Optimization Methods for Personalizing Large Language Models through Retrieval Augmentation

Alireza Salemi
University of Massachusetts Amherst
United States
asalemi@cs.umass.edu

Surya Kallumadi
Lowe's Companies, Inc.
United States
surya@ksu.edu

Hamed Zamani
University of Massachusetts Amherst
United States
zamani@cs.umass.edu

ABSTRACT

This paper studies retrieval-augmented approaches for personalizing large language models (LLMs), which potentially have a substantial impact on various applications and domains. We propose the first attempt to optimize the retrieval models that deliver a limited number of personal documents to large language models for the purpose of personalized generation. We develop two optimization algorithms that solicit feedback from the downstream personalized generation tasks for retrieval optimization–one based on reinforcement learning whose reward function is defined using any arbitrary metric for personalized generation and another based on knowledge distillation from the downstream LLM to the retrieval model. This paper also introduces a pre- and post-generation retriever selection model that decides what retriever to choose for each LLM input. Extensive experiments on diverse tasks from the language model personalization (LaMP) benchmark reveal statistically significant improvements in six out of seven datasets.

CCS CONCEPTS

• **Computing methodologies** → **Natural language generation**;
• **Information systems** → **Learning to rank**; **Personalization**.

KEYWORDS

Ranking optimization; retrieval-augmented generation; personalization; text generation

ACM Reference Format:
Alireza Salemi, Surya Kallumadi, and Hamed Zamani. 2024. Optimization Methods for Personalizing Large Language Models through Retrieval Augmentation. In *Proceedings of the 47th Int'l ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '24), July 14–18, 2024, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Personalization has been extensively explored by information retrieval (IR), recommender systems, and human-computer interaction communities, particularly in the context of information access [17, 36, 51]. Even though the recent advancements in large language models (LLMs) have revolutionized various applications, the existing commercial and open-source LLMs exhibit a significant limitation by failing to tailor their generated outputs according to the backgrounds and historical preferences of their users. As LLM-powered conversational agents become more prevalent, the need for *LLM personalization* becomes increasingly apparent [43]. LLM personalization has diverse applications, from customizing educational content and curating news feeds to improving e-commerce suggestions and delivering personalized healthcare information.

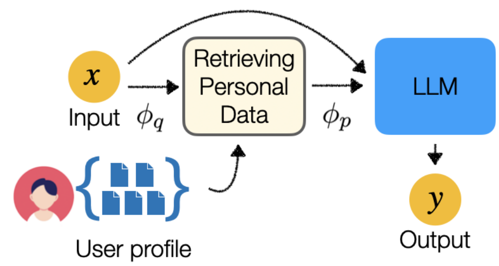

Figure 1: An overview of retrieval augmentation approaches for LLM personalization. First, the query function ϕ_q produces a query from the input x. Relevant personal information is then retrieved and fed to the personalized prompt generation function ϕ_p for LLM consumption.

Various approaches can be envisioned to personalize LLMs: (1) fine-tuning LLM parameters, either entirely or partially, for individual users, (2) integrating latent user representations with LLMs, and (3) enriching LLM prompts with user-specific content and/or context. The first two approaches involve adjusting LLM architecture and parameters, which is costly or even impractical in terms of storage, computation cost, and/or time. Besides, they cannot perform well for cold-start users. As an instantiation of retrieval-enhanced methods [55], the third approach, on the other hand, is applicable to any off-the-shelf LLM. To efficiently and effectively utilize the potentially extensive personal data for each active user, it is essential to implement a retrieval mechanism. As presented in Figure 1, this mechanism selects personal information that best enhances the LLM for the purpose of personalized text generation.

This paper focuses on optimization of personal information retrieval for the purpose of personalizing LLMs. For this purpose, standard learning-to-rank optimization methods are not applicable, since they typically require query-document-relevance triplets for training and it is not clear what documents is "relevant" for the downstream personalized text generation tasks. We study two retrieval optimization solutions for personalizing LLMs. First, we develop a reinforcement learning approach, where the training process involves sampling documents from the user's profile with respect to the probabilities generated based on retrieval scores. The

SIGIR '24, July 14–18, 2024, Washington, DC, USA.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

sampled documents are then fed to the LLM and its downstream performance for producing personalized output texts (using any arbitrary metric) is used to compute a reward function for optimizing the retrieval model. Second, we optimize the retrieval model by distilling knowledge from the LLM. We minimize the divergence between the retrieval score distribution and a target distribution computed using the downstream performance of LLM on producing personalized output texts using each individual retrieved document. To gauge the efficacy of these two optimization methods, we apply them to train dense retrieval models for LLM personalization.

Moreover, we observe that LLM personalization has multiple dimensions and existing retrieval models do not address all of them. For instance, retrieving *recent* user interactions may be needed for effective personalized text generation for an input, while a keyword-matching, a semantic matching, or a model that is aware of user's writing style may be optimal for another input. According to this observation, we hypothesize that a retrieval selection model that selects a model from a diverse pool of retrievers can impact LLM personalization. Following this hypothesis, we further develop *a pre- and a post-generation model mode for retrieval selection* that decides what retrieval model should be chosen for each given input. In our study, these models could choose between (1) no retrieval (i.e., no personalization), (2) recency-based retrieval, (3) term matching (i.e., BM25 [41]), (4) zero-shot semantic matching (Contriever [19]), and (5, 6) the two dense retrieval models developed in this paper that are specifically trained for LLM personalization. To optimize the retrieval selection models, we align the probability distribution obtained from the LLM downstream performance and the scores generated by the selection model for various retrieval models.

We evaluate our models using the LaMP benchmark [43], consisting of seven diverse personalization tasks, including three personalized text classification (binary, categorical, and ordinal) and four personalized text generation tasks. The methods proposed in this paper advance the state-of-the-art performance on six out of seven tasks in LaMP with statistically significant improvements. Our best-performing method exhibits an average of 5.5% state-of-the-art improvements across all LaMP datasets. Comparing to the non-personalized LLM, our best approach demonstrate 1.0%-33.8% improvements across all tasks, with an average improvement of 15.3%. To facilitate the research on this domain, we share our codes and trained model parameters to support future research, promoting transparency and reproducibility. [1]

2 RELATED WORK

Personalized Text Generation. Personalization has been a focal point of research in various domains, particularly within search and recommendation systems [5, 10, 14, 45, 56]. This exploration spans diverse contexts, encompassing areas such as query auto-completion [21] and collaborative personalized search [51]. Within the NLP community, personalization has been a subject of exploration in various applications, including but not limited to dialogue agents [31, 39, 46, 49, 58, 60], review[27] and recipe generation [30], translation[50], headline generation [1], and classification tasks [13, 16], such as personalized sentiment analysis [33].

