Comparing Retrieval-Augmentation and Parameter-Efficient Fine-Tuning for Privacy-Preserving Personalization of Large Language Models

Alireza Salemi and **Hamed Zamani**
University of Massachusetts Amherst
{asalemi,zamani}@cs.umass.edu

Abstract

Privacy-preserving methods for personalizing large language models (LLMs) are relatively under-explored. There are two schools of thought on this topic: (1) generating personalized outputs by personalizing the input prompt through retrieval augmentation from the user's personal information (RAG-based methods), and (2) parameter-efficient fine-tuning of LLMs per user that considers efficiency and space limitations (PEFT-based methods). This paper presents the first systematic comparison between two approaches on a wide range of personalization tasks using seven diverse datasets. Our results indicate that RAG-based and PEFT-based personalization methods on average yield 14.92% and 1.07% improvements over the non-personalized LLM, respectively. We find that combining RAG with PEFT elevates these improvements to 15.98%. Additionally, we identify a positive correlation between the amount of user data and PEFT's effectiveness, indicating that RAG is a better choice for cold-start users (i.e., user's with limited personal data).

1 Introduction

Personalizing large language models (LLMs) has recently emerged as a critical topic in natural language processing (Salemi et al., 2024b,a; Kumar et al., 2024) due to its applications in various real-world systems, such as personalized recommender system (Hua et al., 2023; Chen, 2023), virtual assistants (Li et al., 2024b; Kocaballi et al., 2019), and targeted content generation (Alhafni et al., 2024). These systems benefit from tailoring responses and actions based on individual user preferences.

While various approaches exist for personalizing LLMs, they can be categorized into two schools of thought: those that only modify the input provided to the LLMs and those that alter the parameters of the LLMs. Retrieval-augmented generation (RAG) can be considered part of the first group, where personalized information is retrieved from the user's profile and used to generate a personalized prompt for the LLMs (Salemi et al., 2024b). In the second group, parameter-efficient fine-tuning (PEFT), such as low-rank adaptation (LoRA) (Hu et al., 2022), can be used to tune the parameters of LLMs for each user data separately for personalization, as keeping a whole set of model parameters for each user is impractical in real-world applications. Both of these approaches preserve the privacy of users as they do not update LLM parameters and do not create input prompts using data from other users.

We conduct an extensive set of experiments to compare these two schools of thought on seven diverse datasets obtained from the Language Model Personalization (LaMP) benchmark (Salemi et al., 2024b). In more detail, LaMP consists of three text classification tasks and four text generation tasks. Each input in this benchmark is treated as a separate user, with its own specific input, expected output, and user profile, making it an ideal test case for evaluating the personalization methods explored in this paper. Our experiments show that personalizing LLMs using RAG results in an average improvement of 14.92% over the non-personalized baseline, while PEFT-based personalization leads to only a 1.07% improvement. Additionally, we demonstrate that combining both approaches achieves the best results, with a 15.98% improvement over the non-personalized baseline. Furthermore, our analysis provides insight into why PEFT does not perform as well for personalizing LLMs. In most cases, we found a positive correlation between the size of the user profile and the performance improvement, suggesting that the lack of sufficient data per user is a key reason for PEFT's underperformance. To encourage future research in this area, we have open-sourced our codebase.[1]

[1]The code and data are available at: https://github.com/LaMP-Benchmark/LaMP

2 Problem Formulation

This paper focuses on personalized text generation, aiming to produce outputs that are tailored to the preferences of a user. We assume access to a dataset $T = \{(x_i, y_i, P_i)\}_{i=1}^{|T|}$, where x_i is the input prompt from user u_i, y_i is the expected output for user u_i, and P_i is the user profile. Here, a user profile P_i consists of a set of unstructured text documents for the user u_i, denoted as $P_i = \{d_{(i,j)}\}_{j=1}^{|P_i|}$.

This paper aims to utilize the information about user u_i available in profile P_i to construct a personalized LLM $M_i = \text{PERSONALIZE}(M, P_i)$ by applying a transformation PERSONALIZE to the LLM M. This function can either modify the parameters of the LLM M to construct M_i or simply alter the input to the LLM based on the profile P_i. We focus on comparing different methods for designing the transformation PERSONALIZE while keeping privacy, which means we cannot use information from other users to personalize the LLM for a user.

