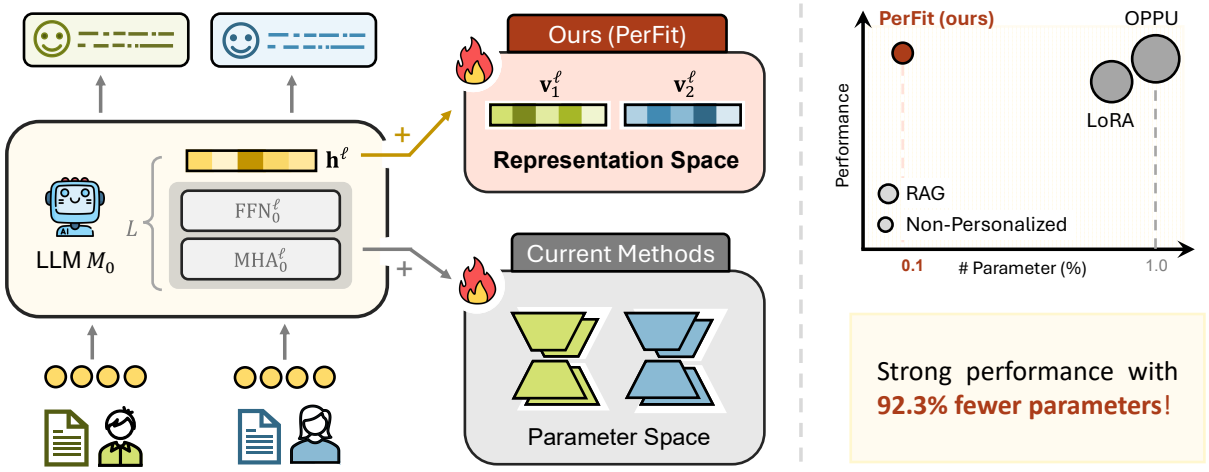
Abstract

Personalization has become a pivotal field of study in contemporary intelligent systems. While large language models (LLMs) excel at general knowledge tasks, they often struggle with personalization, i.e., adapting their outputs to individual user expectations. Existing approaches that steer LLM behavior to meet users' implicit preferences and behavior patterns, primarily relying on tune-free methods (e.g., RAG, PAG) or parameter fine-tuning methods (e.g., LoRA), face challenges in effectively balancing effectiveness and efficiency. Moreover, the mechanisms underlying personalized preferences remain underexplored. To address these challenges, we first **uncover key patterns of user-specific information embedded in the representation space**. Specifically, we find that (1) personalized information lies within a low-rank subspace represented by vectors, and (2) these vectors demonstrate both a collective shift shared across users and a personalized shift unique to each individual user. Building on these insights, we introduce `PerFit`, a novel **two-stage solution that directly fine-tunes interventions in the hidden representation space** by addressing both collective and user-specific shifts, thereby achieving precise steering of LLM with minimal parameter overhead. Experimental results demonstrate that `PerFit` delivers strong performance across six datasets while **cutting the number of parameters by an average of 92.3%** compared to the state-of-the-art methods.

1 Introduction

Large language models (LLMs) demonstrate remarkable abilities in text generation and complex reasoning (Radford et al.; Chang et al., 2024; Hu et al., 2024; Zhang et al., 2024d,c; Zhu et al., 2024; Wang et al., 2023, 2024a), thanks to comprehensive pre-training on diverse and large-scale datasets that equip them with broad general knowledge. Nonetheless, their optimization for wide-ranging tasks means they often struggle to adapt to individual user preferences. For instance, different users may expect distinct outputs even when given the same input. Accordingly, integrating user tastes and preferences into LLMs has propelled personalized large language models (PLLMs) to the forefront of research (Liu et al., 2025; Chen, 2023; Zhang et al., 2024e; Liu et al., 2024). In real-world scenarios, user preferences are often implicit, like writing style and tone (Salemi et al., 2023; Tan et al., 2024b; Zhuang et al., 2024). Enabling LLMs to grasp this implicit information and generalize effectively to user queries remains a core research challenge for PLLMs.

Existing techniques can be broadly categorized into **tune-free methods**, such as retrieval-augmented generation (RAG) (Fan et al., 2024) and profile-augmented generation (PAG), and **parameter-efficient fine-tuning methods** (PEFT), like low-rank adaptation (LoRA) (Hu et al., 2021; Yang et al., 2024). Non-tuned methods (Madaan et al., 2022; Salemi et al., 2023; Zhuang et al., 2024) emphasize efficiency and flexibility by leveraging external information or user profiles without modifying model parameters, but often struggle to achieve high personalization and generalization capability, especially when retrieved contexts contain noise that is misaligned with the user's real intent (Shi et al., 2023). In contrast, parameter fine-tuning methods (Tan et al., 2024b,a; Wagner et al., 2024; Qi et al., 2024) update model parameters based on user data, enabling deeper and better personalization. Taking into account both model performance and the protection of user privacy, a prevalent approach is to allocate an individual PEFT module for each user (Tan et al., 2024b,a; Qi et al., 2024; Wagner et al., 2024; Gao and Zhang, 2024). LoRA still requires millions of parameters for good performance, though it reduces parameter counts (Wu et al., 2024; Cho et al., 2024; Guo et al., 2024). This leads to high communication costs and



(a) Personalized Fine-Tuning (PEFT vs. Ours) (b) Our Advantage

Figure 1: Illustration of our personalized fine-tuning method in representation space PerFit: (a) instead of tuning parameters, PerFit directly fine-tunes the hidden representations, where 🔥 represents fine-tuning with learnable parameters. (b) Experimental results show PerFit similarly strong performance on six datasets while reducing parameters by 92.3% on average compared to OPPU (Tan et al., 2024b).

limited scalability in edge-cloud setups—where the base LLM runs in the cloud and personalized parameters reside on user devices such as phones (Qi et al., 2024; Wagner et al., 2024) (Appendix B). Therefore, **striking a balance between effectiveness and efficiency** remains a significant challenge for existing methods.

To solve this problem, we take the initial step of investigating how personalized information is captured by LLMs, thereby laying the groundwork for our alternative lightweight fine-tuning solution. This effort is motivated by recent advances in activation engineering (Wang et al., 2024b; Arditi et al.; Turner et al., 2023; Zhang et al., 2024b), which allows precise control of LLM outputs by targeting internal representation interventions related to attributes like harmlessness (Bolukbasi et al., 2016; Park et al.), truthfulness (Li et al., 2023), and humor (Von Rütte et al., 2024). Therefore, the key question we investigate in this paper is:

> *Does personalized information induce discernible patterns in LLMs' hidden representation space that enable efficient guidance of model behavior?*

We conduct exploratory experiments to uncover personalized information encoded in the hidden representation space, named δ-vectors (Section 2), revealing **two key observations**. (1) The δ-vectors can be effectively represented within a low-dimensional orthogonal subspace (Observation 1). This suggests learning a **low-rank subspace** to get interventions representing user information in the representation space. (2) Vectors for all users in the low-rank subspace exhibit a clear **collective shift**, characterized by a common direction of deviation. Based on the collective shift, the vectors subsequently disperse towards multiple directions for different users (Observation 2). This suggests a two-stage approach to learn the collective and personalized shifts, respectively.

The intriguing findings inspire our personalized fine-tuning approach, which directly fine-tunes LLMs in the low-rank hidden representation subspace rather than model parameters, named PerFit. Specifically, we first train the collective shift using data from all users, and then, based on this, learn the personalized shifts for each user. To the best of our knowledge, **this is the first work to fine-tune LLMs in representation space tailored to personalized LLM tasks**. The learned collective shift, combined with the personalized shift, is directly added to the model's hidden representation space as an intervention to steer the model's output toward fulfilling individual users' personalized requirements. Experimental results demonstrate that PerFit delivers strong performance across six personalization datasets while **cutting the number of parameters by an average of 92.3%** compared to the LoRA-based methods.