[1] https://github.com/LaMP-Benchmark/LaMP

With the emergence of LLMs and their application across various NLP tasks, Salemi et al. [43] proposed a retrieval-augmented approach for personalizing LLMs. They also introduced LaMP, a benchmark designed to assess the performance of personalized NLP models across diverse classification and short text generation tasks. The work by Li et al. [26] addresses a similar issue, focusing on personalized long text generation. Furthermore, Mysore et al. [35] assesses the capabilities of LLMs in the role of writing assistants. Various approaches have been explored for personalizing LLMs, encompassing techniques such as summarizing user profile [40], aligning language models with personalized human feedback [22], automatic prompt generation tailored to individual users [25], and incorporating long and short-term memory-based personalization strategies [57]. In this study, we adhere to the methodology outlined by Salemi et al. [43] and conduct experiments using the LaMP benchmark, focusing on training a component for retrieving personal information from user profile.

Retrieval Optimization in Retrieval-Augmented Generation. The optimization of retrieval models within the RAG (Retrieval-Augmented Generation) pipelines has emerged as a focal point in recent research, particularly in the context of question answering. Yang and Seo [52] focuses on distilling knowledge from the LM to the retriever by minimizing the KL-divergence between the LM's performance for each document in the retrieved set and the assigned score by the retriever to that document in the set. Additionally, Izacard and Grave [20] employs the attention weights of the LM to determine the importance of each document. This information is then utilized to distill knowledge from the LM to the retriever, aligning with the objectives set forth by Yang and Seo [52]. The approach presented by Wang et al. [47] involves the use of reinforcement learning, where the reward function is derived from the performance of the LM. In this work, we adopt methods similar to that of Wang et al. [47] and Yang and Seo [52], given the absence of relevance data in the LaMP benchmark. Notably, our work stands out as the pioneering effort in leveraging feedback from LLMs to train personalized retrievers for personalizing LLMs. Furthermore, in all the previously mentioned approaches, the language model is trained after/with the retrieval model. In contrast, our approach assumes the language model is frozen, and our focus is solely on optimizing the retrieval model.

Information Access with Multiple Retrieval Models. Combining rank lists generated by different retrievers has been extensively explored in the literature [9, 15, 29, 37]. However, the process of rank fusion presents challenges, especially when dealing with discrepancies in scoring scales among retrieval systems or the absence of overlapping documents in the ranked lists [29, 59]. Alternatively, methods for selecting specific retriever from a retriever pool for different datasets has been explored [23]. Furthermore, Arabzadeh et al. [3] investigates the optimal use of dense and sparse retrievers for each query, considering efficiency trade-offs. In our study, we concentrate on the performance-oriented selection of query-specific retrievers from a retriever pool.

3 NOTATIONS AND TASK FORMULATION

Generative language models often take an input x and generate the most probable sequence tokens y. This paper focuses on the task of personalized generation with the goal of generating outputs that are tailored for the preferences and characteristics of the language model user. Let $T = \{(u_1, x_1, y_1), (u_2, x_2, y_2), \cdots, (u_N, x_N, y_N)\}$ be a set of N training instances, each consisting of a user u, an input text x submitted by the user u, and the ground truth personalized output y. For each user u, a user profile P_u exists that can be employed for developing personalized generation models. A user profile P_u is a set of personal documents associated with the user u.

As discussed in Section 1, this paper focuses on retrieval augmented solutions for personalization, depicted in Figure 1. In such solutions, we first retrieve a set of personal documents from the user profile P_u. This is achieved through $L = \mathcal{R}(\phi_q(x); P_u)$ where ϕ_q is a query generation function that produces a search query string given the LLM input x and $\mathcal{R}$ is a retrieval model that retrieves personal documents from P_u given a query produced by ϕ_q. Hence, $\mathcal{R}$ returns a list of personal documents L. A prompt generation function ϕ_p is then applied to the LLM input and the retrieved result list as follows: $\phi_p(x, L)$. The constructed personalized prompt is then fed into a LLM M. The goal is to minimize the error between the generated output and the ground truth personalized output y. We assume that the LLM M is given and we do not aim at updating the LLM parameters for personalized generation. The rational behind this decision is that (1) fine-tuning M is often very expensive, and more importantly (2) the LLM M can memorize personal information if it is fine-tuned on data retrieved from the user's personal data P_u. Such memorization can put the user's privacy at risk. That said, this paper focuses on minimizing the personalized text generation error by solely updating the retrieval results L.

Section 4 studies methods for optimizing the retrieval model $\mathcal{R}_\theta$ parameterized by θ for updating the result list L. Section 5 extends Section 4 by exploring optimization solutions for retrieval model selection from a set of pre-defined retrieval models for updating the result list L.

4 LEARNING TO RETRIEVE FOR PERSONALIZING LLMS

Learning-to-rank (LTR) methods are often employed to train ranking models for search and recommendation [6]. For personalized LTR, the user's profile and long-term history are often utilized. Such personalized implicit feedback signals are often document-level and directly provided by the user, such as ratings, clicks, views, dwell time, and/or purchases [5, 56]. In the context of retrieval-augmented personalized text generation, user feedback manifests in the form of text written or edited by the user (i.e., the label y_i for each input x_i), taking into account the user preferences and interests. Therefore, accessing user feedback on a per-document basis within the user profile for training retrieval models is not feasible; neither is collecting document-level feedback through annotation, e.g., crowdsourcing. The main reason is that we do not know what documents serve the LLM best for generating personalized outputs for each input text. Therefore, learning to rank documents for LLM personalization is fundamentally different from developing personalized search or recommendation engines.

Considering these, we propose optimization methods that leverage feedback from the LLM itself, obtained by evaluating the impact of retrieved documents on the LLM performance for generating personalized outputs. Our first method uses the LLM performance (in terms of any arbitrary metric) to form a reward function and employs reinforcement learning [48] for optimization. Our second approach is based on knowledge distillation from the LLM to the retriever based on the LLM's performance in terms of personalized text generation. They are described in the subsequent subsections.