3 LLM Personalization Approaches

3.1 RAG for Personalizing LLMs

Given an input prompt x from the user, we use the query generation function ϕ_q to create a query. This query is then passed through the retriever R, which retrieves k documents from the user's profile P_u. Finally, the prompt generation function ϕ_p combines the retrieved documents and the input prompt to generate a personalized prompt, which is used as the input to the LLM M to generate a more tailored response, formally, defined as:

$$\bar{y} = M(\phi_p(x, R(\phi_q(x), k))) \tag{1}$$

Note that this approach does not modify the LLM itself. Instead, it adjusts its input, using a tailored prompt to the user based on the retrieved documents from the user profile. This allows us to personalize the LLM's response without altering its underlying structure and parameters, which works on any black-box LLM. The implementation details for ϕ_q and ϕ_p are provided in Table 3 in Appendix B. In our experiments, we used a wide range of retrieval models: BM25 (Robertson et al., 1994) as a lexical-matching retrieval model, Contriever (Izacard et al., 2022) as a semantic matching retrieval model, Recency (Salemi et al., 2024b) as a time-aware retrieval model, and RSPG (Salemi et al., 2024a) as an ensemble model that chooses an appropriate retrieval model per input. More information is provided in Appendix A.

3.2 PEFT for Personalizing LLMs

Keeping a separate LLM for each user is infeasible for systems with many users. For example, storing FlanT5-XXL (Chung et al., 2024) requires 45 GB per user. Conversely, a LoRA adapter with $r = 8$ for the same model only needs 55 MB. For 1 million users, this totals 55 TB, making it more practical for real-world applications. Thus, using PEFT is a more cost-efficient solution.

This approach uses a user profile P_u for learning user-specific parameters, resulting in a personalized LLM M_u. There are different ways to do this; we apply LoRA to the LLM M and train the model using the documents in P_u. LoRA fine-tunes LLMs by injecting trainable low-rank matrices into the model's weight matrices. Instead of updating the full weights during training, LoRA decomposes the weight update into two smaller, low-rank matrices $A \in \mathbb{R}^{d \times r}$ and $B \in \mathbb{R}^{r \times k}$, where r is the rank parameter. The original weights $W_0 \in \mathbb{R}^{d \times k}$ remain frozen, and only A and B are trained to approximate the updates ($W = W_0 + AB$). The parameter r controls the capacity of low-rank approximation.

To train the LLM on a user profile P_u, for each document d_i in P_u, we convert it into an input-output pair $(x_i, y_i) = \text{CONVERT}(d_i)$, and then train the model with the seq2seq cross-entropy loss (Sutskever et al., 2014) to generate y_i in its output given x_i as the input. There are different ways to implement CONVERT function. If the user profile consists of input-output pairs, they can be directly used for training. Alternatively, when the profile does not consist of explicit input-output pairs, these pairs can be automatically generated from each document from the profile. Whenever the profile contains input-output pairs for a user, we use those directly. Otherwise, we define text completion as the training task. The function used for different tasks is summarized in Table 4 in Appendix C.

3.3 PEFT-RAG for Personalizing LLMs

This approach integrates both PEFT and RAG to personalize an LLM. First, we train the LLM M on a user profile P_u using the method described in Section 3.2, resulting in the personalized model M_u. Next, we apply the RAG personalization approach outlined in Section 3.1, denoted as:

$$\bar{y} = M_u(\phi_p(x, R(\phi_q(x), k))) \tag{2}$$

where M_u is a personalized model with PEFT, R is a retriever, ϕ_q and ϕ_p are the query and prompt generation functions, and k is the number of retrieved documents. The key difference from Equation 1 is that this first trains the LLM on the user-specific profile to learn preferences before applying RAG.

4 Experiments

Setup. Following Salemi et al. (2024b), FlanT5-XXL (Chung et al., 2024), with 11 billion parameters, is used. The experiments are conducted on the LaMP benchmark (Salemi et al., 2024b), consisting of seven personalized tasks: three text classification—binary, categorical, and ordinal—and four text generation tasks. For binary classification, we use accuracy; for categorical, both accuracy and F1 scores; and for ordinal, MAE and RMSE. For text generation, ROUGE-1 and ROUGE-L (Lin, 2004) are used. The detailed setup is in Appendix A.

