Personalize Your LLM: Fake it then Align it

Yijing Zhang[†] Dyah Adila[†] Changho Shin[†] Frederic Sala[†]

[†]University of Wisconsin-Madison
{yzhang2637, adila, cshin23, fredsala}@wisc.edu

November 28, 2024

Abstract

Personalizing large language models (LLMs) is essential for delivering tailored interactions that improve user experience. Many existing personalization methods require fine-tuning LLMs for each user, rendering them prohibitively expensive for widespread adoption. Although retrieval-based approaches offer a more compute-efficient alternative, they still depend on large, high-quality datasets that are not consistently available for all users. To address this challenge, we propose **CHAMELEON**, a scalable and efficient personalization approach that uses (1) self-generated personal preference data and (2) representation editing to enable quick and cost-effective personalization. Our experiments on various tasks, including those from the LaMP personalization benchmark, show that CHAMELEON efficiently adapts models to personal preferences, improving instruction-tuned models and outperforms two personalization baselines by an average of 40% across two model architectures.

1 Introduction

Large language models (LLMs) have transformed natural language processing (NLP), achieving excellent performance across a wide range of tasks. Their use has already expanded into diverse domains and user bases [17, 39, 46, 47]. This has motivated the need for personalization, i.e. tailoring these models to individual user preferences and specific contexts [23].

Current personalization methods are often impractical for large-scale deployment. Fine-tuning approaches [9, 26, 40] are resource-intensive, making it prohibitively expensive to customize models for each individual user. In contrast, retrieval-based methods [12, 14, 37] offer greater computational efficiency but suffer from a significant drawback: they rely on large high-quality datasets that are not consistently available for all users. These limitations impede the effective scaling of personalization, especially given the diverse and rapidly evolving nature of user preferences.

To achieve scalable personalization, we argue that two essential conditions must be met: (1) data efficiency, which enables effective personalization with minimal user interaction, and (2) compute efficiency, allowing for deployment across a large user base. We propose CHAMELEON, a new approach that fulfills both requirements by using synthetic, self-generated data to capture user preferences and uses representation editing to tailor its behavior to each user's unique preferences [2].

For each user, we begin with a small amount of historical data—sometimes as little as a single sample. Using this data, we prompt the LLM to generate two characteristic descriptions: one that reflects the user's personal preferences based on their history and another that represents a contrasting or non-personalized profile (e.g., "funny" versus "formal"). From these descriptions, we create synthetic user preference data. We then identify two distinct embedding spaces—personalized and non-personalized—derived from the synthetic preference data. Finally, we edit the LLM's embeddings to enhance the influence of the personalized subspace while diminishing the influence of the non-personalized subspace.

With this data- and compute-efficient approach, we improve instruction-tuned models and two LLM personalization baselines by an average of 40% in the LaMP personalization benchmark [37]. In summary, our contributions are:

1. We introduce CHAMELEON, an LLM personalization framework that leverages self-generated user preference data and embedding editing techniques, providing **scalable, user-tailored personalization that is nearly cost-free**.
2. On extensive evaluation using the LaMP benchmark [37], we show that CHAMELEON improves upon instruction-tuned models and two LLM personalization benchmarks by an average of 40% on two model architectures.

3. CHAMELEON can effectively personalize for new, unseen users without user history by leveraging profiles from other users with similar characteristics and preferences.

2 Related Work

Our work seeks to address the personalization problem for LLMs using representation editing as an efficient technique to align models with user preferences. We give a brief overview of related areas.

Personalized LLMs. Unlike general LLMs that produce uniform responses for all users, personalized LLMs adapt to the specific linguistic and communication preferences of individual users [9]. Fine-tuning is a common method for achieving this, by training models on user-specific or task-specific data to personalize their behavior [44]. Approaches like P-RLHF [26], Persona-Plug [28], and ALOE [45] exemplify this strategy. However, fine-tuning is resource-intensive, making it impractical to personalize models for individual users at scale. Parameter-efficient fine-tuning (PEFT) [40] reduces the computational burden but still requires large amounts of user data, which is often scarce and difficult to obtain in user personalization task [51].

Retrieval-based methods personalize model outputs by incorporating user-specific information retrieved at inference time [10, 22, 29, 37, 41, 49]. While these methods avoid the need for tuning, they struggle with LLMs' limited context lengths, especially when dealing with long user histories. Although long-context models [13, 30, 34] allow for processing larger user histories, this incurs a high cost as many models are charged per token. Attempts to address this issue by summarizing retrieved information have been made [31, 36]. However, these approaches are vulnerable to distractions from irrelevant information [38], particularly when user behavior or preferences shift [6, 15].

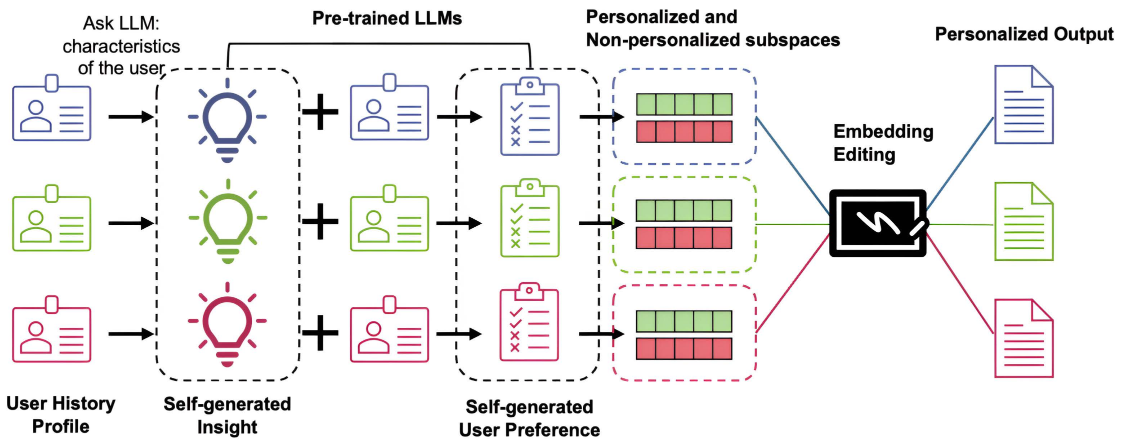
The closest work to ours is LLM-REC [32], a prompt-based approach that personalizes LLMs using summaries of selected top user history data. Our method takes this a step further by generating self-preference data, identifying embedding spaces that capture personalized versus non-personalized preferences, and performing personalization through representation editing. This enables a more data- and compute-efficient personalization process, making it possible to adapt models at scale to evolving user preferences quickly. Our approach represents a significant step toward scalable, real-time personalization that caters to dynamic user preference data.