4.1 Retrieval Optimization for Personalized Generation using Reinforcement Learning

This section introduces ROPG-RL–a reinforcement learning approach that encourages the retrieval model to produce rankings that lead to more accurate personalized generation. We utilize the vanilla policy gradient optimization algorithm, drawing upon rewards supplied by the downstream LLM, as depicted in Figure 2 (a). In this approach, we establish a parameterized policy (here, the retrieval model) that assigns a probability to each action (specifically, selecting personal documents to be fed to the LLM). Subsequently, we formulate a reward function, well aligned with the goal of personalized text generation, that the parameterized policy aims to maximize. This approach enables us to effectively refine and improve the performance of the retrieval model. The specifics of our methodology will be discussed in the subsequent paragraphs.

Parameterized Policy (π_θ). In this context, defining the policy function necessitates a clear delineation of the states and actions applicable within the scope of this study. In this formulation, *the policy is parameterized through the retrieval model.* Essentially, given a query, the retrieval model undergoes training to assign a higher probability to the documents from the user profile that are deemed more valuable for personalizing the LLM. Here, an action is considered as selecting a document given a query.[2] The state, on the other hand, corresponds to the given query itself. Meaning that the goal is to update the policy such that it produces more effective results for personalization. In this study, we restrict our focus to trajectories comprising a single state. This implies that the model initiates from the initial state, executes a singular action, and concludes the trajectory. The probability of each action is computed using the following formula:

$$\pi_\theta(d|x) = \frac{\exp\left(\mathcal{R}_\theta(\phi_q(x), d)\right)}{\sum_{d' \in P_u} \exp\left(\mathcal{R}_\theta(\phi_q(x), d')\right)} \quad : \quad \forall d \in P_u \qquad (1)$$

where ϕ_q is the query generation function as explained in Section 3, $\mathcal{R}_\theta$ denotes the retrieval model parameterized by θ, and P_u is a set of documents containing user's personal data (see Section 3). The probability assigned by the policy model to any document that is not in the user profile would be zero. Note that the user profile can grow over time that leads to inefficient calculation of policy function. To address this efficiency issues, we can approximate the policy function, either using hierarchical softmax, similar to [32, 34, 53], or by marginalization through top l approximation. Without loss

[2]We also explored scenarios where multiple documents are sampled without replacement from the policy network, however, we observed no or little improvement compared to a much simpler and more efficient approach where only one document is sampled for updating the policy parameters.

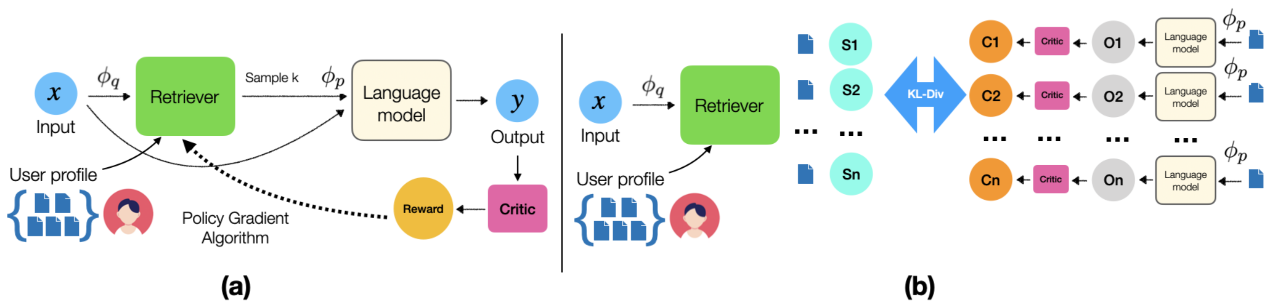

Figure 2: Overview of training dense retrievers for personalizing LLMs using LLMs feedback with policy gradient optimization (a) and knowledge distillation (b). ϕ_q represents the query generation function, ϕ_p is the prompt generation function, and "Critic" denotes the evaluation metric employed for the personalized task.

of generality we choose the second approach and compute the policy function based on P_u^l, a set of l documents from P_u that achieve highest retrieval scores according to our initial retrieval weights. In our experiments, we set $l = 16$. Of course, leveraging the complete user profile during training could potentially yield superior performance, albeit at the cost of increased training time.

Reward Function (R). Defining trajectories involving the selection of multiple documents from the user profile, with rewards for each, is computationally intensive. This is due to the need to compute rewards using the LLM for every sampled set of documents from the user profile. On the contrary, if we limit trajectories to selecting only one document from the user profile, we can precompute the rewards for each document in the profile. Therefore, we only consider trajectories with one action. Moreover, to expedite the learning process and minimize variance, we subtract a previously calculated evaluation score from the effectiveness of the current sampled document. The reward function is then defined as:

$$\textsc{Reward}(d; x, y) = \quad \textsc{Eval}\left(y, M(\phi_p(x, [d]))\right) - \textsc{Eval}\left(y, M(\phi_p(x, [d_b]))\right) \tag{2}$$

where d is a sampled document from the user profile P_u using the parameterized policy π_θ and d_b is the document selected by the policy with initial weighting (i.e., the retrieval model prior to fine-tuning with RL). M is the LLM that generates personalized text based on the given personalized prompt ϕ_p. Eval denotes an arbitrary metric for evaluating personalized text generation.

Training Objective (J). The objective in ROPG-RL is to maximize the expected reward obtained by the parameterized policy of the retriever (π_θ). To achieve this objective, we employ a gradient ascent algorithm with the update rule of $\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\pi_\theta)$. For each mini-batch $B \subset T$, the objective function is presented below:

$$\arg\max_\theta \frac{1}{|B|} \sum_{(u,x,y)\in B} \mathop{\mathbb{E}}_{d\sim\pi_\theta} \left[\textsc{Reward}(d; x, y) \log \pi_\theta(d|x)\right] \tag{3}$$

4.2 Retrieval Optimization for Personalized Generation using Knowledge Distillation

An alternative approach for training a retrieval model with feedback from the LLM involves knowledge distillation from the LLM to the retrieval model. Contrary to ROPG-RL, this approach, called ROPG-KD, considers the relative usefulness of different items in the user profile for the LLM on performing the downstream personalized task. Indeed, this approach endeavors to allocate a higher probability to items that are more useful than others for the LLM.

Conversely, ROPG-RL only considers the impact of each (or possibly a subset of documents grouped together in trajectories with more than one action) on the final score that the LLM achieves. Hence, it seeks to reward the model for actions that yield positive outcomes and penalize for actions that result in negative consequences. This implies that where there are no favorable actions, the model would face punishment; however, this is not the case with knowledge distillation. On the other hand, RL optimization processes tend to be less stable and are more susceptible to overfitting.