How do PEFT- and RAG-based approaches perform for LLM personalization? The results of PEFT- and RAG-personalization and the non-personalized baseline are reported in Table 1. The results suggest that using PEFT improves performance compared to non-personalized LLMs in 5 out of 7 datasets. Similarly, the RAG approach leads to performance improvements across all datasets. Comparing PEFT with RAG, the results indicate that using RAG is more effective than using PEFT, as shown in Table 1. Specifically, PEFT achieves a 1.07% improvement over non-personalized LLMs, whereas the RAG approach achieves an average improvement of 14.92%. This clearly indicates that retrieval-augmented generation is a superior approach for personalizing LLMs. Note that the different retrieval models in Table 1 achieve varying levels of improvement. However, RSPG (Salemi et al., 2024a), which dynamically selects the best retrieval model for each instance, outperforms all other retrieval models in terms of overall performance. Additionally, the rank parameter r also influences performance. Specifically, on LaMP-1, LaMP-3, and LaMP-6, increasing r leads to noticeable improvements in performance, while it does not significantly affect other tasks.

How does the combination of PEFT and RAG impact the personalization performance? To address this, we use the best retrieval model from Table 1 and combine it with each user's personalized LLM, trained using PEFT, to perform RAG personalization with PEFT. The results of this experiment are reported in Table 1. The findings suggest that combining RAG with PEFT leads to improvements over RAG in 4 out of 7 tasks. Additionally, this approach results in a 15.98% improvement over the non-personalized LLM, which is 0.44% more relative improvement over RAG personalization. Thus, this combination appears to be an effective for enhancing personalization.

How does profile size and data presence in training corpus affect performance? We create a regression plot[2] between the profile size and relative improvement obtained by the best personalized LLM using PEFT- and RAG-based personalization versus the non-personalized LLM. We define improvement as 1 if there is a gain over no-personalization and -1 if there is no gain. Since many users do not experience any change in their performance, we excluded them from the analysis. This plot is depicted in Figure 1. This figure indicates that, for 5 out of 7 datasets, there is a positive correlation between the number of items in the user profile and performance improvement for PEFT. For the LaMP-5 task, where we observe a negative or zero correlation, one explanation is that the profiles consist of abstracts from papers authored by the user, which are often collaborative works. Here, users with larger profiles tend to be senior researchers who may not have been directly involved in the writing process. We found that in 94% (17 out of 18) of performance drop cases, the user was not the primary author on most papers in their profile. Thus, training on such data is less effective for personalizing the LLM for their preferences. Finally, when considering all users across all tasks, we observe a positive correlation between performance improvement and the number of items in a user's profile. Conversely, we observe a negative correlation between the improvement in RAG-based personalization and the non-personalized baseline. This indicates that as the profile grows, retrieval models face difficulty in identifying and retrieving the relevant documents to personalize the LLM. These observations suggest that one reason PEFT does not perform as well as RAG for personalizing LLMs may be the insufficient amount of training data per user, which limits the model's ability to learn from the user.

Another observation in Table 1 is that PEFT performs better on the LaMP-6 task compared to other tasks. Since FlanT5 (Chung et al., 2024) is trained on public datasets, and the LaMP-6 dataset con-