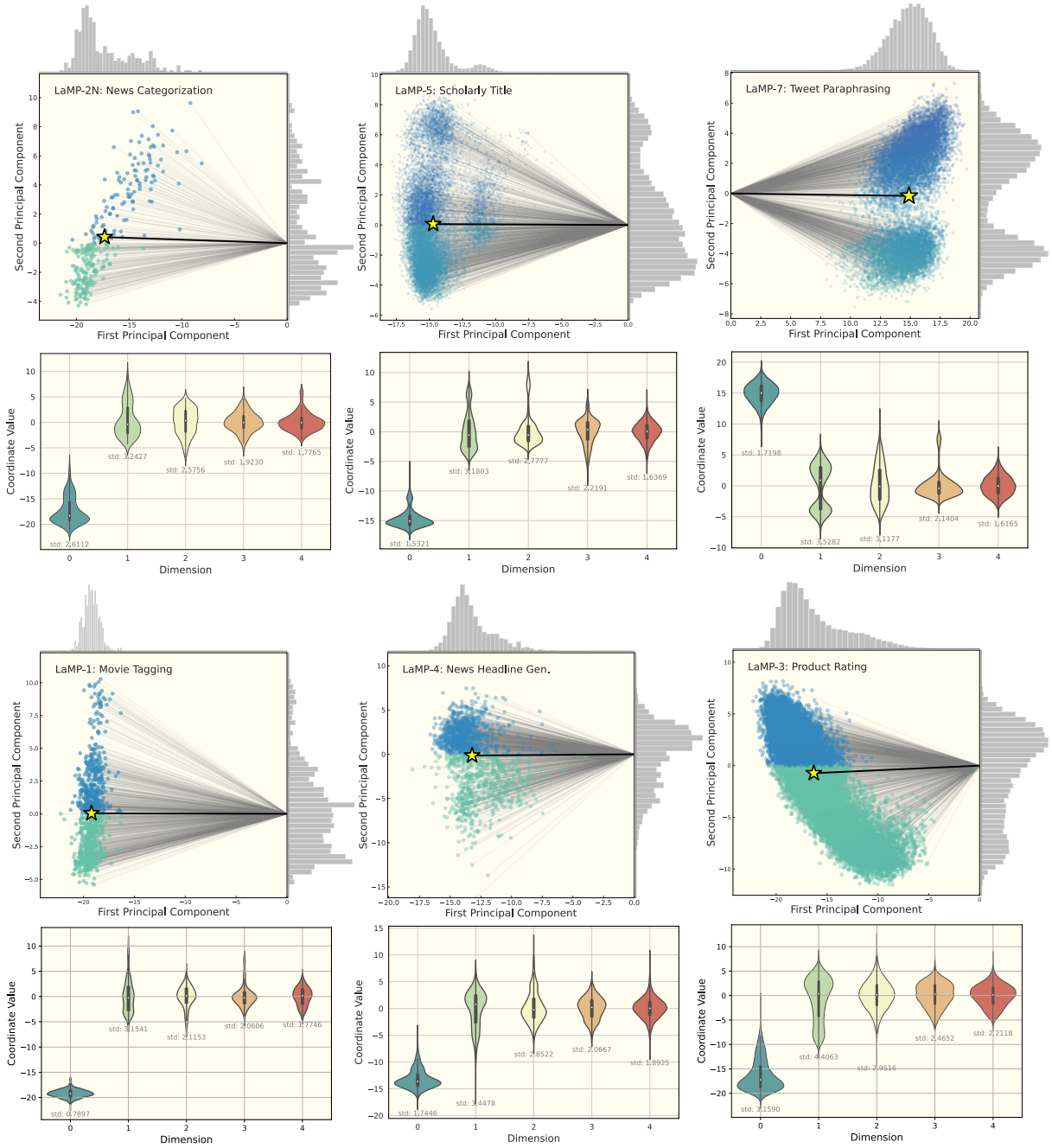


Figure 2: (1) The first row depicts the low-rank vector representations projected onto the first two principal components, with mean vectors indicated by yellow stars, demonstrating directional bias in the reduced-dimensional space. (2) The second row comprises violin plots of the coordinate value distributions across the first five feature dimensions. Here, the zeroth dimension shows a significant mean shift from zero, indicating a shared directional bias (i.e., collective shift) among users, whereas the remaining dimensions have means near zero with relatively large standard deviations, reflecting individual user variability (i.e., personalized shifts).

2 Uncovering Personalization in Representation Space

Building on the insights of activation steering (Appendix C.2), in this section, we aim to investigate *whether patterns related to personalized information exist within the hidden representation space.* If so, we can develop methods that leverage these patterns to guide personalization directly in representation space, *achieving a better balance between effectiveness and parameter efficiency.*

2.1 Extracting personalization vectors

Following the analysis paradigm of activation engineering (Arditi et al.), for each user $u_i \in \mathcal{U}$, given their original query set $\mathcal{Q}_i^{\text{orig}}$,

Table 1: Minimum feature dimensions r needed to explain 0.8 and 0.9 of variance. ‰ represents the r as a per mille of total dimensions.

	0.8		0.9		0.95	
	r	‰	r	‰	r	‰
LaMP-2N	1	3.65	4	14.60	20	72.90
LaMP-5	4	0.98	40	9.77	203	49.56
LaMP-7	3	0.73	32	7.81	177	43.21

we enhance each query by incorporating the most relevant personalized information. The resulting personalization-enhanced query set is denoted as $\mathcal{Q}_i^{\text{per}}$. At layer ℓ, let $\mathbf{h}_t^{(\ell)}(q) \in \mathbb{R}^d$ be the hidden state (residual stream activation) corresponding to the last token t of the input query q. The mean residual representations for the original and personalized inputs are defined as $\boldsymbol{m}_i^{(\ell)} = \frac{1}{|\mathcal{Q}_i^{\text{orig}}|} \sum_{q \in \mathcal{Q}_i^{\text{orig}}} \mathbf{h}_t^{(\ell)}(q), \boldsymbol{n}_i^{(\ell)} = \frac{1}{|\mathcal{Q}_i^{\text{pers}}|} \sum_{q \in \mathcal{Q}_i^{\text{pers}}} \mathbf{h}_t^{(\ell)}(q)$. The *difference-in-means* (Belrose, 2024) personalization vector at the layer ℓ is then $\mathbf{v}_i^{(\ell)} = \boldsymbol{n}_i^{(\ell)} - \boldsymbol{m}_i^{(\ell)}$, which captures the principal change in the model's internal representation induced by personalized information of user i.

Note that personalized information, unlike clear-cut traits such as harmlessness or helpfulness that can be manipulated via a single vector, is inherently more complex and diverse. Therefore, we consider each user a special personality and analyze all users together to capture both collective and personalized aspects. The collection of $\mathbf{v}_i^{(\ell)}$ for all users $i \in \mathcal{U}$ is called **δ-vectors** in this paper for simplicity.

Using Llama2-7B as the base (i.e., non-personalized) LLM (Tan et al., 2024b,a; Kong et al., 2024), we conducted an analytical study on the widely-used personalization benchmark LaMP (Salemi et al., 2024b). To isolate the personalized information, We focus on the residual stream representation of the last token, $\mathbf{h}^\ell := \mathbf{h}_n^\ell$, which aggregates information from the entire input sequence at layer ℓ, specifically analyzing the 16th layer following previous activation steering approaches (Arditi et al.). The personalized information we concatenate for each user $\mathcal{Q}_i^{\text{orig}}$, is derived via the BM25 algorithm to identify the most relevant details of each query from the user's historical documents.

2.2 Observations

Based on the δ-vectors, which isolate the personalized information of all users, we proceed to uncover the underlying personalization patterns. Below are the key observations.