Considering all the aforementioned aspects, we propose an alternative approach based on knowledge distillation. In the context of knowledge distillation, the primary objective is to encourage the retriever model to assign higher similarity scores to the documents from the user profile that are more useful for the language model in fulfilling its task. Figure 2 (b) illustrates the pipeline for this approach. To accomplish this objective, we employ Equation (1) to allocate a probability to individual elements within the user profile. Subsequently, we use the following function to produce the target probability distribution:

$$p_t(d|x) = \frac{\exp\left(\textsc{Eval}\left(y, M(\phi_p(x, [d]))\right)\right)}{\sum_{d'\in P_u} \exp\left(\textsc{Eval}\left(y, M(\phi_p(x, [d']))\right)\right)} \quad : \quad \forall d \in P_u$$

where Eval is an arbitrary metric for evaluating personalized text generation models and M denotes the LLM being used. Similar to ROPG-RL, for efficiency purposes, we approximate the distribution p_t by only focusing on the top l retrieved document w.r.t. the initial retrieval parameters. Inspired by previous work on knowledge distillation in IR [52], for each mini-batch $B \subset T$, we minimize the following loss function based on KL-divergence:

$$\arg\min_\theta \frac{1}{|B|} \sum_{(u,x,y)\in B} \sum_{d\in P_u^l} p_t(d|x) \log \frac{\pi_\theta(d|x)}{p_t(d|x)} \tag{4}$$

4.3 Retrieval Model Architecture

The proposed optimization solutions can be applied to any neural ranking model. Without loss of generality, we use them to train dense retrieval models. We adopt Contriever [19], a pre-trained bi-encoder model for dense retrieval. Contriever encodes the query and

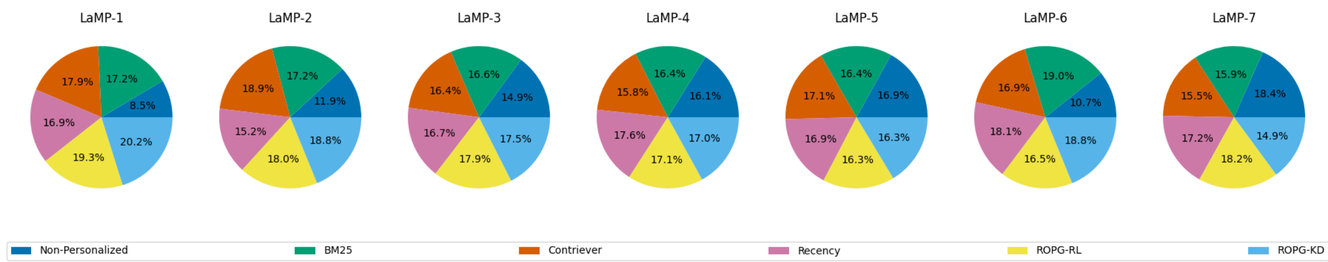

Figure 3: Relative winning rate for each selected retrieval model. When multiple retrieval models get the highest score, we consider all of them with the highest score as the winner.

document text using an encoder with shared parameters and applies dot product to compute the relevance score. After training, an exact or approximate nearest neighbor (kNN) algorithm is used to index the learned document representations for each user profile. We use exact kNN in our experiments. During inference, each document is scored independently and the scored documents are sorted in descending order with respect to their score.

5 RETRIEVAL MODEL SELECTION FOR PERSONALIZING LLMS

We hypothesize that there are multiple aspects to LLM personalization and each existing retrieval model does not address all of them. For instance, retrieving recent user interactions may lead to the highest personalized text generation performance for an input, while a keyword-matching model or a semantic matching model may be optimal for another input. To validate this, we utilize the LaMP benchmark–a recent benchmark consisting of seven diverse tasks for training and evaluating personalized LLMs [43]. Statistics of these datasets are presented in Table 1. More information on the LaMP benchmark is provided in Section 6.1. We evaluate personalization of an 11B parameter FlanT5-XXL [8] using the following retrieval augmentation approaches: (1) no personalization (i.e., no retrieval augmentation), (2) term matching retrieval using BM25 [41], (3) zero-shot semantic matching model using Contriever [19], (4) ranking documents in the user profile based on recency, and (5 & 6) the proposed retrieval models for LLM personalization–ROPG-RL and ROPG-KD. Figure 3 illustrates the winning rate achieved by each of these models when retrieving from the user profile. To compute the winning rate, we first evaluate the recommended evaluation metric for each of the datasets by the LaMP benchmark (i.e., accuracy for categorical classification, MAE for ordinal classification, and Rouge-1 for text-generation tasks). For each retrieval model, we then count the number of inputs for which it achieves the highest personalized text generation performance among the mentioned six models. If two or more retrieval models achieve the same and the highest performance, they all get rewarded. The count is then normalized to estimate the winning rate.

As evident in Figure 3, there is no consistent winner across all tasks in the LaMP benchmark. For instance, between the best text generation performance for 8.5% to 18.4% of the inputs across datasets can be achieved when no personalization is conducted. Recency-based ranking can lead to the best performance for 15.2% to 18.1% of the inputs depending on the dataset. Even the proposed ROPG-RL and ROPG-KD models provide the highest performance for 14.9% to 20.2% of the inputs across different datasets in LaMP.

Given these observations, we hypothesize that selecting what ranking function to use for each input, or even when to apply personalization, can improve end-to-end personalized text generation performance. According to this hypothesis and inspired by the query performance prediction literature [7, 42, 44, 54], we introduce two retriever selection models. In the first approach, referred to as RSPG-Pre, we retrieve items from the user profile using each retriever in the retriever pool to construct personalized prompts. These constructed prompts are then fed into RSPG-Pre for selection and consumption by the LLM. In the second approach, termed RSPG-Post, the personalized prompts produced by all retrieval models are also presented to the LLM, and the resulting outputs, along with the original prompts, are fed into RSPG-Post for retrieval model selection. These models are presented below.