Representation Editing for Personalization. Representation editing has become an important technique for model alignment, involving the direct manipulation of a model's latent representations to improve its performance and align it with desired attributes [24, 42]. For example, Han et al. [19] demonstrated that steering LLM text embeddings can guide model output *styles*. Similarly, [18, 25] show that adjusting embeddings during inference can enhance specific attributes, such as honesty or truthfulness, in the generated outputs. Liang et al. [27] found that representation editing can control aspects of text generation, such as safety, sentiment, thematic consistency, and *linguistic style*. These findings highlight the potential of using representation editing to guide models for personalization tasks. For visual generation models like Stable Diffusion, embedding-based personalization has long been recognized as an established technique [3, 4, 20, 48].

Despite the growing interest in representation editing, little research has explored its application for personalizing LLMs, as proposed in our work. The most closely related study is Adila et al. [2], where the authors use embedding editing for general, rather than personalized, alignment to broad human preferences, relying on self-generated synthetic data. Our approach advances this notion by introducing a tailored mechanism that generates personalized synthetic data for each user and adapts embedding editing techniques for both individual and group-based personalization.

3 CHAMELEON: Personalization through Representation Editing

We present CHAMELEON, an almost **cost-free** alignment personalization framework with representation editing using self-generated synthetic user preference data. Figure 1 illustrates our technique. We achieve personalization with two stages: (1) self-generating user preference data (Section 3.1), and (2) representation editing using the self-generated data (Section 3.2). Additionally, we extend CHAMELEON to support **scalable user groups**, enabling efficient alignment at a group level (Section 3.3).



Ask LLM: characteristics of the user
Pre-trained LLMs
Personalized and Non-personalized subspaces
Personalized Output
Embedding Editing
User History Profile
Self-generated Insight
Self-generated User Preference

Figure 1: CHAMELEON identifies two separate subspaces, one personalized and one non-personalized, from self-generated user characteristic insights. Based on these subspaces, we modify the LLM embeddings during inference.

3.1 Self-generated Preference Data

Our method for generating self-preference data uses generic, non-personalized LLMs to identify user-specific characteristics and preferences from the available user history. Using these identified characteristics, we prompt the model to generate tailored responses for each user. This process consists of three key steps: (1) selecting relevant user history, (2) generating insights from the selected history, and (3) producing synthetic user preference data guided by these insights.

User History Selection. User's historical behavior usually contains important information regarding their characteristics, linguistic patterns, and preferred interactions. However, not all historical behaviors serve as reliable indicators of user preferences. Adapting the model using redundant and generic user behavior may not result in high-quality personalized LLMs. Selecting and filtering for representative user historical behavior is thus important. Although recent studies showed success in using retrieval-based re-rankers [50] and encoder-based user history selection [28], they can struggle when user preferences shift rapidly or when there's limited historical data. To address this, we focus on a more lightweight and adaptable approach to user history selection.

Since our approach relies on embedding editing to adapt the model, we need to identify user-representative historical data. The first step is to define what makes this data "representative." We leverage sentence embeddings for their strong ability to capture both the meaning and context of text [35]. Our goal is to find the most informative and relevant k embedding pieces that reflect key user preferences. A lightweight approach to find such data is to perform principal component analysis (PCA) on the embeddings [16]. CHAMELEON deploys a PCA-based approach in the user history selection process, where we pick top k user histories from the provided user history profile. Specifically, for each user u, given a set of user history $\mathcal{H}_u = \{h_u^i\}$ where each h_u^i represents an individual user history sample with index i, we have

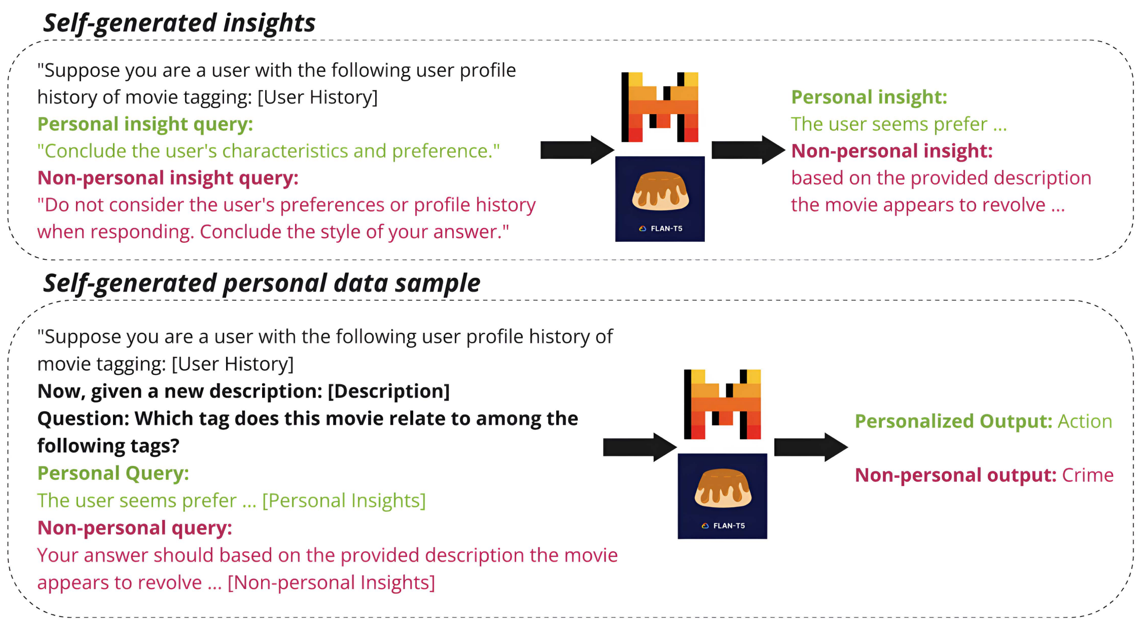
$$e_u^i = \textbf{SentenceEmbedder}(h_u^i). \tag{1}$$

Then, we have that $\boldsymbol{W}_u$ are the top k principal components of $\boldsymbol{E_u} = [e_u^1, e_u^2, \ldots, e_u^N]^\top$ and the projection of each embedding is $z_u^i = e_u^i \boldsymbol{W}_u$. We next find the top k history data embeddings:

$$\boldsymbol{E}_u^k = \underset{i \in [1,\ldots,N]}{\arg\,\text{top-}k} \left|\left|z_u^i\right|\right|, \tag{2}$$

and get top k history data $\boldsymbol{H}_u^k = \{h_u^i : i \in \boldsymbol{E}_u^k\}$.

Insight Generation. We query an instruction-tuned general-purpose LM to analyze and infer characteristics specific to individual users. For each user u, given the selected set of user history $\boldsymbol{H}_u^k$ from the previous step, we query the LM (denoted as ω) and generate two distinct styles of responses: one as a personalized agent (C^P) and the other as a non-personalized/neutral agent (C^N). The personalized agent (C^P) draws on the user's historical data $\boldsymbol{H}_u^k$, concluding insights about the user's preferences, behaviors, and style. The neutral agent (C^N) is asked to give characteristics of impersonal and general responses. It represents the standard behavior of the model when user personalization is absent. Then, for each user u, we have an personalized-neutral insights pair (c_u^P, c_u^N).