> **Observation 1** (Low-rank Subspace)**.** *The δ-vectors can be effectively represented within a low-dimensional orthogonal subspace, significantly reducing the original feature space dimensionality.*

We performed singular value decomposition (SVD) (Stewart, 1993) on the obtained δ-vectors to determine the intrinsic rank required to represent them with minimal loss of information. Table 1 reveals that the effective rank is significantly lower than the full dimensionality of the feature matrix, accounting for approximately 0.073% of the original dimensions. This observation suggests that the δ-vectors vectors lie predominantly within a low-dimensional orthogonal subspace, suggesting substantial redundancy in the high-dimensional representations.

> **Observation 2** (Collective and Personalized Shifts)**.** *The δ-vectors exhibit a collective shift, accompanied by personalized shifts reflecting individual variability.*

We further plotted the mean and standard deviation of each dimension within the low-rank subspace based on the SVD. As shown in Figure 2, there is a significant shift with small variance in the low-rank subspace, indicating a collective shift across all vectors.

2.3 Personalized Shifts: A Case Study

This section aims to answer the question: "*Do the personalized shifts of the δ-vectors encompass personalization?*" The personalized information regarding implicit styles in the LaMP dataset is challenging to quantify. To explore this, we select samples within the representation space for a case study to determine whether nearby representations exhibit similar styles.

For instance, in the context of Tweet Paraphrase, the template of added personalized information and the queries is illustrated in Figure 3. Using the collective shift vector as a reference, we identify the top 10 users with the nearest vectors, the top 10

Personalized Information:
abstract: <abstract>. title: <title>

Query:
Given this author's 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.
abstract: <query_abstract>. title:

Figure 3: Personalized Information template. Replace the content inside the <> with the actual descriptions of the abstract and title for each query.

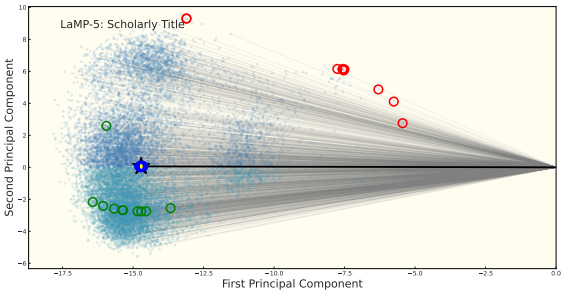


Figure 4: Selected samples from the representation space. The red circle, green circle, and blue circle represent the 10 points that are farthest, at an intermediate distance, and closest to the collective vector.

Closest-10 Closest Users: {title}

- #1: Turning Cliques Into Paths To Achieve Planarity
- #2: **MetaSpace II:** Object and full-body tracking for interaction and navigation in social VR
- #3: **Vmotion:** Designing A Seamless Walking Experience In Vr
- #4: Neural networks and machine learning in bioinformatics **- theory and applications**
- #5: Why traditional usability criteria fall short in ambient assisted living environments
- #6: Intelligent Web Service **- From Web Services** to .Plug&Play. Service Integration
- #7: Outage probability guaranteed relay selection in cooperative communications
- #8: Towards an Object-Oriented Programming Language for Physarum Polycephalum Computing **: A Petri Net Model Approach**
- #9: Contextual Grouping of Labels
- #10: Axioms For Centrality

Figure 5: Closest-10 closest users. These sentences indicate that a majority of individuals prefer to employ punctuation marks, such as commas and dashes.

Intermediate-10 Users: {title}

- #1: Exploiting temporal influence in online recommendation
- #2: On measuring affects of github issues' commenters
- #3: Evaluation of tone mapping operators **using** a High Dynamic Range display
- #4: Differential Entropy Preserves Variational Information of Near-Infrared Spectroscopy Time Series **Associated With** Working Memory
- #5: Toward real-time endoscopically-guided robotic navigation **based on** a 3D virtual surgical field model
- #6: Time-varying noise estimation for speech enhancement and recognition **using** sequential Monte Carlo method
- #7: On collision-free reinforced barriers for multi domain IoT **with** heterogeneous UAVs
- #8: Dense depth maps **by** active color illumination and image pyramids
- #9: New results on optimizing rooted triplets consistency
- #10: Machine learning-based detection of open source license exceptions

Figure 6: Intermediate-10 closest users. The titles of these examples rarely use punctuation marks; instead, they favor terms such as 'based on' and 'using' to indicate specific methodologies. Additionally, the descriptions of the titles are more precise compared to those of the closest users.

vectors at an intermediate distance, and the 10 farthest vectors. We then present their corresponding personalized information regarding the titles generated by the user previously, aligning with the user's

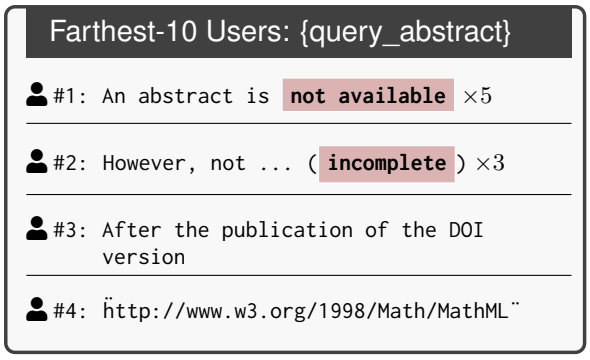


Figure 7: Farthest-10 closest users. The distant points are all outliers, and the query of users lacks effective abstract information. As shown above, there is no indication of the user's methodology.

preferences, as illustrated in Figure 5, Figure 6, and Figure 7, respectively. The points we select are highlighted in Figure 4.

From these examples, it is evident that while the style may not convey significant information, users with personalized vectors that represent different regions in the embedding space exhibit clear and intuitive differences in their corresponding personalized information.

Moreover, an interesting discovery is that the **points that are farther from the collective shift tend to be outliers.** The queries associated with these outlier points lack effective information, resulting in the inability to incorporate meaningful personalization. Consequently, compared to the collective shift, there is a considerable deviation in these instances. This further validates and supports the direct correlation between δ-vectors and personalized information in the embedding space. It also demonstrates that the personalized shift, based on the collective shift, can effectively reflect individualized information.

3 Methodology: PerFit

These findings have practical implications: understanding and isolating personalized representations enables the development of more efficient, lightweight fine-tuning methods with reduced computational demand. Leveraging the observations, the personalized method PerFit is proposed to **directly fine-tune the representation low-rank subspace and the intervention vector**, rather than the model parameters. Inspired by the representation fine-tuning paradigm (Wu et al., 2024), we propose a novel two-stage formulation specifically designed to achieve the personalization goal [1].

Intuitive Explanation. PerFit is designed to align with our key observations (Figure 8).

- $\mathbf{R}$ is an orthogonal matrix that projects vectors from a high-dimensional space onto a low-dimensional subspace, consistent with the **low-rank subspace observation** (Observation 1). Its transpose, $\mathbf{R}^\top$, performs the inverse mapping by projecting vectors from the low-dimensional subspace back to the original high-dimensional space. The intervention vector $\mathbf{v}$ corresponds to the δ-vectors , and the model directly learns these vectors during training.
- Two-stage fine-tuning functions $\phi_{\Delta\Theta^{(2)}} \circ \phi_{\Delta\Theta^{(1)}}$ are designed based on Observation 2 that $\Delta\Theta^{(1)}$ is tuned by all users' data $\mathcal{U}$ to get the **collective shift** for the first stage. Then, we fine-tune $\Delta\Theta_i^{(2)}$ for each user $u_i \in \mathcal{U}$ to get the **personalized shifts**.