Optimizing Retriever Selection Models. The training pipeline for retrieval selection is illustrated in Figure 4. Let $\mathbf{R}$ be a set of retrieval models in the pipeline and $\mathcal{S}_\omega$ be a retrieval selection model parameterized by ω that produces a selection score for each retrieval model in $\mathbf{R}$. We use a knowledge distillation loss from the downstream LLM performance to train the retrieval selection model. For this purpose, the target selection probability distribution over retrieval models in $\mathbf{R}$ for an input (u, x, y) is computed as:

$$p_{TS}(\mathbf{R}_i|x; u, y) = \frac{\exp\left(\textsc{Eval}\left(y, M(\phi_p(x, \mathbf{R}_i(\phi_q(x); P_u)))\right)\right)}{\sum_{j=1}^{|\mathbf{R}|} \exp\left(\textsc{Eval}\left(y, M(\phi_p(x, \mathbf{R}_j(\phi_q(x); P_u)))\right)\right)} \quad (5)$$

where Eval is an arbitrary evaluation metric for personalized text generation, and M is the LLM used for text generation. The retrieval selection model probability for the retriever $\mathbf{R}_i \in \mathbf{R}$ is calculated as:

$$p_{S_\omega}(\mathbf{R}_i|x) = \frac{\exp(\mathcal{S}_\omega(\mathbf{R}_i, x))}{\sum_{j=1}^{|\mathbf{R}|} \exp(\mathcal{S}_\omega(\mathbf{R}_j, x))} \quad (6)$$

For a mini-batch $B \subset T$, we use KL-divergence loss as follows to train the retrieval selection models:

$$\frac{1}{|B|} \sum_{(u,x,y)\in B} \sum_{i=1}^{|\mathbf{R}|} p_{TS}(\mathbf{R}_i|x; u, y) \log \frac{p_{S_\omega}(\mathbf{R}_i|x)}{p_{TS}(\mathbf{R}_i|x; u, y)} \quad (7)$$

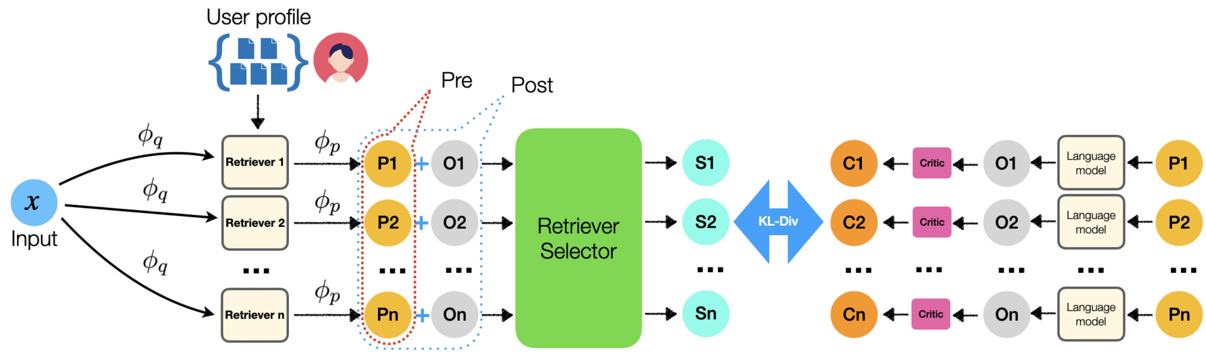

Figure 4: Pipeline for training the RSPG models. We minimize the KL-divergence between the scores generated for each prompt using RSPG and the performance of the prompt used for the evaluation of the LLM in a personalized task.

Table 1: Statistics of the datasets within the LaMP benchmark [43] with time-based data separation.

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	-

Selection Model Architecture. To select a retrieval model for the given input, we initiate the process by employing all retrieval models (see Figure 3 for the list of retrieval models in our experiments) to retrieve relevant documents. As mentioned earlier, we envision two categories of retrieval selection models: pre-generation and post-generation models. We use an encoder-only model for estimating a selection score for each retrieval model. For the pre-generation scenario, the input to the encoder is the LLM prompt constructed by each retrieval model: $\phi_p(x, \mathbf{R}_i(\phi_q(x); P_u))$. For the post-generation scenario, this prompt is concatenated with the LLM's output text given this prompt. The encoder's final layer representation is then fed into a linear projection layer for producing a scalar value as the selection score. Given the long length of prompts in retrieval selection, we use Longformer [4] as the encoder in our retrieval selection model.[3] Once all retrieval models are scored using this approach, we choose the one with the highest selection score and feed the corresponding prompt to the LLM.

6 EXPERIMENTS

6.1 Experimental Setup

Datasets. We adopt the LaMP benchmark [43]–a public benchmark that encompasses a diverse set of personalized text generation tasks.[4] Specifically, the benchmark comprises three personalized text classification tasks and four personalized text generation tasks. They include (1) Personalized Citation Identification (binary classification), (2) Personalized Movie Tagging (categorical classification with 15 classes), (3) Personalized Product Rating (ordinal classification from 1 to 5-star rating for e-commerce products), (4) Personalized News Headline Generation, (5) Personalized Scholarly Title Generation, (6) Personalized Email Subject Generation, and (7) Personalized Tweet Paraphrasing.

We use the time-based separation setting offered by LaMP for data splitting. In this setting, the data for each user is split into train, development, and test sets based on their timestamp, modeling a real-world scenario in which personalized outputs for test inputs are generated using the personal documents created earlier by that user. The reason behind opting for a time-based separation in the LaMP benchmark is to investigate the impact of recency in our experiments. Table 1 reports the statistics of the datasets.

Evaluation Metrics. Following Salemi et al. [43], we evaluate LaMP-1 using Accuracy, LaMP-2 using Accuracy and F1-measure, and LaMP-3 using mean absolute error (MAE) and root mean squared error (RMSE). We use ROUGE-1 and ROUGE-L [28] to evaluate text generation performance on text generation datasets (LaMP-4 to LaMP-7). Statistically significant differences are identified using two-tailed paired t-test for ROUGE-1/ROUGE-L/MAE/RMSE and McNemar test for Accuracy/F1.

Training Configurations. An integral part of our training pipeline is the evaluation function Eval. We use the standard metrics suggested by the LaMP benchmark for each dataset to implemented the Eval function. In more detail, we measure accuracy for the binary and categorical classification tasks (LaMP-1 and LaMP-2) and ROUGE-1 [28] for the text generation tasks (LaMP-4, LaMP-5, LaMP-6, and LaMP-7). Given that the evaluation metric for LaMP-3 is MAE (Mean Absolute Error), where lower values are preferable, we need to adapt the evaluation function to use it as a reward

[3] https://huggingface.co/allenai/longformer-base-4096
[4] The LaMP benchmark is available at https://lamp-benchmark.github.io/.

Table 2: Templates used to create prompt to augment the input for LLM with the retrieved items from the user profile (i.e., ϕ_p). Function concat concatenates the strings in its first argument by placing the second argument between them. Function add_to_paper_title adds the string in its first argument to the paper's title in the LaMP-1 task. Function PPEP creates the prompt for each retrieved item from the profile. [INPUT] is the task's input (x).