"Suppose you are a user with the following user profile history of movie tagging: [User History]

Personal insight query:
"Conclude the user's characteristics and preference."

Non-personal insight query:
"Do not consider the user's preferences or profile history when responding. Conclude the style of your answer."

Personal insight:
The user seems prefer ...

Non-personal insight:
based on the provided description the movie appears to revolve ...

"Suppose you are a user with the following user profile history of movie tagging: [User History]

Now, given a new description: [Description]
Question: Which tag does this movie relate to among the following tags?

Personal Query:
The user seems prefer ... [Personal Insights]

Non-personal query:
Your answer should based on the provided description the movie appears to revolve ... [Non-personal Insights]

Personalized Output: Action

Non-personal output: Crime

Figure 2: Self-generated user preference data: we use the generated conclusion of user characteristics to guide the personal answer generation.

Generating Synthetic User Preference Data Once the insights are generated, we use the insight pairs as prompt guidance to generate synthetic user preference data. For each user u and each user query q_u, given the pre-selected history set $\mathcal{H}_u$ and insight pair $(c_u^{i,P}, c_u^{i,N})$, we have our general-purpose LM (ω) separately generate personalized and neutral preference outputs $(\hat{y}_u^{i,P}, \hat{y}_u^{i,N})$ to query q_u^i conditioned on $(c_u^{i,P}, c_u^{i,N})$ respectively. We then concatenate the outputs $(\hat{y}_u^{i,P}, \hat{y}_u^{i,N})$ with user history $\mathcal{H}_u$ and obtain the self-generated preference pair $(p_u^{i,P}, p_u^{i,N})$ for each user query q_u^i. By applying this procedure to all user queries, we obtain self-generated preference data pairs (P_u^P, P_u^N).

Note that we do not apply any prompt tuning; rather, we use a predefined set of prompt templates and a frozen LLM for all generations. Figure 2 illustrates the full process, with prompting details in Appendix A.3.

3.2 Representation Editing

Next, using the self-generated user preference data, we align the model with users' preferences with a technique inspired by ALIGNEZ [2]. We first identify two subspaces in the model's embedding space (denoted as vector $\theta \in \mathbb{R}^d$ in LM ω's latent space) that correspond with the users' preferences. We use singular value decomposition (SVD) on the preference data embeddings to capture directions of the personalized embeddings $\theta_{l,u}^P$. Next, we employ CCS-based identification [5] to find the hyperplane that best separates the non-personalized embeddings from the personalized ones and denote the directions of the hyperplane as $\theta_{l,u}^N$, where the SVD and CCS-based unsupervised methods help avoid overfitting to the noise inherent in synthetic data. A detailed explanation is provided in Appendix A.4.

With the personalized and non-personalized subspaces θ^P and θ^N, we perform embedding editing on the MLP outputs of the most impactful decoder layers (i.e. layers that have lowest average CSS loss) during the inference phase to adapt the LLM to users' preferences. More concretely, given x_l, the output of the MLP of layer $l \in L$, where L is the set of layers with lowest average CSS loss, we strengthen the personalized direction by

$$\hat{x}_{l,u} \leftarrow x_l + \frac{\langle x_l, \theta_{l,u}^P \rangle}{\langle \theta_{l,u}^P, \theta_{l,u}^P \rangle} \theta_{l,u}^P$$

and remove the non-personalized direction by

$$\hat{x}_{l,u} \leftarrow \hat{x}_{l,u} - \frac{\langle \hat{x}_{l,u}, \theta_{l,u}^N \rangle}{\langle \theta_{l,u}^N, \theta_{l,u}^N \rangle} \theta_{l,u}^N.$$

These edits are performed for each user query.

Models →		Mistral Instruct				Flan T5 XXL			
Dataset	Metric	Instruct Model	LLM -REC	ALOE	CHAMELEON	Instruct Model	LLM -REC	ALOE	CHAMELEON
LaMP2	Acc. ↑	0.198	0.262	0.307	**0.396**	0.238	0.214	0.333	**0.420**
	F-1 ↑	0.236	0.309	0.220	**0.349**	0.171	0.146	0.255	**0.311**
LaMP3	MAE ↓	0.497	0.484	0.423	**0.407**	0.456	0.798	0.427	**0.400**
	RMSE ↓	0.944	0.976	0.888	**0.815**	0.818	1.439	0.786	**0.714**
LaMP7	R-1 ↑	0.354	0.183	0.362	**0.381**	0.333	0.225	0.376	**0.429**
	R-L ↑	0.295	0.144	0.313	**0.334**	0.292	0.196	0.331	**0.385**

Table 1: CHAMELEON outperforms all baselines in personalization for users with history. Best performance is highlighted in **bold**. Metrics where higher values indicate better performance are shaded in blue cells, while metrics where lower values are preferable are marked with green cells.

Models →		Mistral Instruct		Flan T5 XXL	
Dataset	Metric	ALOE	CHAMELEON	ALOE	CHAMELEON
LaMP2	Acc. ↑	0.227	**0.363**	0.109	**0.390**
	F-1 ↑	0.177	**0.338**	0.040	**0.304**
LaMP3	MAE ↓	0.522	**0.442**	0.544	**0.413**
	RMSE ↓	0.906	**0.903**	1.030	**0.839**
LaMP7	R-1 ↑	0.185	**0.377**	0.251	**0.420**
	R-L ↑	0.155	**0.331**	0.206	**0.373**

Table 2: CHAMELEON performance compared ALOE on new unseen users.

3.3 Group-scale Personalization

Individually aligning the model for multiple users is inefficient when scaling to a large user base [11]. To overcome this, we extend CHAMELEON to group-scale alignment. Instead of aligning for each user separately, we combine the history data of all users into a single group and perform collective alignment. Specifically, we aggregate the synthetic self-preference data for all users into one set, $(P^P, P^N) = \{(p_u^{i,P}, p_u^{i,N}) \in (P_u^P, P_u^N) | u \in U\}$, where U is the set of users in the group. (P^P, P^N) is then used to find direction vectors for representation editing.

This approach enables efficient personalization by processing all users simultaneously, leading to faster alignment. In Section 4.4, we show that group-scale personalization outperforms the single-user setting. Furthermore, this method allows us to leverage data from other users for those with no available history, enabling personalization for new or unseen users (see Experiment 4.2).

4 Experiments

We begin by detailing our experimental setup in Section 4.1, followed by experiments to validate the following key claims about CHAMELEON:

- Aligns LLMs to user-specific preferences (Section 4.2),
- Generalizes to unseen users (Section 4.3),
- Group-scale personalization improves performance (Section 4.4),
- Outperforms compute extensive methods like DPO in time-constrained scenarios (Section 4.5).
- Editing both personalized and non-personalized embedding spaces improves performance and outperforms best performance (Section 4.6).

In Section 4.7, we perform ablation study to understand the effect of the number of user history data to CHAMELEON performance.