PerFit — Personalized Fine-Tuning in Representation Space

$$\Phi_{\text{PerFit}}(\mathbf{h}) = \left(\phi_{\Delta\Theta^{(2)}} \circ \phi_{\Delta\Theta^{(1)}}\right)(\mathbf{h}), \quad (1)$$

where for $s = 1, 2$, $\phi_{\Delta\Theta^{(s)}} :=$

$$\mathbf{x} + \underbrace{\mathbf{R}^{(s)\top}\left(\mathbf{W}^{(s)}\mathbf{x} + \mathbf{b}^{(s)} - \mathbf{R}^{(s)}\mathbf{x}\right)}_{\text{intervention vector } \mathbf{v}^{(s)}} \quad (2)$$

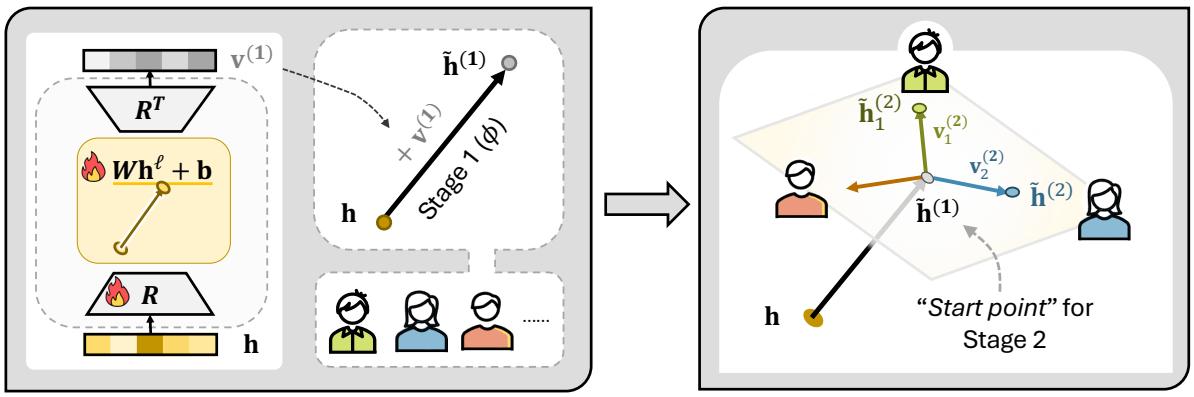
Here, $\mathbf{h}, \mathbf{x} \in \mathbb{R}^d$ and $\circ$ is the functional composition, and $\Delta\Theta^{(s)} = \left(\mathbf{R}^{(s)}, \mathbf{W}^{(s)}, \mathbf{b}^{(s)}\right)$ are trainable parameter sets with $\mathbf{R}^{(s)}, \mathbf{W}^{(s)} \in \mathbb{R}^{r_s \times d}$, $\mathbf{b}^{(s)} \in \mathbb{R}^{r_s}$, where, $r_s \ll d$, $\mathbf{R}^{(s)}$ is a row-wise orthogonal matrix satisfying $\mathbf{R}^{(s)}(\mathbf{R}^{(s)})^\top = \mathbf{I}_{r_s}$.

4 Experiments

We conduct extensive experiments to evaluate our proposed PerFit method across six diverse tasks from the LaMP benchmark. Our evaluation mainly focuses on the following three research questions:

- **RQ1.** How does PerFit perform compared to state-of-the-art personalized approaches in

[1] For simplicity, we remove the layer index ℓ in the notation.



(a) **Stage 1**: Collective shift (b) **Stage 2**: Personalized shifts

Figure 8: Illustration of two-stage personalized fine-tuning PerFit. (a) The first stage tunes on all users to obtain the collective shift. (b) The second-stage intervention vector is learned from the intervened representation of stage 1 and fine-tuned individually for each user.

terms of both effectiveness and efficiency? (Section 4.2)

- **RQ2.** To what extent does PerFit improve computational and memory efficiency while maintaining competitive performance? (Section 4.3)
- **RQ3.** How does our two-stage training approach contribute to the model's performance, and what is the impact of each stage? (Section 4.4)

4.1 Experimental Setup

This section outlines the experimental settings for evaluating our proposed PerFit method. We describe the datasets, baseline models, and key implementation parameters used in our evaluation. For additional setup details, please refer to the corresponding subsections in Appendix D.

Datasets. We conduct experiments on six diverse tasks from the LaMP benchmark (Salemi et al., 2024b): three classification tasks (News Categorization, Movie Tagging, Product Rating) and three generation tasks (News Headline Generation, Scholarly Title Generation, Tweet Paraphrasing). Following established practices (Tan et al., 2024b), data from approximately 100 users with the most extensive interaction histories for each task constitute our test set, while the remaining data is used for training the base (i.e., non-personalized) LLM. Detailed dataset statistics are provided in Appendix D.1.

Baselines. PerFit is compared against a range of baselines, all implemented using Llama2-7B as the base model. These include *Non-Tuned Methods*: **Non-Personalized**, Profile Augmented Generation (**PAG**) (Richardson et al., 2023), Retrieval Augmented Generation (**RAG**) (Salemi et al., 2024b) (with $k \in \{1, 2, 4\}$ retrieved documents), and **StyleVector** (Zhang et al., 2025); and *Tuned Methods*: Collective LoRA (Hu et al., 2021) (**LoRA-C**), Personalized LoRA (**LoRA-P**), **OPPU** (Tan et al., 2024b). Details of each baseline are available in Appendix D.2.

Implementation Details. Key training settings are consistent across both training stages: we use the AdamW optimizer with a learning rate of 1×10^{-4}, weight decay of 1×10^{-2}, and BF16 precision. Gradient clipping is applied with a maximum norm of 0.3. Batch sizes are generally 16, with exceptions for Product Rating (batch size 2) and Scholarly Title Generation (batch size 4) due to computational requirements. The base LLM is trained for 3 epochs, and the personal PEFT stage for 2 epochs. For inference, we set the temperature to 0.1, top-k sampling to 10, and top-p sampling to 0.9. PEFT-based methods (LoRA, OPPU) utilize a LoRA rank $r = 8$ and $\alpha = 8$. For our representation-based methods, hyperparameters such as low-rank dimensions, intervention layers, and positions were determined via a 20-trial random search.

4.2 Main Results (RQ1)

We evaluate PerFit against state-of-the-art personalized approaches across six diverse tasks from the LaMP benchmark. The results are presented in Tables 2 and 3, which demonstrate the effectiveness of our method in both personalized classification and generation scenarios.

Table 2: Results on classification tasks. We report Accuracy (Acc) and F1 Score (F1) for LaMP-2N and LaMP-2M, and Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) for LaMP-3. For PerFit, we show the parameter percentage relative to the total model and the parameter reduction compared to OPPU. Blue and red numbers represent Stage-1 and Stage-2 parameters, respectively.

Method	News Categorization (LaMP-2N)		Movie Tagging (LaMP-2M)		Product Rating (LaMP-3)	
	Acc ↑	F1 ↑	Acc ↑	F1 ↑	MAE ↓	RMSE ↓
LoRA-C	0.787	0.538	0.478	0.425	0.223	0.491
LoRA-P	0.591	0.397	0.528	0.383	0.183	0.502
OPPU	0.810	**0.589**	0.600	0.493	**0.179**	**0.443**
Ours (PerFit)	**0.818**	0.586	**0.630**	**0.518**	**0.179**	**0.443**
- Param. Percentage ↓ (%)	0.0058	0.0117	0.0078	0.0010	0.0117	0.0015
- Param. Reduction ↑ (%)	93.75	81.25	91.67	98.44	87.50	97.66

Table 3: Results on generation tasks. We report ROUGE-1 (R-1) and ROUGE-L (R-L) metrics for LaMP-4, LaMP-5, and LaMP-7 tasks. The table compares both *Non-Tuned Methods* and *Tuned Methods* to demonstrate the effectiveness of different personalization approaches. For PerFit, we show the parameter percentage relative to the total model size and the parameter reduction compared to OPPU. Blue and red numbers represent Stage-1 and Stage-2 parameters respectively.