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

Table 3: The performance of our methods and the baselines on the LaMP benchmark. For all metrics, the higher values the better, except for RMSE and MAE which are used in LaMP-3. In this table, the superscript ^{1}, ^{2}, ^{3}, ^{4}, and ^{5} indicate significant improvement over No Personalization, BM25, Recency, Contriever, and RRF, respectively ($p < 0.05$).

Dataset	Metric	No Personalization	Personalization Baselines				Our Methods			
			BM25	Recency	Contriever	RRF	ROPG-RL	ROPG-KD	RSPG-Pre	RSPG-Post
LaMP-1: Personalized Citation Identification	Accuracy ↑	0.502	0.626	0.622	0.636	0.570	0.655^{12345}	0.668^{12345}	0.663^{12345}	$\mathbf{0.672}^{12345}$
LaMP-2: Personalized Movie Tagging	Accuracy ↑	0.359	0.387	0.377	0.396	0.375	0.391^{135}	0.396^{135}	0.405^{1235}	$\mathbf{0.430}^{12345}$
	F1 ↑	0.276	0.306	0.295	0.304	0.299	0.300^{135}	0.306^{135}	0.314^{1235}	$\mathbf{0.339}^{12345}$
LaMP-3: Personalized Product Rating	MAE ↓	0.308	0.298	0.296	0.299	0.314	0.286^{145}	0.290^{15}	0.282^{12345}	$\mathbf{0.264}^{12345}$
	RMSE ↓	0.611	0.611	0.605	0.616	0.614	0.591^{145}	0.604^{15}	0.585^{12345}	$\mathbf{0.568}^{12345}$
LaMP-4: Personalized News Headline Generation	ROUGE-1 ↑	0.176	0.186	0.189	0.183	0.190	0.191^{1}	0.187^{1}	0.190^{1}	$\mathbf{0.203}^{12345}$
	ROUGE-L ↑	0.160	0.171	0.173	0.169	0.176	0.177^{1}	0.172^{1}	0.176^{1}	$\mathbf{0.186}^{12345}$
LaMP-5: Personalized Scholarly Title Generation	ROUGE-1 ↑	0.478	0.477	0.475	**0.483**	0.478	0.475	0.477	$\mathbf{0.483}^{1235}$	0.480
	ROUGE-L ↑	0.428	0.427	0.426	**0.433**	0.428	0.427	0.428	0.431^{1235}	0.429
LaMP-6: Personalized Email Subject Generation	ROUGE-1 ↑	0.335	0.412	0.403	0.401	0.394	0.394	0.415^{1345}	0.426^{12345}	$\mathbf{0.433}^{12345}$
	ROUGE-L ↑	0.319	0.398	0.389	0.386	0.381	0.381	0.400^{1345}	0.411^{12345}	$\mathbf{0.418}^{12345}$
LaMP-7: Personalized Tweet Paraphrasing	ROUGE-1 ↑	0.449	0.446	0.444	0.440	0.446	0.448^{4}	0.441	0.450^{2345}	$\mathbf{0.461}^{12345}$
	ROUGE-L ↑	0.396	0.394	0.393	0.390	0.395	0.397^{4}	0.391	0.400^{2345}	$\mathbf{0.409}^{12345}$

function. The modification is as follows:

$$\textsc{Eval}_{\text{LaMP-3}}(y, \hat{y}) = \frac{\max(|1-y|, |5-y|) - \text{MAE}(y, \hat{y})}{\max(|1-y|, |5-y|)} \quad (8)$$

where $\hat{y}$ is the prediction and y is the target output. This reward function normalizes the MAE score by measuring its distance from the worst score achievable based on the model's prediction.[5] In this scenario, a correct prediction by the model results in a score of 1, while in the worst-case prediction, it receives a score of 0.

In this paper, we use the Adam optimizer [24] with a learning rate of 10^{-5}. We dedicate 5% of the training steps to warmup with a linear scheduler. We also use gradient clipping with the value of 1. To accommodate the task requirements, we set the maximum input and output lengths to 512 tokens for LLMs following Salemi et al. [43]. However, we use the maximum input length of 1024 for retrieval selection in order to incorporate a prompt and the corresponding output from the LLM. We train the retrieval models for 10 and the retriever selection models for 20 epochs. In all experiments, following Salemi et al. [43], we utilize FlanT5-XXL[8]–an instruction-tuned open-source LLM with 11B parameters. We use a beam size of 4 in beam search for text generation [18]. The effective batch size in all experiments is set to 64 (8 accumulation steps with batch size 8). We performed all the experiments on a single A100 Nvidia GPU with 80GB memory and 128GB of RAM.

In all experiments, following Salemi et al. [43], we use the non-template parts of the LLM input x as the query for personal document retrieval. We followed Salemi et al. [43] for prompt templates (i.e., ϕ_p) for each dataset in LaMP. They are listed in Table 2. In all experiments, the process of creating personalized prompts for evaluating models involves retrieving four items from the user profile. In crafting documents from each user profile, we adhere to the approach established in [43], appending the date of the document to it. The date is prefixed with *date: [date]*. For implementing BM25, we use the `rank_bm25` library.[6] All the neural models in this paper are implemented using the PyTorch library [38].

***Baseline Methods*.** We compare the proposed approaches against the following retrieval models for personalized text generation.

- **No Personalization:** We employ FlanT5-XXL [8] as a non-personalized baseline. In this baseline, the model is presented with the original task's input without any modification.

[5] According to the implementation of the LaMP benchmark, the worst score attainable by a model in the LaMP-3 task is denoted as $\max(|1-y|, |5-y|)$.

[6] https://github.com/dorianbrown/rank_bm25

- **Personalized Baselines:** Following Salemi et al. [43], we utilize BM25 [41], Recency, and Contriever [19] to retrieve items from the user profile for LLM personalization. There are, of course, many other neural ranking models that may outperform these baselines on some retrieval benchmarks. However, it is important to note that our optimization approaches can be applied to any neural ranking model, including any missing baseline from this list. That being said, we do not aim at comparing different model architectures, instead we aim at demonstrating the impact of our optimization methods. Note that no other retrieval models have ever been used on LaMP and, to the best of our knowledge, this list consists of all methods in the retrieval-augmented LLM personalization literature. Furthermore, we apply Reciprocal Rank Fusion (RRF) [9] to integrate the retrieval lists generated by all the retrievers. This fusion-based approach is employed for comparison with our retriever selection method. Subsequently, we employ these retrieved items to formulate a personalized input prompt for FlanT5-XXL.