[2] https://seaborn.pydata.org/generated/seaborn.regplot.html

Dataset	Metric	No Personalization	PEFT Personalization $r=8$	PEFT $r=16$	PEFT $r=32$	PEFT $r=64$	RAG Personalization BM25	RAG Recency	RAG Contriever	RAG RSPG	PEFT-RAG Personalization $r=8$	PEFT-RAG $r=16$	PEFT-RAG $r=32$	PEFT-RAG $r=64$
LaMP-1: Personalized Citation Identification	Accuracy ↑	0.502	0.502	0.502	0.504	0.506	0.626	0.622	0.636	**0.672**	0.670	0.668	0.671	0.671
LaMP-2: Personalized Movie Tagging	Accuracy ↑	0.359	0.360	0.360	0.360	0.359	0.387	0.377	0.396	0.430	0.430	**0.431**	0.430	0.430
	F1 ↑	0.276	0.278	0.278	0.278	0.277	0.306	0.295	0.304	0.339	0.341	**0.342**	0.341	0.341
LaMP-3: Personalized Product Rating	MAE ↓	0.308	0.308	0.307	0.306	0.301	0.298	0.296	0.299	0.264	0.264	0.265	0.264	**0.259**
	RMSE ↓	0.611	0.607	0.607	0.602	0.600	0.611	0.605	0.616	0.568	0.568	0.570	0.564	**0.562**
LaMP-4: Personalized News Headline Generation	ROUGE-1 ↑	0.176	0.178	0.177	0.178	0.178	0.186	0.189	0.183	0.203	0.203	**0.204**	**0.204**	0.203
	ROUGE-L ↑	0.160	0.162	0.162	0.163	0.163	0.171	0.173	0.169	0.186	0.186	0.186	**0.187**	0.186
LaMP-5: Personalized Scholarly Title Generation	ROUGE-1 ↑	0.478	0.478	0.478	0.477	0.478	0.477	0.475	**0.483**	0.480	0.481	0.480	0.480	0.479
	ROUGE-L ↑	0.428	0.429	0.429	0.428	0.428	0.427	0.426	**0.433**	0.429	0.431	0.431	0.431	0.431
LaMP-6: Personalized Email Subject Generation	ROUGE-1 ↑	0.335	0.342	0.342	0.341	0.343	0.412	0.403	0.401	0.433	0.436	0.436	0.436	**0.437**
	ROUGE-L ↑	0.319	0.325	0.326	0.325	0.326	0.398	0.389	0.386	0.418	**0.422**	**0.422**	**0.422**	0.421
LaMP-7: Personalized Tweet Paraphrasing	ROUGE-1 ↑	0.449	0.449	0.449	0.449	0.449	0.446	0.444	0.440	**0.461**	0.460	0.460	0.460	0.460
	ROUGE-L ↑	0.396	0.397	0.397	0.396	0.396	0.394	0.393	0.390	**0.409**	**0.409**	**0.409**	0.408	**0.409**

Table 1: The performance of utilized LLM personalization approaches on the LaMP benchmark.

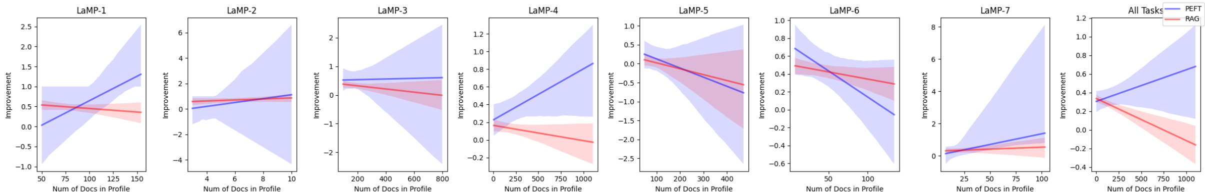

Figure 1: Correlation (with confidence interval) between profile item count and performance improvement of PEFT- and RAG-based personalization in comparison with no personalization on the tasks in the LaMP benchmark.

tains PII information, the Avocado (Oard, Douglas et al., 2015) corpus was not included in its training corpus. Consequently, it has not been exposed to this corpus. We believe that the improvement on this task is due to the LLM encountering this corpus for the first time and thus learning more from it. In contrast, other datasets are based on public data that FlanT5 is already trained on them. This suggests that training the LLM using PEFT on private user data can yield considerable improvements.

5 Related Work

Retrieval-Augmented Generation has proven effective for text generation in knowledge grounding for textual (Izacard and Grave, 2021b; Asai et al., 2024; Petroni et al., 2021) and multi-modal (Salemi et al., 2023a,b; Gui et al., 2022), and reducing hallucinations (Agrawal et al., 2024; Shuster et al., 2021). In RAG, a retriever— general (Wang et al., 2024; Salemi and Zamani, 2024b) or task-specific (Izacard and Grave, 2021a; Izacard et al., 2024)—retrieves relevant documents based on the input, which are used to generate a response. The quality of retrieval and the LLM's ability to use the retrieved information (Kim et al., 2024; Salemi and Zamani, 2024a; Lewis et al., 2020; Zamani et al., 2022) are crucial to performance.

Parameter Efficient Fine-Tuning optimizes LLMs to specific tasks without full model training (Houlsby et al., 2019), updating only a small parameter set to reduce computational cost while maintaining performance(Liao et al., 2023; Han et al., 2024). Low-Rank Adaptation (LoRA), a well-known PEFT method, adapts LLMs with minimal parameter updates by introducing low-rank decomposition into weight matrices and injecting trainable low-rank matrices into frozen weights (Hu et al., 2022; Li et al., 2024c; Lialin et al., 2024).