4.1 Experimental Setup

Datasets and Tasks. We evaluate CHAMELEON using the LaMP language model personalization benchmark, which collects different personalization tasks and datasets into one benchmark [37]. Our evaluation focuses on three specific personalization tasks: (1) Personalized Movie Tagging (LaMP 2), (2) Personalized Product Rating (LaMP 3), and (3) Personalized Tweet Paraphrasing (LaMP 7). We adhered to the user-based data split provided by the LaMP benchmark, using the default training and test splits. Additional details about the datasets and tasks can be found in Appendix A.2.

Evaluation Metrics. We use the evaluation metrics established by the LaMP benchmark for each task. For Personalized Movie Tagging (LaMP 2), we measure Accuracy (Acc.) and F-1 Score (F-1). For Personalized Product Rating (LaMP 3), we assess performance using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). For Personalized Tweet Paraphrasing (LaMP 7), we apply the ROUGE-1 (R-1) and ROUGE-L (R-L) metrics.

Baseline 1: Non-personalized Instruction-tuned Models. We evaluate CHAMELEON against two general purpose instruction-tuned models: Mistral-7B-v0.3-Instruct [21] and Flan-T5 XXL [8]. Both models are assessed using the same set of user queries as CHAMELEON, following the same prompt format and using the same pre-selected user history profile—excluding any insights. Additional prompt details can be found in Appendix A.3.

Baseline 2: Personalization Methods. We also compare CHAMELEON against two personalization techniques, namely LLM-REC [32], a prompting-engineering personalization method, and ALOE [45], a supervised Fine-tuning (SFT) personalization method.

Group Personalization Setup. To implement group-scale personalization (Section 3.3), we randomly select 100 users from the training split of the LaMP benchmark. Using PCA-based history selection (Section 3.1), we choose up to 10 user history entries per profile. For each user, we generate personalized and neutral insight pairs along with self-generated preference data. Any data where the personalized and non-personalized outputs are identical is discarded. We then combine the self-generated preference data for all users, perform group-scale alignment, and evaluate the personalized model on unseen user queries from the LaMP test split (Section 4.3). This process is repeated for different random sets of 100 users, and we report the average performance.

4.2 Aligns LLMs to user-specific preferences

Setup. We compare CHAMELEON with the previously mentioned baselines. In the self-insight generation process, user history data is fed directly to the models using simple prompts (see Appendix A.3), without access to human annotations.

Results. As shown in Table 1, CHAMELEON consistently outperforms all baselines. Remarkably, these improvements are achieved with minimal user history data and without any training and fine-tuning, surpassing an SFT-based method (ALOE). **These results validate our claim that CHAMELEON can effectively align LLMs to individual user preferences**.

4.3 Generalizes to unseen users

Setup. We also assess CHAMELEON's ability to personalize for new, unseen users who have no prior history. In this evaluation, we run both CHAMELEON and ALOE on the LaMP training split and evaluate their performance on test samples from users not included in the training data. This experimental setup is not applicable to instruct models and LLM-REC, as both of these methods use prompt-based personalization and do not differentiate between seen and unseen users.

Results. Table 2 demonstrates that CHAMELEON achieves strong personalization performance even with new, unseen users, **validating our claim that CHAMELEON can effectively generalize to users without prior history**. In contrast, ALOE struggles in this scenario, suggesting that it may overfit to the characteristics of users in the training set.

4.4 Group-scale personalization improves performance

Setup. To assess the effectiveness of group-scale personalization compared to single-user personalization, we run CHAMELEON on groups of varying sizes. We experiment with group sizes of $\{1, 20, 40, 60, 80, 100\}$ on both LaMP2 and LaMP3 tasks, while keeping the amount of generated insights and preference data per user constant.

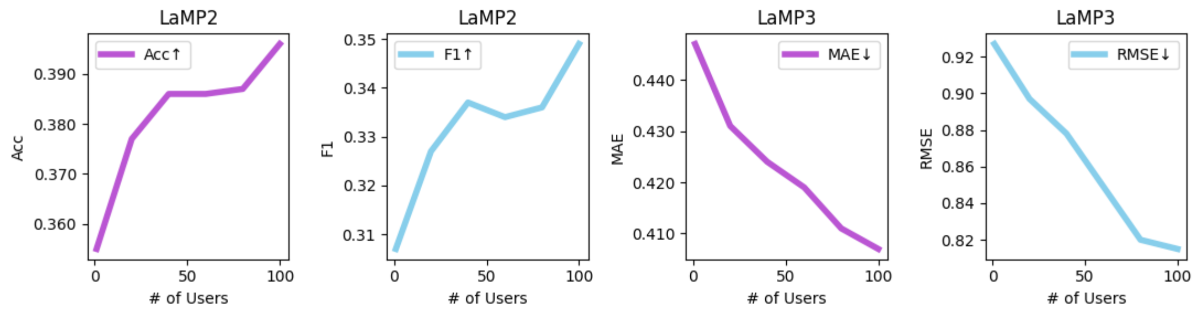


Figure 3: The change of performance when different number of users are given to CHAMELEON

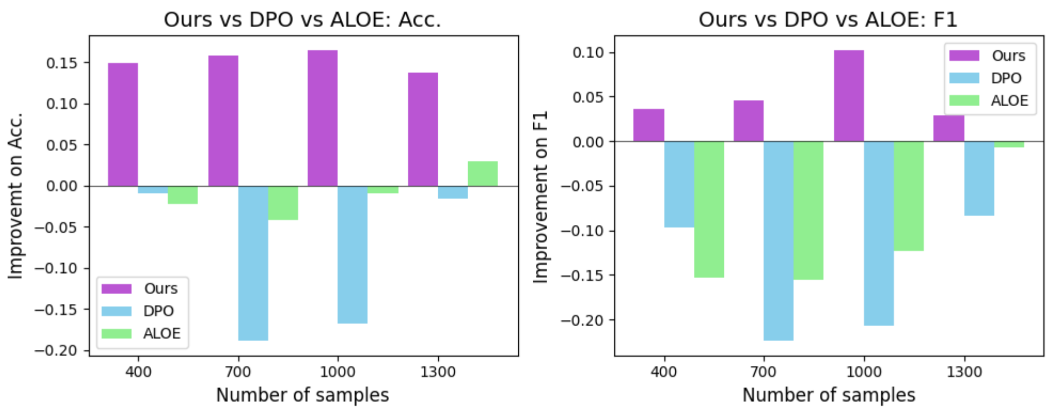


Figure 4: CHAMELEON compared with DPO and ALOE in time-constrained scenarios. The columns denotes the improvement from the instruction-tuned model.

Results. Figure 3 reveals a clear trend: as the number of users in the group increases, personalization performance consistently improves. This is evident both when shifting from a single-user setup (left-most point, where number of users = 1) to group personalization, and as the group size grows. **These results support our claim that group personalization offers performance gain compared to single-user personalization**.