Method	News Headline Gen. (LaMP-4)		Scholarly Title Gen. (LaMP-5)		Tweet Paraphrasing (LaMP-7)	
	R-1 ↑	R-L ↑	R-1 ↑	R-L ↑	R-1 ↑	R-L ↑
Non-Tuned Methods						
Non-Personalized	0.030	0.029	0.145	0.118	0.126	0.123
PAG	0.098	0.082	0.149	0.121	0.135	0.124
RAG (k=1)	0.101	0.085	0.152	0.122	0.149	0.140
RAG (k=2)	0.106	0.088	0.167	0.132	0.136	0.130
RAG (k=4)	0.110	0.092	0.169	0.135	0.164	0.157
StyleVector	0.104	0.086	0.156	0.125	0.132	0.127
Tuned Methods						
LoRA-C	0.186	0.167	0.476	0.415	0.527	0.474
LoRA-P	0.120	0.108	0.489	0.435	0.398	0.333
OPPU	0.191	0.171	0.519	0.442	**0.539**	**0.483**
Ours (PerFit)	**0.207**	**0.186**	**0.521**	**0.451**	0.525	0.472
- Param. Percentage ↓ (%)	0.0117	0.0015	0.0039	0.0010	0.0078	0.0039
- Param. Reduction ↑ (%)	87.50	97.66	95.83	98.44	91.67	93.75

Personalized Classification Tasks. On classification tasks, PerFit achieves superior performance across all metrics. For LaMP-2N, our method attains the highest accuracy of 81.8%, surpassing OPPU by 0.8 percentage points. In LaMP-2M, PerFit achieves the best results with 63.0% accuracy and 51.8% F1 score, demonstrating substantial improvements over baselines. For LaMP-3, PerFit achieves comparable performance to OPPU with an MAE of 0.179 and RMSE of 0.443, while utilizing significantly fewer parameters.

Personalized Generation Tasks. In generation tasks, PerFit demonstrates consistent improvements over existing approaches. For LaMP-4, our method achieves the highest ROUGE-1 score of 20.7% and ROUGE-L score of 18.6%, outperforming both *Non-tuned* and *Tuned* baselines. On LaMP-5, PerFit achieves the best performance with a ROUGE-1 score of 52.1% and ROUGE-L score of 45.1%. While OPPU achieves marginally better performance on LaMP-7, our method maintains competitive results while utilizing significantly fewer parameters.

4.3 Efficiency Analysis (RQ2)

We conduct a detailed analysis of parameter efficiency based on the results in the main tables. As shown in Tables 2 and 3, our PerFit method consistently achieves state-of-the-art or highly competitive performance while dramatically reducing the number of trainable parameters. Specifically, in the first stage, PerFit requires only 0.0058% to 0.0117% of the total model parameters for classification tasks, and 0.0039% to 0.0117% for generation tasks. In the second stage, it uses an even

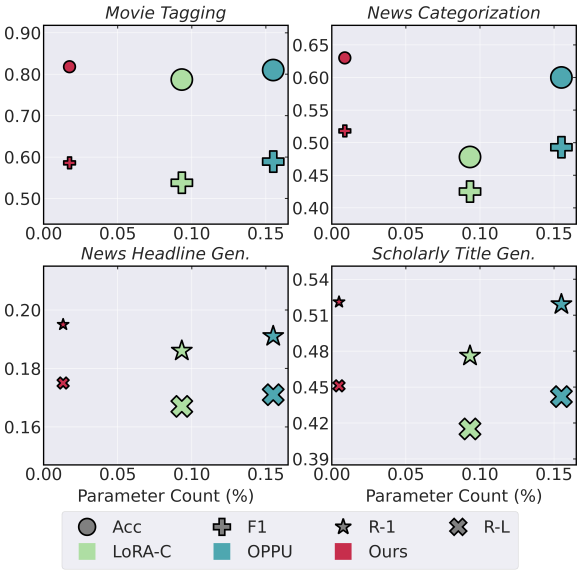


Figure 9: Performance versus parameter count on four datasets. Marker size reflects the relative training time[2].

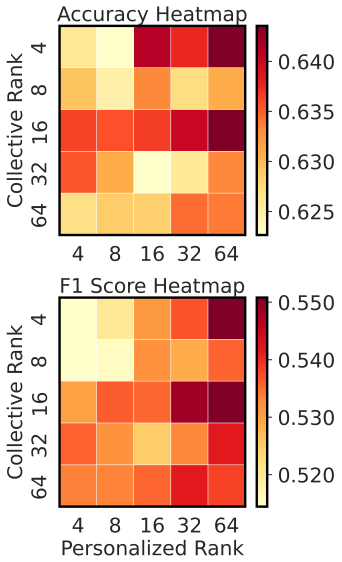


Figure 10: Impact of Collective and Personalized Rank on Movie Tagging Performance.

smaller proportion of 0.0010% to 0.0015% for classification tasks and 0.0010% to 0.0039% for generation tasks. This two-stage design achieves a remarkable parameter reduction of **81.25%** to **98.44%** compared to strong baselines such as OPPU. This substantial reduction highlights the efficiency of our approach in both memory and computational cost.

The accompanying Figure 9 provides a visual summary of these findings, plotting model performance against the proportion of trainable parameters for four representative datasets. Notably, `PerFit` not only reduces parameter count but also achieves a **17.0%** to **35.8%** reduction in training time compared to existing fine-tuning baselines. This demonstrates that our method delivers both parameter and runtime efficiency without sacrificing performance, making it a practical and scalable solution for personalized adaptation in LLMs.

4.4 Ablation Study (RQ3)

To validate our two-stage design and low-rank subspace intervention, we conduct an ablation study across diverse tasks (Table 4). Using only Stage-1 (collective shift learning) results in 2.6%-16.4% accuracy drops, confirming the importance of personalized adaptation in Stage-2. When training only Stage-2, both Ours@Stage-2 (C+P) and Ours@Stage-2 (P) configurations show limited performance without Stage-1's collective information. However, the higher rank configuration (C+P) still outperforms Ours@Stage-2 (P), demonstrating that increased rank helps capture more dimensions of user-specific information, though with diminishing returns. This aligns with Observation 1, suggesting that essential personalized information lies within a lower rank subspace.

4.5 Hyperparameter Analysis

Layer-wise Intervention. Figure 11 (left and middle) presents the results of intervening at a single layer for both *Movie Tagging* and *News Headline Gen.* tasks. We observe a clear trend: as the intervention layer moves from lower (earlier) to higher (later) layers, the overall performance—across all metrics—steadily declines. This finding is particularly intriguing when contrasted with prior work in knowledge editing (KE), where middle layers are typically used for learning and storing new knowledge (Meng et al., 2022). In our case, however, intervening at earlier layers yields better results. We hypothesize that this difference arises because, unlike KE tasks that typically require editing a small set of knowledge points, our method's first stage must absorb and encode a large amount of user-specific information. This process likely depends more heavily on modifications to

[2]Larger markers indicate longer training times. Note that these training times refer to the first stage of training and are provided for reference only, as they are influenced by various factors including dataset size and hardware specifications. The size primarily serve to illustrate the relative time relationships between different methods.

[3]@*Stage-2* degenerates into a one-stage model, equivalent to the ReFT model detailed in Appendix D.2

Table 4: Ablation results across diverse tasks, evaluating the impact of different training stages and configurations. Ours@Stage-2 (C+P) denotes the configuration where the rank is set to the sum of both stages' ranks, while Ours@Stage-2 (P) represents the configuration with only the Stage-2 rank[3]. *ref. LoRA-P* represents the reference values using LoRA-P.