Personalizing LLMs is an important topic, where Salemi et al. (2024b) introduced a RAG-based method for personalizing LLMs and the LaMP benchmark for evaluating personalized tasks. Li et al. (2023) explored this in long-form text generation, while others have focused on personalized assistants (Mysore et al., 2023; Zhang et al., 2024; Lu et al., 2024). Various techniques have been studied, such as training retriever with LLM feedback on personalized outputs (Salemi et al., 2024a), summarizing user profiles (Richardson et al., 2023), alignment with personalized feedback (Jang et al., 2023), and automatic personalized prompt generation (Li et al., 2024a). PEFT was used to personalize LLMs by a shared pool of adapters for users (Tan et al., 2024), which raises privacy concerns as the model is trained on data from multiple users.

6 Conclusion

This paper investigates personalizing LLMs using RAG and PEFT. The results indicate that RAG significantly outperforms PEFT. Furthermore, combining RAG with PEFT leads to improvements in personalized tasks compared to personalization using RAG or PEFT alone. Finally, we provide evidence suggesting that the insufficient number of documents per user contributes to the poor performance of PEFT in personalizing LLMs.

Limitations

This work has limitations related to resource intensivity and adaptor loading and retrieval latency.

Resource Intensivity. Personalizing LLMs, particularly using PEFT with LoRA can be resource-intensive. Training models with LoRA requires significant computational resources. This can lead to increased costs and extended training times, which may limit the scalability of these methods in resource-constrained environments. For this reason, in this paper, we were only able to conduct our experiments on a single LLM, FlanT5-XXL (Chung et al., 2024), which has 11 billion parameters, following Salemi et al. (2024b). While running similar experiments on other LLMs could provide valuable insights, it is prohibitively expensive. In this work, we utilized over 10,000 hours of A100 GPU computation for training and experimentation. Based on the average figures reported by Dodge et al. (2022), the total computational effort for these experiments would result in the generation of at least 400 kilograms of CO2 if conducted on cloud-based GPU providers. Since we ran the experiments locally, which may be less efficient in terms of energy usage and CO2 emissions, the actual carbon footprint could be even higher. This highlights the environmental cost associated with large-scale LLM experiments, further complicating scalability. Scaling this study to include multiple LLMs would significantly increase the computational costs, making it infeasible for us to pursue at this time.

In addition to the high cost of training these models, storing them can also be challenging. For example, if each adapter requires 200 MB of disk space, a website with 100,000,000 users would need 20 PB of disk space just to store the adapters. Addressing these challenges is crucial for the practical deployment of such systems. Studying solutions to overcome issues will be an important step towards the real-world application of personalized LLMs.

Adaptor Loading and Retrieval Latency. In the context of our comparison between RAG and PEFT methods, adaptor loading and retrieval latency emerge as critical factors influencing overall system performance. For RAG models, retrieval latency is a prominent concern. The process of querying external databases and loading relevant information incurs time costs that can impact the responsiveness of the system. High retrieval latency may hinder the efficiency of real-time applications where prompt responses are crucial. Additionally, complications such as the need for efficient indexing and managing a large corpus of data can further exacerbate latency issues. On the other hand, PEFT approaches involve adapting pre-trained models through the use of adaptors or additional parameters. The process of loading these adaptors can introduce overhead, particularly when dealing with large-scale models or numerous adaptors. While PEFT is designed to be more resource-efficient compared to full fine-tuning, the integration and initialization of adaptors still require computational resources and time. This overhead can affect the deployment and operational efficiency of PEFT-based systems, especially in scenarios requiring frequent updates or real-time interactions.

Acknowledgment

This work was supported in part by the Center for Intelligent Information Retrieval, in part by Google, in part by Lowe's, and in part by Microsoft. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

Garima Agrawal, Tharindu Kumarage, Zeyad Alghamdi, and Huan Liu. 2024. Can knowledge graphs reduce hallucinations in LLMs? : A survey. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3947–3960, Mexico City, Mexico. Association for Computational Linguistics.

Bashar Alhafni, Vivek Kulkarni, Dhruv Kumar, and Vipul Raheja. 2024. Personalized text generation with fine-grained linguistic control. In *Proceedings of the 1st Workshop on Personalization of Generative AI Systems (PERSONALIZE 2024)*, pages 88–101, St. Julians, Malta. Association for Computational Linguistics.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

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

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. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Jesse Dodge, Taylor Prewitt, Remi Tachet des Combes, Erika Odmark, Roy Schwartz, Emma Strubell, Alexandra Sasha Luccioni, Noah A. Smith, Nicole DeCario, and Will Buchanan. 2022. Measuring the carbon intensity of ai in cloud instances. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 1877–1894, New York, NY, USA. Association for Computing Machinery.

Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 56–60, Vancouver. Association for Computational Linguistics.

Liangke Gui, Borui Wang, Qiuyuan Huang, Alexander Hauptmann, Yonatan Bisk, and Jianfeng Gao. 2022. KAT: A knowledge augmented transformer for vision-and-language. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 956–968, Seattle, United States. Association for Computational Linguistics.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey. *Preprint*, arXiv:2403.14608.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

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

Wenyue Hua, Lei Li, Shuyuan Xu, Li Chen, and Yongfeng Zhang. 2023. Tutorial on large language models for recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1281–1283.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Preprint*, arXiv:2112.09118.

Gautier Izacard and Edouard Grave. 2021a. Distilling knowledge from reader to retriever for question answering. In *International Conference on Learning Representations*.

Gautier Izacard and Edouard Grave. 2021b. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2024. Atlas: few-shot learning with retrieval augmented language models. *J. Mach. Learn. Res.*, 24(1).

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

To Eun Kim, Alireza Salemi, Andrew Drozdov, Fernando Diaz, and Hamed Zamani. 2024. Retrieval-enhanced machine learning: Synthesis and opportunities. *Preprint*, arXiv:2407.12982.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA.

Ahmet Baki Kocaballi, Shlomo Berkovsky, Juan C Quiroz, Liliana Laranjo, Huong Ly Tong, Dana Rezazadegan, Agustina Briatore, and Enrico Coiera. 2019. The personalization of conversational agents in health care: Systematic review. *J Med Internet Res*, 21(11):e15360.

Ishita Kumar, Snigdha Viswanathan, Sushrita Yerra, Alireza Salemi, Ryan A. Rossi, Franck Dernoncourt, Hanieh Deilamsalehy, Xiang Chen, Ruiyi Zhang, Shubham Agarwal, Nedim Lipka, and Hamed Zamani. 2024. Longlamp: A benchmark for personalized long-form text generation. *Preprint*, arXiv:2407.11016.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Cheng Li, Mingyang Zhang, Qiaozhu Mei, Weize Kong, and Michael Bendersky. 2024a. Learning to rewrite prompts for personalized text generation. In *Proceedings of the ACM on Web Conference 2024*, WWW '24. ACM.

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

Hao Li, Chenghao Yang, An Zhang, Yang Deng, Xiang Wang, and Tat-Seng Chua. 2024b. Hello again! llm-powered personalized agent for long-term dialogue. *Preprint*, arXiv:2406.05925.

Yixiao Li, Yifan Yu, Chen Liang, Nikos Karampatziakis, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2024c. Loftq: LoRA-fine-tuning-aware quantization for large language models. In *The Twelfth International Conference on Learning Representations*.

Vladislav Lialin, Sherin Muckatira, Namrata Shivagunde, and Anna Rumshisky. 2024. ReloRA: High-rank training through low-rank updates. In *The Twelfth International Conference on Learning Representations*.

Baohao Liao, Yan Meng, and Christof Monz. 2023. Parameter-efficient fine-tuning without introducing new latency. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4242–4260, Toronto, Canada. Association for Computational Linguistics.

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

Zhuoran Lu, Sheshera Mysore, Tara Safavi, Jennifer Neville, Longqi Yang, and Mengting Wan. 2024. Corporate communication companion (ccc): An llm-empowered writing assistant for workplace social media. *Preprint*, arXiv:2405.04656.

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

Oard, Douglas, Webber, William, Kirsch, David A., and Golitsynskiy, Sergey. 2015. Avocado research email collection.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics.

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. *Preprint*, arXiv:2310.20081.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at trec-3. In *Text Retrieval Conference*.

Alireza Salemi, Juan Altmayer Pizzorno, and Hamed Zamani. 2023a. A symmetric dual encoding dense retrieval framework for knowledge-intensive visual question answering. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 110–120, New York, NY, USA. Association for Computing Machinery.

Alireza Salemi, Surya Kallumadi, and Hamed Zamani. 2024a. Optimization methods for personalizing large language models through retrieval augmentation. In *Proceedings of the 47th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24. (to appear).

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2024b. LaMP: When large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7370–7392, Bangkok, Thailand. Association for Computational Linguistics.