4.5 Outperforms DPO in time-constrained scenario

Setup. We compare CHAMELEON with DPO [33] and ALOE [45], a tuning-based alignment and SFT-based personalization methods, in a time-constrained scenario where alignment must be performed quickly. In this setup, we fix the time allowed for all methods and get the number of samples for each method within that time. This setup reflects real-world situations where instant personalization is required for new users with little to no available data. Hyperparameter details for DPO and ALOE are provided in Appendix A.5.

Results As shown in Figure 4, CHAMELEON consistently delivers stable personalization gains in the time-constrained scenario, whereas both ALOE and DPO struggle with limited sample availability. This supports our claim that **CHAMELEON is more suitable than resource-intensive approaches in time-sensitive scenarios**.

4.6 Editing both personalized and non-personalized embedding improves performance.

Setup. To examine the individual effects of personalized and non-personalized profile, we conducted an experiment on only editing personalized/non-personalized embedding space on the LaMP2 and LaMP3 tasks on Mistral instruct models.

Results. We report the metric for each case in Table 3. CHAMELEON rely on editing in personalized embedding space to give personalized outputs, and removing non-personalized embedding space follows previous studies that removing spurious or unwanted concept subspaces from embeddings boosts model accuracy on rare class predictions [1, 7].

Models →		Mistral Instruct		
Dataset	Metric	Only Personalized	Only Non-personalized	Both
LaMP2	Acc. ↑	0.356	0.346	**0.396**
	F-1 ↑	0.276	0.268	**0.349**
LaMP3	MAE ↓	0.484	0.494	**0.407**
	RMSE ↓	0.900	1.005	**0.815**

Table 3: Embeddings to edit effect to performance of CHAMELEON.

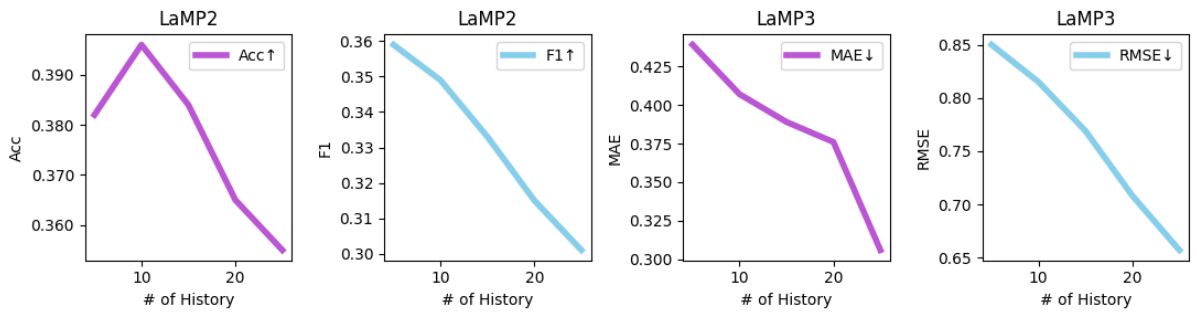


Figure 5: The change of performance when different number of history data per user are given to CHAMELEON

4.7 Ablations

Setup. To examine the impact of the amount of user history data on performance, we run CHAMELEON on the LaMP2 task, varying the number of history per user as $\{5, 10, 15, 20, 25\}$, while keeping the number of users in the group constant.

Results. Figure 5 illustrates that when the amount of user history data is small, the performance improvement of CHAMELEON is limited. This limitation likely arises from the difficulty in generating accurate personalization insights with insufficient data. Conversely, when the amount of history data is too big, the performance of CHAMELEON declines. We hypothesize that this deterioration occurs because too many history profiles may introduce unrelated or outdated samples, hindering effective personalization.

5 Discussion

Limitations. While CHAMELEON successfully delivers scalable personalization with minimal costs, it has some limitations. A key challenge is its dependence on the quality of the self-generated preference data. Although aligning the model with this data yields promising results, the effectiveness of the personalization largely depends on how accurately and comprehensively user preferences are captured by the base LLM. Future research could focus on developing more refined metrics to capture personal characteristics better, ensuring more precise and reliable self-alignment.

One potential risk with CHAMELEON is the possibility of malicious input in user history. Since CHAMELEON relies on a limited amount of user history to generate self-preference data for alignment, harmful or biased history inputs could unintentionally lead the model to produce toxic or malicious responses. This highlights the need for strong safeguards, such as thorough filtering and ethical review processes, to prevent the model from aligning with or reinforcing negative behaviors while still delivering effective personalization.

Ethical Considerations. Privacy has long been a problem for LLM personalization, as personalizing LLMs usually require large-scale personal data and preferredly (human) labeled, which could lead to potential privacy leaks. Though personalization dataset, like LaMP benchmark dataset used in our experiments, is publicly accessible an does not raise privacy concerns, personal data collection and usage still presents significant challenge in personalizing LLMs. With our approach, we only acquire a very small portion of user historical data and resolve data labeling problem with self-generation technique. And since self-generated user preference data are fake synthetic data for performing alignment, it can possibly reduce the risk of privacy leaks.

Conclusion. We present CHAMELEON, a novel light-weight, scalable approach for personalizing LLMs without access to large-scale human-annotated personal data and individual fine-tuning. By leveraging the ability to conclude and capture user characteristics and preferences, CHAMELEON adjusts the model embeddings during inference to generate outputs that are more aligned with user preferences. Our experiments show that CHAMELEON significantly enhance the personalization ability of base language models using only a small portion of real user data, and it is able to adapt models with multiple user expectations within one single alignment process.

This work represents an initial step toward achieving cost-free, rapid, group-scale personalization that current personalization methods struggle to address.

References

[1] Dyah Adila, Changho Shin, Linrong Cai, and Frederic Sala. Zero-shot robustification of zero-shot models, 2024. URL `https://arxiv.org/abs/2309.04344`.

[2] Dyah Adila, Changho Shin, Yijing Zhang, and Frederic Sala. Is free self-alignment possible?, 2024. URL `https://arxiv.org/abs/2406.03642`.

[3] Yuval Alaluf, Elad Richardson, Gal Metzer, and Daniel Cohen-Or. A neural space-time representation for text-to-image personalization. *ACM Trans. Graph.*, 42(6), December 2023. ISSN 0730-0301. doi: 10.1145/3618322. URL `https://doi.org/10.1145/3618322`.

[4] Moab Arar, Andrey Voynov, Amir Hertz, Omri Avrahami, Shlomi Fruchter, Yael Pritch, Daniel Cohen-Or, and Ariel Shamir. Palp: Prompt aligned personalization of text-to-image models, 2024. URL `https://arxiv.org/abs/2401.06105`.

[5] Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering latent knowledge in language models without supervision. In *The Eleventh International Conference on Learning Representations*, 2023.

[6] Micah Carroll, Davis Foote, Anand Siththaranjan, Stuart Russell, and Anca Dragan. Ai alignment with changing and influenceable reward functions, 2024. URL `https://arxiv.org/abs/2405.17713`.