Method	News Categorization		Movie Tagging		News Headline		Tweet Paraphrasing	
	Acc ↑	F1 ↑	Acc ↑	F1 ↑	R-1 ↑	R-L ↑	R-1 ↑	R-L ↑
Ours	0.818	0.586	0.630	0.518	0.195	0.175	0.525	0.472
@Stage-1	0.792	0.529	0.466	0.415	0.189	0.169	0.493	0.450
@Stage-2 (C+P)	0.803	0.604	0.620	0.496	0.194	0.175	0.483	0.438
@Stage-2 (P)	0.801	0.594	0.599	0.473	0.190	0.171	0.478	0.433
ref. LoRA-P	0.591	0.397	0.528	0.383	0.120	0.108	0.398	0.333

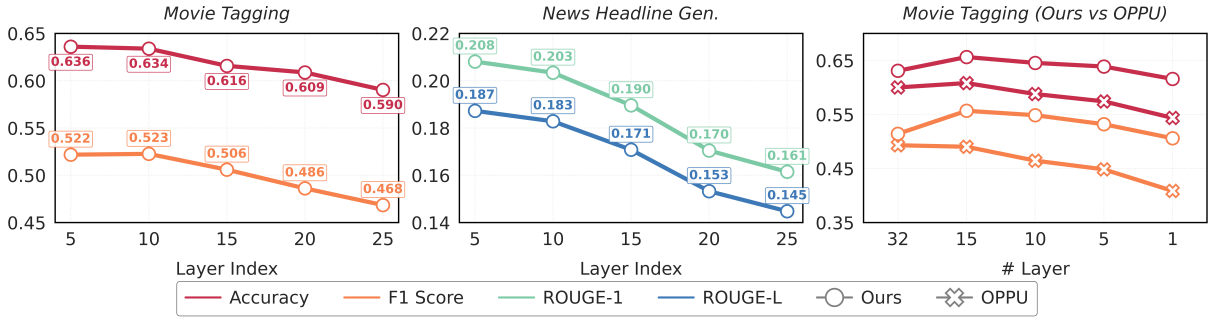


Figure 11: Layer-wise and cumulative intervention analysis. **Left & Middle:** Performance metrics (Acc, F1, R-1, R-L) versus single intervention layer position for Movie Tagging and News Headline Generation tasks. **Right:** Performance on Movie Tagging versus number of intervention layers[4].

lower-level model parameters, which are responsible for foundational feature extraction and representation.

Cumulative Intervention. As shown in the right panel of Figure 11, increasing the number of intervention layers generally leads to improved performance on the *Movie Tagging* task. This suggests that leveraging more layers allows the model to better capture and utilize personalized information. However, we observe that when interventions are applied to as many as all layers, performance unexpectedly drops. This indicates that editing too many layers may introduce negative side effects, possibly due to interference or redundancy among the interventions at different layers.

Collective vs. Personalized Rank. Figure 10 presents a heatmap analysis of the impact of collective (Stage-1) and personalized (Stage-2) rank on *Movie Tagging* performance, measured by both accuracy and F1 score. Overall, we observe that increasing either the collective rank or the personalized rank generally leads to improved performance. However, the effect of the personalized rank appears to be more pronounced: even when the collective rank is low, a sufficiently high personalized rank can achieve near-optimal results. This suggests that while both components contribute to model capacity, the personalized rank plays a more critical role in capturing user-specific information. These findings highlight the importance of allocating sufficient capacity to the personalized subspace, and indicate that effective personalization can be achieved even with a modest collective rank, provided the personalized rank is adequately set.

5 Conclusions and Future Work

By uncovering fundamental patterns in user-specific information—including shared collective and unique personalized shifts—our work introduces a novel two-stage method that fine-tunes interventions directly in the hidden representation space. Through extensive experiments across six diverse tasks, we demonstrate that this approach achieves efficient personalization with significantly reduced parameter overhead. This work paves the way for scalable, effective personalization in intelligent systems and reveals insights into user-specific information in LLMs. Future work could explore finer-grained personalization styles, such as community-level and group-level relationships.

[4]The layers are selected symmetrically around layer 15, with the spacing between layers determined as (# Model Layers / # Intervened Layer).

References

Andy Arditi, Oscar Balcells Obeso, Aaquib Syed, Daniel Paleka, Nina Rimsky, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Nora Belrose. 2024. Diff-in-means concept editing is worst-case optimal: Explaining a result by sam marks and max tegmark, 2023. *URL https://blog. eleuther. ai/diff-in-means*.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, and 1 others. 2024. A survey on evaluation of large language models. *ACM transactions on intelligent systems and technology*, 15(3):1–45.

Junyi Chen. 2023. A survey on large language models for personalized and explainable recommendations. *arXiv preprint arXiv:2311.12338*.

Ruizhe Chen, Xiaotian Zhang, Meng Luo, Wenhao Chai, and Zuozhu Liu. 2024. Pad: Personalized alignment of llms at decoding-time. *arXiv:2410.04070*.

Yae Jee Cho, Luyang Liu, Zheng Xu, Aldi Fahrezi, and Gauri Joshi. 2024. Heterogeneous lora for federated fine-tuning of on-device foundation models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12903–12913.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proc. of KDD*.

Chao Gao and Sai Qian Zhang. 2024. Dlora: Distributed parameter-efficient fine-tuning solution for large language model. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13703–13714.

Pengxin Guo, Shuang Zeng, Yanran Wang, Huijie Fan, Feifei Wang, and Liangqiong Qu. 2024. Selective aggregation for low-rank adaptation in federated learning. *arXiv preprint arXiv:2410.01463*.

Jerry Zhi-Yang He, Sashrika Pandey, Mariah L Schrum, and Anca Dragan. 2024. Cos: Enhancing personalization and mitigating bias with context steering. *arXiv:2405.01768*.

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

Minda Hu, Licheng Zong, Hongru Wang, Jingyan Zhou, Jingjing Li, Yichen Gao, Kam-Fai Wong, Yu Li, and Irwin King. 2024. SeRTS: Self-rewarding tree search for biomedical retrieval-augmented generation. In *Proc. of EMNLP Findings*.

Xiaoyu Kong, Jiancan Wu, An Zhang, Leheng Sheng, Hui Lin, Xiang Wang, and Xiangnan He. 2024. Customizing language models with instance-wise lora for sequential recommendation. In *Proc. of NeurIPS*.

Allison Lau, Younwoo Choi, Vahid Balazadeh, Keertana Chidambaram, Vasilis Syrgkanis, and Rahul G Krishnan. 2024. Personalized adaptation via in-context preference learning. *arXiv:2410.14001*.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36:41451–41530.

Jiahong Liu, Zexuan Qiu, Zhongyang Li, Quanyu Dai, Jieming Zhu, Minda Hu, Menglin Yang, and Irwin King. 2025. A survey of personalized large language models: Progress and future directions. *arXiv preprint arXiv:2502.11528*.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*.

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 *Proc. of WSDM*.

Aman Madaan, Niket Tandon, Peter Clark, and Yiming Yang. 2022. Memory-assisted prompt editing to improve gpt-3 after deployment. In *Proc. of EMNLP*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Forty-first International Conference on Machine Learning*.

Sriyash Poddar, Yanming Wan, Hamish Ivison, Abhishek Gupta, and Natasha Jaques. 2024. Personalizing reinforcement learning from human feedback with variational preference learning. *arXiv:2408.10075*.

Jiaxing Qi, Zhongzhi Luan, Shaohan Huang, Carol Fung, Hailong Yang, and Depei Qian. 2024. Fdlora: Personalized federated learning of large language model via dual lora tuning. *arXiv:2406.07925*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, and 1 others. Improving language understanding by generative pre-training.

Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. 2024. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. *Proc. of NeurIPS*.

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:2310.20081*.

Alireza Salemi, Surya Kallumadi, and Hamed Zamani. 2024a. Optimization methods for personalizing large language models through retrieval augmentation. In *Proc. of SIGIR*.

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

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2024b. LaMP: When Large Language Models Meet Personalization. *arXiv preprint*. ArXiv:2304.11406.

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.

Ruizhe Shi, Yifang Chen, Yushi Hu, Alisa Liu, Hannaneh Hajishirzi, Noah A Smith, and Simon S Du. 2024. Decoding-time language model alignment with multiple objectives. *arXiv:2406.18853*.

Gilbert W Stewart. 1993. On the early history of the singular value decomposition. *SIAM review*, 35(4):551–566.