6.2 Empirical Results

This section provides empirical evidence to answer research questions that shed light into the proposed approaches in this paper.

How does personalization using the proposed approaches impact text generation performance? To answer this question, we compare our methods with the non-personalized baseline, i.e., FlanT5-XXL without augmentation with personal information. Table 3 presents the results on the LaMP benchmark. LLM Personalization using both ROPG-RL and ROPG-KD improves the performance on LaMP-1, LaMP-2, LaMP-3, LaMP-4, and LaMP-6. After applying the retrieval model selection methods (RSPG-Pre and RSPG-Post), the non-personalized model is beaten on all datasets and in terms of all metrics. The performance gains are statistically significant in almost all cases. This is an important finding in the sense that no personalized baseline could perform better than a non-personalized LLM on LaMP-7, while RSPG-Pre and RSPG-Post demonstrate that personalized LLMs can ultimately demonstrate performance gain on these datasets. This finding also suggests the impact of retrieval augmentation for the purpose of LLM personalization.

How do ROPG optimization algorithms impact text generation performance? To answer this, we must compare ROPG-RL and ROPG-KD with the Contriever baseline, since they are initialized with the Contriever model and fine-tuned using our proposed ROPG-RL and ROPG-KD algorithms. The results in Table 3 suggest that applying ROPG-RL to Contriever yields performance gain on LaMP-1, LaMP-3, LaMP-4, and LaMP-7. ROPG-KD additionally outperforms Contriever on LaMP-6. Notably ROPG-KD performs better than ROPG-RL on the tasks with binary feedback from the language model (LaMP-1 and LaMP-2). Comparing ROPG-RL and ROPG-KD suggests there is no clear winner among them. This once again attests that each personalization task have different requirements, thus motivating the need for retrieval selection in retrieval-augmented LLM personalization.

How effective are the proposed retrieval selection methods? To assess the efficacy of retriever selection, we measure its success rate in selecting the best performing retriever in the retriever pool **R** for each input. Note that for inputs with multiple best-performing retrieval models, a selection is considered successful if any of them is selected. In this experiment, we incorporated several unsupervised Query Performance Prediction (QPP) methods, including WIG [61], NQC [12], and $\sigma_{\max}$ and $\sigma_{x\%}$ [11], to perform a comparative analysis with our proposed method. To accomplish this, we assign the task to each QPP approach of providing a score to the retrieved results for each retriever. The retriever with the highest score is then selected for that particular input. It is important to note that since Recency does not provide score for retrieved results, we consider the reciprocal rank of each document as its score. In no personalization retriever, we assign a score of zero to all items in the profile. It is crucial to highlight that the utilization of supervised QPP methods, such as BERT-QPP [2], was not feasible due to their reliance on query-document relevance labels, which are unavailable in our datasets.

Table 4: The success rate of models in selecting the best performing retriever for each input. The superscript * indicates significant improvement over the best baseline ($p < 0.05$).

Dataset	Success Rate					
	WIG	NQC	$\sigma_{\max}$	$\sigma_{50\%}$	RSPG-Pre	RSPG-Post
LaMP-1: Personalized Citation Identification	0.858	0.824	0.848	0.847	0.865*	**0.874***
LaMP-2: Personalized Movie Tagging	0.908	0.918	0.910	0.910	0.936*	**0.962***
LaMP-3: Personalized Product Rating	0.896	0.890	0.896	0.894	0.903	**0.920***
LaMP-4: Personalized News Headline Generation	0.401	0.404	0.394	0.398	0.401	**0.447***
LaMP-5: Personalized Scholarly Title Generation	0.562	0.572	0.562	0.557	**0.600***	0.577
LaMP-6: Personalized Email Subject Generation	0.613	0.606	0.614	0.617	0.633*	**0.641***
LaMP-7: Personalized Tweet Paraphrasing	0.851	0.840	0.852	0.851	0.860	**0.898***

The results in Table 4 demonstrate the superior performance of both the RSPG-Pre and RSPG-Post across nearly all datasets. Specifically, RSPG-Post achieves a significant improvement over all baselines for the all datasets, except for the LaMP-5 dataset. In this dataset, RSPG-Pre shows a significant improvement compared to the baselines. Likewise, RSPG-Pre outperforms all baselines in all datasets except LaMP-4, with significant improvements observed in LaMP-1, LaMP-2, LaMP-5, and LaMP-6. Overall, the outcomes of this experiment suggest that the proposed approach for retrieval model selection consistently outperforms the baselines.

The results also indicate that for all classification datasets (i.e., LaMP-1, LaMP-2, and LaMP-3) and LaMP-7 for generation, both pre- and post-generation models achieve a success rate of over 80%. This suggests that the model has to some extent successfully learned to choose the most suitable retriever for each input. Conversely, for the remaining text generation datasets (i.e., LaMP-4, LaMP-5, and LaMP-6), the accuracy is lower, ranging from 40% to 65%. We attribute this to the inherent complexity of text generation tasks compared to text classification. Comparing RSPG-Pre and RSPG-Post, the latter exhibits higher success rate in all tasks except LaMP-5. This suggests that employing the generated output in retrieval selection can have a substantial impact on the performance.

Looking back to the results in Table 3, the post-generation retrieval selection model (RSPG-Post) performs better than the pre-generation selection model (RSPG-Pre) on six out of seven datasets;

Table 5: Studying the impact of ROPG algorithms on the end-to-end performance of our pipeline with retrieval model selection. In this table, the superscript * indicates significant improvement over w/o ROPG approach ($p < 0.05$).