[7] Ching-Yao Chuang, Varun Jampani, Yuanzhen Li, Antonio Torralba, and Stefanie Jegelka. Debiasing vision-language models via biased prompts, 2023. URL `https://arxiv.org/abs/2302.00070`.

[8] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022. URL `https://arxiv.org/abs/2210.11416`.

[9] Christopher Clarke, Yuzhao Heng, Lingjia Tang, and Jason Mars. Peft-u: Parameter-efficient fine-tuning for user personalization, 2024. URL `https://arxiv.org/abs/2407.18078`.

[10] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1126–1132, 2023.

[11] Zhenlong Dai, Chang Yao, WenKang Han, Ying Yuan, Zhipeng Gao, and Jingyuan Chen. Mpcoder: Multi-user personalized code generator with explicit and implicit style representation learning, 2024. URL `https://arxiv.org/abs/2406.17255`.

[12] Dario Di Palma. Retrieval-augmented recommender system: Enhancing recommender systems with large language models. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1369–1373, 2023.

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

[14] Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501, 2024.

[15] Matija Franklin, Hal Ashton, Rebecca Gorman, and Stuart Armstrong. Recognising the importance of preference change: A call for a coordinated multidisciplinary research effort in the age of ai. *arXiv preprint arXiv:2203.10525*, 2022.

[16] Felipe L. Gewers, Gustavo R. Ferreira, Henrique F. De Arruda, Filipi N. Silva, Cesar H. Comin, Diego R. Amancio, and Luciano Da F. Costa. Principal component analysis: A natural approach to data exploration. *ACM Computing Surveys*, 54(4):1–34, May 2021. ISSN 1557-7341. doi: 10.1145/3447755. URL `http://dx.doi.org/10.1145/3447755`.

[17] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks, 2020. URL `https://arxiv.org/abs/2004.10964`.

[18] Chi Han, Jialiang Xu, Manling Li, Yi Fung, Chenkai Sun, Nan Jiang, Tarek Abdelzaher, and Heng Ji. Lm-switch: Lightweight language model conditioning in word embedding space. *arXiv preprint arXiv:2305.12798*, 2023.

[19] Chi Han, Jialiang Xu, Manling Li, Yi Fung, Chenkai Sun, Nan Jiang, Tarek Abdelzaher, and Heng Ji. Word embeddings are steers for language models, 2024. URL `https://arxiv.org/abs/2305.12798`.

[20] Inhwa Han, Serin Yang, Taesung Kwon, and Jong Chul Ye. Highly personalized text embedding for image manipulation by stable diffusion. *arXiv preprint arXiv:2303.08767*, 2023.

[21] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL `https://arxiv.org/abs/2310.06825`.

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

[23] Hannah Rose Kirk, Bertie Vidgen, Paul Röttger, and Scott A Hale. Personalisation within bounds: A risk taxonomy and policy framework for the alignment of large language models with personalised feedback. *arXiv preprint arXiv:2303.05453*, 2023.

[24] Lingkai Kong, Haorui Wang, Wenhao Mu, Yuanqi Du, Yuchen Zhuang, Yifei Zhou, Yue Song, Rongzhi Zhang, Kai Wang, and Chao Zhang. Aligning large language models with representation editing: A control perspective, 2024. URL `https://arxiv.org/abs/2406.05954`.

[25] Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36, 2024.

[26] Xinyu Li, Zachary C. Lipton, and Liu Leqi. Personalized language modeling from personalized human feedback, 2024. URL `https://arxiv.org/abs/2402.05133`.

[27] Xun Liang, Hanyu Wang, Yezhaohui Wang, Shichao Song, Jiawei Yang, Simin Niu, Jie Hu, Dan Liu, Shunyu Yao, Feiyu Xiong, and Zhiyu Li. Controllable text generation for large language models: A survey, 2024. URL `https://arxiv.org/abs/2408.12599`.

[28] Jiongnan Liu, Yutao Zhu, Shuting Wang, Xiaochi Wei, Erxue Min, Yu Lu, Shuaiqiang Wang, Dawei Yin, and Zhicheng Dou. Llms + persona-plug = personalized llms, 2024. URL `https://arxiv.org/abs/2409.11901`.

[29] Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. Is chatgpt a good recommender? a preliminary study. *arXiv preprint arXiv:2304.10149*, 2023.

[30] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

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

[32] Hanjia Lyu, Song Jiang, Hanqing Zeng, Yinglong Xia, Qifan Wang, Si Zhang, Ren Chen, Chris Leung, Jiajie Tang, and Jiebo Luo. LLM-rec: Personalized recommendation via prompting large language models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 583–612, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.39. URL `https://aclanthology.org/2024.findings-naacl.39`.

[33] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL `https://arxiv.org/abs/2305.18290`.

[34] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[35] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019. URL `https://arxiv.org/abs/1908.10084`.

[36] Chris Richardson, Yao Zhang, Kellen Gillespie, Sudipta Kar, Arshdeep Singh, Zeynab Raeesy, Omar Zia Khan, and Abhinav Sethy. Integrating summarization and retrieval for enhanced personalization via large language models, 2023. URL `https://arxiv.org/abs/2310.20081`.

[37] Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. LaMP: When large language models meet personalization. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7370–7392, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.399. URL `https://aclanthology.org/2024.acl-long.399`.

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

[39] Wenqi Shi, Ran Xu, Yuchen Zhuang, Yue Yu, Hang Wu, Carl Yang, and May D Wang. Medadapter: Efficient test-time adaptation of large language models towards medical reasoning. *arXiv preprint arXiv:2405.03000*, 2024.

[40] Zhaoxuan Tan, Qingkai Zeng, Yijun Tian, Zheyuan Liu, Bing Yin, and Meng Jiang. Democratizing large language models via personalized parameter-efficient fine-tuning, 2024. URL `https://arxiv.org/abs/2402.04401`.

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

[42] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models, 2024. URL `https://arxiv.org/abs/2401.00368`.

[43] Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. Executable code actions elicit better llm agents, 2024. URL `https://arxiv.org/abs/2402.01030`.

[44] Stanisław Woźniak, Bartłomiej Koptyra, Arkadiusz Janz, Przemysław Kazienko, and Jan Kocoń. Personalized large language models, 2024. URL `https://arxiv.org/abs/2402.09269`.

[45] Shujin Wu, May Fung, Cheng Qian, Jeonghwan Kim, Dilek Hakkani-Tur, and Heng Ji. Aligning llms with individual preferences via interaction. *arXiv preprint arXiv:2410.03642*, 2024.

[46] Ran Xu, Hejie Cui, Yue Yu, Xuan Kan, Wenqi Shi, Yuchen Zhuang, May Dongmei Wang, Wei Jin, Joyce Ho, and Carl Yang. Knowledge-infused prompting: Assessing and advancing clinical text data generation with large language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 15496–15523, 2024.