Alireza Salemi, Mahta Rafiee, and Hamed Zamani. 2023b. Pre-training multi-modal dense retrievers for outside-knowledge visual question answering. In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval*, ICTIR '23, page 169–176, New York, NY, USA. Association for Computing Machinery.

Alireza Salemi and Hamed Zamani. 2024a. Evaluating retrieval quality in retrieval-augmented generation. In *Proceedings of the 47th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24. (to appear).

Alireza Salemi and Hamed Zamani. 2024b. Towards a search engine for machines: Unified ranking for multiple retrieval-augmented large language models. In *Proceedings of the 47th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24. (to appear).

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2024. Text embeddings by weakly-supervised contrastive pre-training. *Preprint*, arXiv:2212.03533.

Hamed Zamani, Fernando Diaz, Mostafa Dehghani, Donald Metzler, and Michael Bendersky. 2022. Retrieval-enhanced machine learning. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2875–2886, New York, NY, USA. Association for Computing Machinery.

Kai Zhang, Yangyang Kang, Fubang Zhao, and Xiaozhong Liu. 2024. LLM-based medical assistant personalization with short- and long-term memory coordination. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2386–2398, Mexico City, Mexico. Association for Computational Linguistics.

A Experiments Setup

Tasks & Datasets. In this paper, we conduct our experiments using the LaMP benchmark (Salemi et al., 2024b), which is designed to evaluate the personalization of LLMs. Each sample in the benchmark represents a user, including an input prompt, an expected output, and a set of items forming the user profile, provided in either structured or unstructured format. The benchmark features 7 personalized tasks: three text classification tasks and four text generation tasks. The dataset statistics for this benchmark are detailed in Table 2. We focus on the time-based configuration of this benchmark, as it provides shared users between the training and test sets. This allows us to train the models on user profiles for the PEFT approach.

RAG Configuration. For personalization LLMs using retreival-augmentation , we adopt the experimental setup used by Salemi et al. (2024b) and Salemi et al. (2024a). Specifically, we employ the BM25 (Robertson et al., 1994) retrieval model implemented in the *rank_bm25* library[3], as well as Contriever (Izacard et al., 2022), Recency, and RSPG (Salemi et al., 2024a). In all experiments, we retrieve $k = 4$ documents to personalize the LLM. Following (Salemi et al., 2024b), we utilize FlanT5-XXL (Chung et al., 2024) with 11 billion parameters as the LLM in our experiments. We configure the model with an input length of 512 tokens and an output length of 128 tokens. For generating outputs, we use beam search (Freitag and Al-Onaizan, 2017) with a beam size of 4.

PEFT Configuration. To train the LLMs for each user, we use the PEFT library[4]. We train the models for 50 epochs on each user profile with a learning rate of 5×10^{-4}, applying 5% of the steps as warmup with linear scheduler. The Adam optimizer (Kingma and Ba, 2015) is used with a weight decay of 10^{-4}, and a batch size of 16 is achieved through gradient accumulation. We use LoRA with dropout rate of 0.1 and $\alpha = 32$ in all experiments. LoRA is applied to all keys, queries, and values in the transformer (Vaswani et al., 2017). Following (Salemi et al., 2024b), we use FlanT5-XXL (Chung et al., 2024) with 11 billion parameters as the LLM in our experiments. The model is configured with an input length of 512 tokens and an output length of 128 tokens. For generating outputs, we use beam search (Freitag and Al-Onaizan, 2017) with a beam size of 4. We train the model on up to 32 Nvidia A100 GPUs with 80GB VRAM and 128GB RAM for up to 7 days. In total, over 10,000 GPU hours have been used for the experiments reported in this paper. To reduce the cost of training LLMs per user, we train an LLM only for each user present in the test sets, rather than for all users in the benchmark. In total, 37,560 adapters were trained, which occupy approximately 18 TB of disk space.

B Implementation of ϕ_q and ϕ_p for RAG Personalization

To implement the query generation function ϕ_q, following Salemi et al. (2024b), we extract and use the non-template portions of the user's input prompt as the query. For further details on the template used for generating inputs in the LaMP