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

Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. 2023. Linear representations of sentiment in large language models. *arXiv preprint arXiv:2310.15154*.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*.

Dimitri Von Rütte, Sotiris Anagnostidis, Gregor Bachmann, and Thomas Hofmann. 2024. A language model's guide through latent space. In *International Conference on Machine Learning*, pages 49655–49687. PMLR.

Nicolas Wagner, Dongyang Fan, and Martin Jaggi. 2024. Personalized collaborative fine-tuning for on-device large language models. *arXiv:2404.09753*.

Hongru Wang, Minda Hu, Yang Deng, Rui Wang, Fei Mi, Weichao Wang, Yasheng Wang, Wai-Chung Kwan, Irwin King, and Kam-Fai Wong. 2023. Large language models as source planner for personalized knowledge-grounded dialogue. *arXiv:2310.08840*.

Hongru Wang, Huimin Wang, Lingzhi Wang, Minda Hu, Rui Wang, Boyang Xue, Yongfeng Huang, and Kam-Fai Wong. 2024a. Tpe: Towards better compositional reasoning over cognitive tools via multi-persona collaboration. In *NLPCC*, pages 281–294. Springer.

Weixuan Wang, Jingyuan Yang, and Wei Peng. 2024b. Semantics-adaptive activation intervention for llms via dynamic steering vectors. *arXiv preprint arXiv:2410.12299*.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. 2024. ReFT: Representation Finetuning for Language Models. *arXiv preprint*. ArXiv:2404.03592 [cs].

Menglin Yang, Jialin Chen, Yifei Zhang, Jiahong Liu, Jiasheng Zhang, Qiyao Ma, Harshit Verma, Qianru Zhang, Min Zhou, Irwin King, and 1 others. 2024. Low-rank adaptation for foundation models: A comprehensive review. *arXiv preprint arXiv:2501.00365*.

Jinghao Zhang, Yuting Liu, Wenjie Wang, Qiang Liu, Shu Wu, Liang Wang, and Tat-Seng Chua. 2025. Personalized text generation with contrastive activation steering. *arXiv preprint arXiv:2503.05213*.

Kai Zhang, Lizhi Qing, Yangyang Kang, and Xiaozhong Liu. 2024a. Personalized llm response generation with parameterized memory injection. *arXiv:2404.03565*.

Qi Zhang, Yifei Wang, Jingyi Cui, Xiang Pan, Qi Lei, Stefanie Jegelka, and Yisen Wang. 2024b. Beyond interpretability: The gains of feature monosemanticity on model robustness. *arXiv preprint arXiv:2410.21331*.

Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2024c. A survey on the memory mechanism of large language model based agents. *arXiv:2404.13501*.

Zeyu Zhang, Quanyu Dai, Luyu Chen, Zeren Jiang, Rui Li, Jieming Zhu, Xu Chen, Yi Xie, Zhenhua Dong, and Ji-Rong Wen. 2024d. Memsim: A bayesian simulator for evaluating memory of llm-based personal assistants. *arXiv:2409.20163*.

Zhehao Zhang, Ryan A Rossi, Branislav Kveton, Yijia Shao, Diyi Yang, Hamed Zamani, Franck Dernoncourt, Joe Barrow, Tong Yu, Sungchul Kim, and 1 others. 2024e. Personalization of large language models: A survey. *arXiv preprint arXiv:2411.00027*.

Jiachen Zhu, Jianghao Lin, Xinyi Dai, Bo Chen, Rong Shan, Jieming Zhu, Ruiming Tang, Yong Yu, and Weinan Zhang. 2024. Lifelong personalized low-rank adaptation of large language models for recommendation. *arXiv:2408.03533*.

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

A Example Appendix

B Related Work

The methods for personalized large language models (PLLMs) can be mainly divided into two types based on whether fine-tuning is involved (Liu et al., 2025): one type is the method that does not require fine-tuning of the large language models (LLMs), and the other type is the method that requires fine-tuning.

Tune-Free Methods. Tune-free methods primarily use three approaches: input prompting, vector steering, and logits steering. (1) For input prompting, key approaches include profile-augmented generation (PAG) (Richardson et al., 2023), which uses an instruction-tuned language model to create a textual user profile from the user's personalized data, and retrieval-augmented generation (RAG) (Salemi et al., 2024a), which enhances responses by retrieving relevant entries from user history. (2) Vector steering, as implemented in StyleVector (Zhang et al., 2025), uses a separate LLM to generate contrastive pairs of personalized and non-personalized responses to modify model behavior. This method depends on pre-constructed contrastive pairs and doesn't tune model parameters; it has limited understanding of personalization. (3) Logits steering: CoS (He et al., 2024) achieves personalization by summing the logits from two rounds of outputs from the LLM: one round uses a standard prompt, while the other incorporates a user's explicit context in the prompt. Its main focus differs from our implicit personalization tasks.

Limitations: Although tune-free methods are efficient because they use external data sources, their personalization capabilities are limited since they rely on historical information instead of adapting the model's internal parameters, especially for capturing users' implicit tastes and style.

Fine-Tuning Methods. The one PEFT per user paradigm trains a Parameter-Efficient Fine-Tuning (PEFT) model tailored to each user using low-rank adaptation (LoRA)(Hu et al., 2021; Yang et al., 2024; Zhang et al., 2024a). OPPU(Tan et al., 2024b) encodes personalized user information in PEFT parameters, enhancing the overall user experience. While research following OPPU primarily focuses on framework enhancements, such as parameter collaboration in privacy-sensitive contexts (Qi et al., 2024; Wagner et al., 2024), the area of enhancing personalized fine-tuning remains underexplored.

Limitations. Despite the strong performance of the LoRA architecture, it still requires millions of parameters, which poses a significant burden in personalized scenarios with a large number of users.

Note that some methods for aligning human preferences in LLMs use reinforcement learning. While these approaches vary—some requiring fine-tuning (Rame et al., 2024; Lau et al., 2024; Poddar et al., 2024; Shi et al., 2024) and others not (Chen et al., 2024) — they mainly rely on reward models based on average annotator preferences. This approach requires predefined (explicit) preferences and fails to account for how different users might want different outputs for the same prompt, which differs from our task and objectives that propose a novel personalized fine-tuning method that captures users' implicit tastes and strikes a balance between effectiveness and efficiency.

C Preliminary

C.1 Problem Statement

Let $\mathcal{U} = \{u_i\}_{i=1}^{N}$ be a set of N users. Each user u_i is associated with a set of input queries $\mathcal{Q}_i = \{q_j^{(i)}\}_{j=1}^{n_i}$ and corresponding desired outputs $\mathcal{Y}_i = \{y_j^{(i)}\}_{j=1}^{n_i}$, which implies the user's personalized preferences and expectations. Here, n_i denotes the number of queries for user u_i. The base (i.e., non-personalized) LLM, denoted by M_0, generates generic outputs $\hat{y}_j^0 := M_0(q_j^{(i)})$ for any input query $q_j^i \in \mathcal{Q}_i$. Suppose Θ_0 denotes the base model parameters and Θ_i denotes the parameters for user u_i, and the personalized parameters increment as $\Delta\Theta_i := \Theta_i \setminus \Theta_0$.

Our objective is to adapt M_0 into personalized models M_i for each user u_i such that for every $q_j^{(i)}$, the personalized output $\hat{y}_j^{(i)} = M_i(q_j^{(i)})$ closely matches the desired output $y_j^{(i)}$ **while minimizing parameter overhead** $|\Delta\Theta_i|$. Formally, this can be expressed as minimizing the aggregate loss:

$$\min_{\{M_i\}_{i=1}^{N}} \sum_{i=1}^{N} \sum_{j=1}^{M_i} \mathcal{L}\big(M_i(q_j^{(i)}), y_j^{(i)}\big) \quad \text{s.t.}$$

where $\mathcal{L}(\cdot,\cdot)$ measures the discrepancy between model output and user target. This formulation encapsulates personalized fine-tuning of the base LLM to PLLM.