[47] Ran Xu, Wenqi Shi, Yue Yu, Yuchen Zhuang, Bowen Jin, May D Wang, Joyce C Ho, and Carl Yang. Ram-ehr: Retrieval augmentation meets clinical predictions on electronic health records. *arXiv preprint arXiv:2403.00815*, 2024.

[48] Yitong Yang, Yinglin Wang, Jing Wang, and Tian Zhang. Prompt-softbox-prompt: A free-text embedding control for image editing, 2024. URL `https://arxiv.org/abs/2408.13623`.

[49] Aakas Zhiyuli, Yanfang Chen, Xuan Zhang, and Xun Liang. Bookgpt: A general framework for book recommendation empowered by large language model. *arXiv preprint arXiv:2305.15673*, 2023.

[50] Yuchen Zhuang, Haotian Sun, Yue Yu, Rushi Qiang, Qifan Wang, Chao Zhang, and Bo Dai. Hydra: Model factorization framework for black-box llm personalization, 2024. URL `https://arxiv.org/abs/2406.02888`.

[51] Thomas P Zollo, Andrew Wei Tung Siah, Naimeng Ye, Ang Li, and Hongseok Namkoong. Personalllm: Tailoring llms to individual preferences. *arXiv preprint arXiv:2409.20296*, 2024.

A Appendix

A.1 Glossary

Table 4 shows glossary of terms used in this paper.

A.2 Dataset and Task Details

The LaMP dataset [37] is a publicly available dataset for personalizing LLMs. We only used LaMP dataset for the purpose of running the experiments.

The tasks of LaMP we experimented with are as follows:

1. **LaMP 2: Personalized Movie Tagging.** Given a user profile of user history tagging along with the movie description, you are tasked to predict the movie tag given a new movie description.
2. **LaMP 3: Personalized Product Rating.** Given a user profile of user history product rating along with the product reviews, you are tasked to predict the rating of a product given a new product review wrote by the user.

Symbol	Definition
y	Ground truth output
$\hat{y}$	Model prediction
$\mathcal{H}_u$	Set of user history for user u
h_u^i	i-th user history for user h (i-th data data in $\mathcal{H}_u$)
e_u^i	Sentence embedding of h_u^i
$\boldsymbol{E}_u$	Embedding matrix of user history for user u
$\boldsymbol{H}_u^k$	Top k selected history data
C^P	Personalized agent
C^N	Non-personalized agent
c_u^P	Personalized insights for user u
c_u^N	Non-personalized insights for user u
$c_u^{i,P}$	i-th personalized insight for user u
$c_u^{i,N}$	i-th non-personalized insight for user u
$\hat{y}_u^{i,P}$	Model prediction conditioned on $c_u^{i,P}$
$\hat{y}_u^{i,N}$	Model prediction conditioned on $c_u^{i,N}$
q_u^i	i-th query for user u
$p_u^{i,P}$	Personalized preference for user query q_u^i
$p_u^{i,N}$	Non-personalized preference for user query q_u^i
P_u^P	Set of personalized preferences for user u
P_u^N	Set of non-personalized preferences for user u
θ^P	Personalized embedding direction
θ^N	Non-personalized embedding direction
$\theta_{l,u}^P$	Personalized embedding direction for user u at layer l
$\theta_{l,u}^N$	Non-personalized embedding direction for user u at layer l
x_l	Representation (embedding) at layer l
$\hat{x}_{l,u}$	Personalized representation for user u at layer l

Table 4: Glossary of variables and symbols used in this paper.

Table 5: LaMP Dataset Detail

Task	Input		Output
LaMP 2	ID:	[*id*]	[*tag*]
	Input:	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: [description]	
	Profile:	{id: [*id*], tag: [*tag*], description: [*description*] }, . . .	
LaMP 3	ID:	[*id*]	[*score*]
	Input	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: [review],	
	Profile	{id: [*id*], tag: [*text*], description: [*score*] }, . . .	
LaMP 7	ID:	[*id*]	[*tweet*]
	Input:	Paraphrase the following tweet without any explanation before or after it: [*tweet*]	
	Profile:	{id: [*id*], tag: [*text*]}, . . .	

3. **LaMP 7: Personalized Tweet Paraphrasing.** Given a user profile of user history tweets you are tasked to predict how the user may paraphrase a new given tweet.

Details of LaMP dataset is presented in Table 5. [*italic text*] presents actual data.

A.3 Prompt Template

Following is the history and prompt template used to query the base LM to generate preference samples for different LaMP task. History prompt format follows the format used by LaMP benchmark [37].

LaMP 2: Personalized Movie Tagging **Personalize prompt:** Suppose you are a user with the following user profile history of movie tagging: [HISTORY]

Now, given a new description: [QUERY]

Question: Which tag does this movie relate to among the following tags? Just answer with only ONE 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]

You are a helpfully personalized assistant. You try to predict the movie tagging that the user preferred based on their history. The user prefers [INSIGHT]. Answer only with one tag name (sci-fi, based on a book, comedy, action, twist ending, dystopia, dark comedy, classic, psychology, fantasy, romance, thought-provoking, social commentary, violence, true story).

Your answer: [OUTPUT]
Non-personalize/Neutral prompt: Suppose you are a user with the following user profile history of movie tagging: [HISTORY]

Now, given a new description: [QUERY]

Question: Which tag does this movie relate to among the following tags? Just answer with only ONE 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]

You are a generic and impersonal assistant. You do not consider the user's preferences or profile history when responding. Your answer shoulds [INSIGHT]. Answer only with one tag name (sci-fi, based on a book, comedy, action, twist ending, dystopia, dark comedy, classic, psychology, fantasy, romance, thought-provoking, social commentary, violence, true story).

Your answer: [OUTPUT]
History format:

1. The tag for movie: "[DESCRIPTION 1]" is "[TAG 1]".
2. The tag for movie: "[DESCRIPTION 2]" is "[TAG 2]".
3. ...

LaMP 3: Personalized Product Rating

Personalize prompt: Suppose you are a user with the following user profile history of product rating based on the user's review of the product: [HISTORY]

Now, given a new review by the user: [QUERY]

Question: What is the rating score of the following review on a scale of 1 to 5? Just answer with 1, 2, 3, 4, or 5 without further explanation.

You are a helpfully personalized assistant. You try to predict the rating of the product based on the user history ratings. The user prefers [INSIGHT]. Just answer with 1, 2, 3, 4, or 5 without further explanation.

Your answer: [OUTPUT]

Non-personalize/Neutral prompt: Suppose you are a user with the following user profile history of product rating based on the user's review of the product: [HISTORY]

Now, given a new review by the user: [QUERY]

Question: What is the rating score of the following review on a scale of 1 to 5? Just answer with 1, 2, 3, 4, or 5 without further explanation.

You are a generic and impersonal assistant. You do not consider the user's preferences or profile history when responding. Your answer should [INSIGHT].