[3]This library can be find at `https://github.com/dorianbrown/rank_bm25`

[4]Available at: `https://huggingface.co/docs/peft/en/index`

Task	#train	#dev	#test	Input Length	Output Length	#Profile Size	#classes
LaMP-1: Personalized Citation Identification	6542	1500	1500	51.43 ± 5.70	-	84.15 ± 47.54	2
LaMP-2: Personalized Movie Tagging	5073	1410	1557	92.39 ± 21.95	-	86.76 ± 189.52	15
LaMP-3: Personalized Product Rating	20000	2500	2500	128.18 ± 146.25	-	185.40 ± 129.30	5
LaMP-4: Personalized News Headline Generation	12500	1500	1800	29.97 ± 12.09	10.07 ± 3.10	204.59 ± 250.75	-
LaMP-5: Personalized Scholarly Title Generation	14682	1500	1500	162.34 ± 65.63	9.71 ± 3.21	87.88 ± 53.63	-
LaMP-6: Personalized Email Subject Generation	4821	1250	1250	454.87 ± 889.41	7.37 ± 2.78	55.67 ± 36.32	-
LaMP-7: Personalized Tweet Paraphrasing	13437	1498	1500	29.72 ± 7.01	16.96 ± 5.67	15.71 ± 14.86	-

Table 2: Statistics of the datasets within the LaMP benchmark (Salemi et al., 2024b) with time-based data separation.

Task	Per Profile Entry Prompt (PPEP)	Aggregated Input Prompt(AIP)
1: Citation Identification	"P_i[title]"	add_to_paper_title(concat([PPEP(P_1), ..., PPEP(P_n)], ", and "), [INPUT])
2: Movie Tagging	the tag for the movie: "P_i[description]" is "P_i[tag]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
3: Product Rating	P_i[score] is the score for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
4: News Headline Generation	"P_i[title]" is the title for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
5: Scholarly Title Generation	"P_i[title]" is the title for "P_i[abstract]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and ""). Following the given patterns [INPUT]
6: Email Subject Generation	"P_i[title]" is the title for "P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and "). [INPUT]
7: Tweet Paraphrasing	"P_i[text]"	concat([PPEP(P_1), ..., PPEP(P_n)], ", and ") are written by a person. Following the given patterns [INPUT]

Table 3: Prompts template used to augment the input of the LM with the user profile, following Salemi et al. (2024b). concat is a function that concatenates the strings in its first argument by placing the string in the second argument between them. add_to_paper_title is a function designed to add the string in its first argument to the paper's title in the Personalized Citation Identification task. PPEP is a function that create the prompt for each entry in the retrieved profile entries. [INPUT] is the task's input.

benchmark, we refer the reader to Salemi et al. (2024b). Additionally, we use the same function as Salemi et al. (2024b) to generate personalized prompts for the LLM, as detailed in Table 3.

C Implementation of CONVERT function for PEFT Personalization

The implementation of the input-output generation function for PEFT personalization involves different approaches depending on the task, as shown in Table 4. For LaMP-2, LaMP-3, LaMP-4, LaMP-5, and LaMP-7, user profiles contain input-output pairs that are directly used for training. However, tasks LaMP-1 and LaMP-7 require different strategies since such pairs are unavailable. In LaMP-1, the model is given a title and tasked with generating the corresponding abstract. For LaMP-7, a tweet is randomly split into two sections, and the model is asked to generate the second part based on the first.

Dataset	Profile Format	Generated Input (x_i)	Generated Output (y_i)
LaMP-1: Personalized Citation Identification	title: [title] abstract: [abstract]	Write an abstract for this title: [title]	[abstract]
LaMP-2: Personalized Movie Tagging	description: [description] tag: [tag]	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]	[tag]
LaMP-3: Personalized Product Rating	review: [review] score: [score]	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]	[score]
LaMP-4: Personalized News Headline Generation	article: [article] title: [title]	Generate a headline for the following article: [article]	[title]
LaMP-5: Personalized Scholarly Title Generation	abstract: [abstract] title: [title]	Generate a title for the following abstract of a paper: [abstract]	[title]
LaMP-6: Personalized Email Subject Generation	email: [email] title: [title]	Generate a subject for the following email: [email]	[title]
LaMP-7: Personalized Tweet Paraphrasing	tweet: [tweet]	Complete the following tweet: [first part of the tweet]	[second part of the tweet]

Table 4: The implementation of the input-output generation function for PEFT personalization. The profiles in LaMP-2, LaMP-3, LaMP-4, LaMP-5, and LaMP-7 consist of input-output pairs for the user, which are directly used as training pairs. However, for the LaMP-1 and LaMP-7 tasks, such pairs do not exist. For LaMP-1, we provide the model with a title and ask it to generate the abstract. For LaMP-7, we randomly divide a tweet into two parts and ask the model to generate the second part based on the first part.