C.2 Hidden State Representations and Activation Steering

Hidden State Representation. Our work concentrates on decoder-only transformer architectures (Liu et al., 2018). For the base model M_0, each layer $\ell \in L$ comprises the multi-head attention and feed-forward modules MHA_0^ℓ and FFN_0^ℓ. Thus, the model can be expressed as: $M_0 = \bigcirc_{\ell \in L} \left(\mathrm{FFN}_0^\ell \circ \mathrm{MHA}_0^\ell\right)$, $\bigcirc$ denotes the composition of functions applied in sequence. The parameter set Θ_0 is partitioned accordingly: $\Theta_0 = \bigcup_{\ell \in L} \left(\Theta_0^{\mathrm{MHA}^\ell} \cup \Theta_0^{\mathrm{FFN}^\ell}\right)$, where $\bigcup$ denotes the union of sets. The layer ℓ of the base model M_0 updates the hidden state $\mathbf{h}_t^\ell \in \mathbb{R}^d$ of the token t as follows:

$$\mathbf{h}_t^{\ell+1} = \mathbf{h}_t^\ell + \mathrm{FFN}_0^\ell\big(\mathbf{h}_t^\ell + \mathrm{MHA}_0^\ell(\mathbf{h}_{1:t}^\ell)\big),$$

where MHA_0^ℓ attends causally over tokens 1 through t, d is the hidden dimension.

Activation Steering. Recent studies have explored how certain features are linearly represented in model hidden representation space utilizing activation steering (Tigges et al., 2023; Zhang et al., 2024b; Arditi et al.), such as harmlessness (Bolukbasi et al., 2016; Park et al.), truthfulness (Li et al., 2023), and humor (Von Rütte et al., 2024). These feature directions serve as effective causal mechanisms, enabling precise control over model behavior and outputs via simple linear interventions. Activation steering adds an intervention (i.e., vector) $\mathbf{v}^\ell \in \mathbb{R}^d$ to the hidden state at layer ℓ, modifying the model's behavior: $\tilde{\mathbf{h}}_t^\ell = \mathbf{h}_t^\ell + \mathbf{v}^\ell$. The next layer uses $\tilde{\mathbf{h}}_t^\ell$ instead of $\mathbf{h}_t^\ell$: $\mathbf{h}_t^{\ell+1} = \tilde{\mathbf{h}}_t^\ell + \mathrm{FFN}^\ell\big(\tilde{\mathbf{h}}_t^\ell + \mathrm{MHA}^\ell(\tilde{\mathbf{h}}_{1:t}^\ell)\big)$. This can be applied at any layer(s) to steer the model's output.

D Experiments

D.1 Datasets

Our experiments utilize the LaMP benchmark (Salemi et al., 2024b), a collection of personalization tasks from which we select six distinct tasks - three for classification and three for generation[5]. In alignment with the OPPU framework (Tan et al., 2024b), we recognize the importance of substantial user history for effective model personalization. Consequently, we identify and select about 100 most prolific users (those with the most extensive interaction histories) from the time-ordered LaMP variant to serve as our test cohort. The remaining users' data is allocated for training the collective LLM in the first stage. The dataset statistics are listed in Table 5.

D.2 Baselines

To provide a comprehensive comparison, we evaluate our method against a diverse set of baselines, categorized into **Non-Tuned** and **Tuned** approaches. All baseline models are implemented using Llama2-7B[6] as the foundation model.

Non-Tuned Methods

- **Non-Personalized**: This approach utilizes the pre-trained model without any modifications to generate responses for user queries. It establishes a performance floor for our experiments and serves as a reference point for measuring the effectiveness of personalization techniques.
- **Profile Augmented Generation (PAG)**: This method synthesizes a textual user profile using an instruction-tuned language model (e.g., Vicuna-13B[7]), derived from the user's interaction history. The generated profile is then prepended to each query to provide explicit contextual information about user preferences, enabling the model to generate more personalized responses without parameter updates.
- **Retrieval Augmented Generation (RAG)**: This technique implements the BM25 algorithm to retrieve the most relevant entries from a user's history (with $k = 1, 2, 4$) for each query. These retrieved entries serve as supplementary context for the model during inference, allowing it to access specific historical interactions that may be relevant to the current query.
- **StyleVector** (Zhang et al., 2025): This framework represents a training-free approach that disentangles and encodes personalized writing

[5]We omitted the LaMP-1 citation dataset because we were unable to reproduce results using the OPPU prompt, and for most queries, we could not get outputs in the required format. This may be due to limitations in Llama2's instruction following capabilities. Like OPPU, we did not use the LaMP-6 dataset due to privacy concerns. The remaining six datasets still ensure task diversity.

[6]Llama2-7B open-source model: https://huggingface.co/meta-llama/Llama-2-7b-hf

[7]Vicuna-13B open-source model: https://lmsys.org/blog/2023-03-30-vicuna/

Table 5: Dataset statistics for the LaMP benchmark. We present the average sequence length measured in token count, where #Q represents the quantity of queries, L_{in} and L_{out} denote the average input and output sequence lengths respectively, #History indicates the volume of historical interactions, and #Classes shows the number of classification categories for classification tasks. #Users shows the number of users in the base LLM training stage and personal PEFT training stage (format: first stage/second stage).

Task	#Users	Base LLM Training			Personal PEFT Training				
		#Q	L_{in}	L_{out}	#Q	#History	L_{in}	L_{out}	#Classes
2M	829 / 100	3,181	92.1	-	3,302	55.6	92.6	-	15
2N	274 / 49	3,662	68.2	-	6,033	219.9	63.5	-	15
3	1,543 / 100	22,388	128.7	-	112	959.8	211.9	-	5
4	19,899 / 101	7,275	33.9	9.2	6,275	270.1	25.2	11.1	-
5	14,581 / 101	16,075	162.1	9.7	107	442.9	171.6	10.3	-
7	13,337 / 100	14,826	29.7	18.3	109	121.2	29.4	18.0	-

style as a vector within the LLM's activation space. StyleVector enables style-controlled generation during inference without requiring retrieval mechanisms or parameter storage. The style vector is computed as the mean difference between positive and negative exemplars, and is injected into a specific token representation at a predetermined layer. Unlike our proposed method, StyleVector depends on carefully selected sample pairs, making it particularly sensitive to data quality and quantity, and necessitates more sophisticated vector engineering.

Tuned Methods

- **Personalized LoRA (LoRA-P)** (Hu et al., 2021): This standard parameter-efficient fine-tuning (PEFT) methodology creates individual models that are fine-tuned on each user's historical data. The approach produces user-specific parameter adaptations that capture individual preferences and behaviors through low-rank matrix decompositions of weight updates.

- **Collective LoRA (LoRA-C)** (Hu et al., 2021): This method employs LoRA fine-tuning on the collective history of all users excluding the 100 test users. The approach quantifies the benefits of collaborative training without personalization and provides a model that captures general user behaviors rather than individual preferences.

- **OPPU** (Tan et al., 2024b): This technique integrates a two-stage approach combining collaborative and personalized fine-tuning. The first stage trains on collective user data (similar to LoRA-C), while the second stage adapts these parameters to individual users (similar to LoRA-P). This dual-stage process allows the model to benefit from both collective knowledge and individual customization.

E Broader Impacts

Personalized Large Language Models (LLMs), particularly through methods like fine-tuning in representation space, offer transformative potential for human-computer interaction and information access. This approach, by subtly adapting LLMs via their underlying representation space rather than full model retraining, significantly enhances resource efficiency and scalability, making deep personalization feasible for a broader range of applications and users. This personalized tailoring promises to revolutionize user experience by matching communication style, vocabulary, and level of detail to individual needs, improving efficiency in tasks, and fostering hyper-personalized learning. Such adaptation inherently boosts accessibility, bridging communication gaps for diverse users.