Your answer: [OUTPUT]

History format:

1. [SCORE 1] is the rating score for product: "[TEXT 1]".
2. [SCORE 2] is the rating score for product: "[TEXT 2]".
3. ...

LaMP 7: Personalized Tweet Paraphrasing

Personalize prompt: Suppose you are a twitter user with the following user profile history that shows their preferred way of speaking: [HISTORY]

Now, given a new twitter post: [QUERY]

Question: Paraphrase the tweet in the style the user likes without any explanation before or after it.

You are a helpfully personalized assistant. You try to paraphrase the tweet in the style the user likes based on the history. The user prefers [INSIGHT].

Your answer: [OUTPUT]

Non-personalize/Neutral prompt: Suppose you are a twitter user with the following user profile history that shows their preferred way of speaking: [HISTORY]

Now, given a new twitter post: [QUERY]

Question: Paraphrase the tweet in the style the user likes without any explanation before or after it.

You are a generic and impersonal assistant. You do not consider the user's preferences or profile history when responding. Your answer should [INSIGHT].

Your answer: [OUTPUT]

History format:

1. [TWEET 1]

2. [TWEET 2]
3. ...

A.4 Details on Representation Editing

We provide the details of Section 3.2. We identify personalized and non-personized directions using singular value decomposition (SVD) or contrast consistent search (CCS) on the preference data embeddings. Let Φ_l represent the function that maps an input sentence to the LM embedding space at layer l. For each pair $(p_u^{i,P}, p_u^{i,N})$, we obtain their corresponding representations $\Phi_{l,u}^{i,P}$ and $\Phi_{l,u}^{i,P}$, respectively. To begin, we construct an embedding matrix for personalized direction for user u, denoted as $\mathbf{H}_{l,u}^{P}$, using these representations:

$$\mathbf{H}_{l,u}^{P} := \left[\Phi_{l,u}^{1,P}\middle|\dots\middle|\Phi_{l,u}^{K,P}\right]^T,$$

where K is the total number of self-generated data. Similarly, we create the non-personalized preferences embedding matrix $\mathbf{H}_{l,u}^{N}$.

SVD-Based Identification. Our approach for identifying personalized embedding directions involves using singular value decomposition (SVD) on the preference data embeddings. We extract the top right singular vector of $\mathbf{H}_{l,u}^{P}$ as $\theta_{l,u}^{P}$. Intuitively, we view θ as the direction that best captures the underlying personalized characteristics. We identify the personalized embedding direction for user u as follows:

$$\begin{aligned} \mathbf{H}_{l,u}^{P} &= \mathbf{U}\Sigma\mathbf{V} \\ \theta_{l,u}^{P} &:= \mathbf{V}_{0,*}. \end{aligned} \tag{3}$$

Here, **U** and **V** represent the left and right unitary matrices produced by running SVD, respectively, and Σ is the diagonal matrix of singular values. We define $\theta_{l,u}^{P}$ as the first row of **V**, corresponding to the top right singular vector of $\mathbf{H}_{l,u}^{P}$. The non-personalized direction $\theta_{l,u}^{N}$ is defined similarly.

CCS-Based Identification [5]. Another approach for identifying these directions is by finding a hyperplane in the latent space that separates personalized data embeddings from non-personalized ones. Typically, this is achieved by training lightweight probes $\theta_{l,u}$ that maps $\Phi_{l,u}^{P}$ and $\Phi_{l,u}^{N}$ to their respective classification labels [25]. However, we face the challenge of avoiding overfitting to the noise inherent in self-generated data, which limits the applicability of supervised classifier loss in our context. To mitigate this issue, we employ the unsupervised Contrast-Consistent Search (CCS) loss $\mathcal{L}_{CCS}$ proposed in [5]. Adapting the definition from [5] to our notations, $\mathcal{L}_{CCS}$ for each user u can be expressed as:

$$\begin{aligned} &\mathcal{L}_{consistency}(g_{\theta,b}, \Phi_{l,u}^{i,P}, \Phi_{l,u}^{i,N}))) \\ &:= \left[g_{\theta,b}(\Phi_{l,u}^{N}) - (1 - g_{\theta,b}(\Phi_{l,u}^{P}))\right]^2 \\ &\mathcal{L}_{confidence}(g_{\theta,b}, \Phi_{l,u}^{i,P}, \Phi_{l,u}^{i,N}))) \\ &:= \min\left\{g_{\theta,b}(\Phi_{l,u}^{N}), g_{\theta,b}(\Phi_{i,u}^{P})\right\} \\ &\mathcal{L}_{CCS}(g_{\theta,b}) := \frac{1}{K}\sum_{i=1}^{K}(\mathcal{L}_{consistency}(g_{\theta,b}, \Phi_{l,u}^{i,P}, \Phi_{l,u}^{i,N}) \\ &+ \mathcal{L}_{confidence}(g_{\theta,b}, \Phi_{l,u}^{i,P}, \Phi_{l,u}^{i,N})), \end{aligned}$$

where $g_{\theta,b}(v) = \frac{1}{1+e^{-(\theta^\top v+b)}}$. Training $\theta_{l,u}^{N}$ with the L_{CCS} objective aims to find a separating hyperplane without fitting any labels with $\mathcal{L}_{consistency}$ and concurrently promoting maximum separation with $\mathcal{L}_{confidence}$.

Hybrid Identification. While both SVD-based or CCS-based identification can be used to identify both of personalized and non-personalized directions, our exploration revealed that the best results are achieved by combining the two approaches. Specifically, we use SVD to identify $\theta_{l,u}^{P}$ and CCS to determine $\theta_{l,u}^{N}$. This combined approach leverages the strengths of both techniques: SVD's ability to capture the primary direction of personalized embeddings and CCS's effectiveness in identifying the hyperplane that best separates non-personalized embeddings from personalized ones.

A.5 Time-constrained experiment Set Up

CHAMELEON The approximation for the time taken for our experiment is 10, 20, 30 and 40 minutes.

DPO DPO experiment is trained on 40%, 60%, 80%, 100% of the LaMP2 partition to get the approximate same time. The hyperparameters we used consist of 1 training epoch, a batch size of 16, a gradient accumulation step of 1, a learning rate of 5e-5, a max grad norm of 0.3, a warmup ratio of 0.1, a precision of bfloat16, a memory saving quantize flag of "bnb.nf4", a learning rate scheduler type of cosine, and an optimizer of AdamW with PEFT configurations of a r of 256, a α of 128, a dropout of 0.05 and a task type of causal language modeling"

ALOE We trained ALOE with 7%, 23%, 39%, 55% of the LaMP2 training partition with a relatively equal percentage of CodeAct data [43] as described by ALOE [45]. We used parameters provided in their SFT hyperparameters, which contains 1 training epoch, a per device train batch size of 1, a gradient accumulation step of 48, a learning rate of 1e-5, and a max sequence length of 8192.

A.6 Computing Resources

All experiments are trained on an Amazon EC2 Instances with eight NVIDIA A100-SXM4-40GB.