Dataset	Metric	RSPG-Pre		RSPG-Post		Oracle	
		w/o ROPG	w/ ROPG	w/o ROPG	w/ ROPG	Lower-bound	Upper-bound
LaMP-1: Personalized Citation Identification	Accuracy ↑	0.646	0.663*	0.644	0.672*	0.381	0.798
LaMP-2: Personalized Movie Tagging	Accuracy ↑	0.403	0.405	0.425	0.430	0.296	0.468
	F1 ↑	0.311	0.314	0.330	0.339	0.216	0.380
LaMP-3: Personalized Product Rating	MAE ↓	0.287	0.282	0.269	0.264	0.450	0.181
	RMSE ↓	0.595	0.585	0.577	0.568	0.772	0.462
LaMP-4: Personalized News Headline Generation	ROUGE-1 ↑	0.189	0.190	0.191	0.203*	0.103	0.269
	ROUGE-L ↑	0.174	0.176	0.177	0.186*	0.098	0.243
LaMP-5: Personalized Scholarly Title Generation	ROUGE-1 ↑	0.478	0.483*	0.475	0.480	0.388	0.548
	ROUGE-L ↑	0.428	0.431*	0.427	0.429	0.349	0.492
LaMP-6: Personalized Email Subject Generation	ROUGE-1 ↑	0.426	0.426	0.394	0.433*	0.239	0.511
	ROUGE-L ↑	0.412	0.411	0.381	0.418*	0.228	0.492
LaMP-7: Personalized Tweet Paraphrasing	ROUGE-1 ↑	0.449	0.450	0.448	0.461*	0.410	0.470
	ROUGE-L ↑	0.398	0.400	0.397	0.409*	0.361	0.417

LaMP-5 is the exception, which is explained by the results in Table 4. RSPG-Pre consistently outperforms the baselines significantly in all classification tasks (LaMP-1, LaMP-2, and LaMP-3) as well as in LaMP-6. In the remaining datasets, RSPG-Pre achieves comparable or superior results to the baselines, although the differences are not statistically significant. Finally, RSPG-Post leads to the best personalized text generation performance on six out of seven datasets.

What is the method that results in the highest personalized text generation performance? According to Table 3, RSPG-Post performs best on six out of seven datasets (all but LaMP-5). Contriever, on the other hand, demonstrates the highest performance on LaMP-5. The performance gains by RSPG-Post on all the remaining six datasets are statistically significant, according to a two-tailed paired t-test for generation and ordinal classification datasets and McNemar test for binary and categorical text classification datasets.

What is the impact of ROPG algorithms on retrieval selection results? To investigate the impact of training personalized retrievers using the proposed ROPG algorithms (i.e., both ROPG-RL and ROPG-KD) on the end-to-end performance of the pipeline with retrieval selection, we conduct an analysis by excluding the fine-tuned retrievers from the retriever pool **R**. The outcomes of this experiment along with the Oracle performance (both lower and upper bound for retrieval selection) are reported in Table 5. The results indicate that the models without ROPG in its retriever pools achieve lower performance on all datasets in both pre- and post-generation settings. The only exception is the RSPG-Pre results on LaMP-6, where including ROPG algorithms does not make any significant impact. This suggests that our ultimate performance gain is not just because of the retrieval selection models; Instead, the retrieval optimization approaches presented in Section 4 are effective in enhancing the end-to-end performance of the pipeline.

Finally, a comparison between our results and the Oracle performance in Table 5 provides insight into the potential for further improvements in retrieval selection. For instance, in LaMP-3 and LaMP-4, our best performing method only achieves 68.3% and 75.4% of the Oracle's upper-bound, respectively. This suggests that there are still substantial room for improvement. However, the corresponding numbers for LaMP-1, LaMP-2, LaMP-5, LaMP-6, and LaMP-7 are 84.2%, 91.8%, 88.1%, 84.7%, and 98.0%, respectively, suggesting that our best performing retrieval selection method is performing very close to the upper-bound performance.

7 CONCLUSIONS AND FUTURE WORK

This paper explored personalization of LLMs through a retrieval augmentation pipeline with a focus on optimizing the retrieval component. We introduced two solutions for optimizing ranking models by soliciting personalized feedback from the language model, one based on reinforcement learning where the reward function is defined based on the personalized text generation quality, and another based on knowledge distillation from the language model to the retrieval model. Subsequently, we observed that personalization tasks can benefit from different retrieval models, depending on specific needs and requirements. Given this observation, we developed a pre-generation and a post-generation retriever selection model. Evaluation on seven diverse personalization tasks from the LaMP benchmark showed that our proposed methods outperform competitive baselines on six out of seven datasets with statistically significant improvements. Through careful ablation studies, we demonstrate the impact of each component used our pipeline.

In this work, we solely focused on optimizing and selecting the ranking models for LLM personalization. One limitation of this work lies in the use of static templates for generating prompts for the LLM. In the future, we aim at optimizing the prompt generation component using the feedback obtained from the downstream LLM performance. In addition, all datasets in the LaMP benchmark focus on short text generation tasks. We will explore personalized long text generation methods in the future.

ACKNOWLEDGMENT

This work was supported in part by the Center for Intelligent Information Retrieval, in part by Lowe's, in part by an Amazon Research Award, Fall 2022 CFP, in part by an award from Google, and in part by and award from 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

[1] Xiang Ao, Xiting Wang, Ling Luo, Ying Qiao, Qing He, and Xing Xie. 2021. PENS: A Dataset and Generic Framework for Personalized News Headline Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 82–92. https://doi.org/10.18653/v1/2021.acl-long.7

[2] Negar Arabzadeh, Maryam Khodabakhsh, and Ebrahim Bagheri. 2021. BERT-QPP: Contextualized Pre-trained transformers for Query Performance Prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 2857–2861. https://doi.org/10.1145/3459637.3482063

[3] Negar Arabzadeh, Xinyi Yan, and Charles L. A. Clarke. 2021. Predicting Efficiency/Effectiveness Trade-Offs for Dense vs. Sparse Retrieval Strategy Selection. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 2862–2866. https://doi.org/10.1145/3459637.3482159

[4] Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. arXiv:2004.05150 [cs.CL]

[5] Paul N. Bennett, Ryen W. White, Wei Chu, Susan T. Dumais, Peter Bailey, Fedor Borisyuk, and Xiaoyuan Cui. 2012. Modeling the Impact of Short- and Long-Term Behavior on Search Personalization. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Portland, Oregon, USA) *(SIGIR '12)*. Association for Computing Machinery, New York, NY, USA, 185–194. https://doi.org/10.1145/2348283.2348312

[6] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to Rank: From Pairwise Approach to Listwise Approach. In *Proceedings of the 24th International Conference on Machine Learning* (Corvalis, Oregon, USA) *(ICML '07)*. Association for Computing Machinery, New York, NY, USA, 129–136. https://doi.org/10.1145/1273496.1273513

[7] D. Carmel and E. Yom-Tov. 2010. *Estimating the Query Difficulty for Information Retrieval* (1st ed.). Morgan and Claypool Publishers.

[8] Hyung Won Chung et al. 2022. Scaling Instruction-Finetuned Language Models. arXiv:2210.11416 [cs.LG]

[9] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Boston, MA, USA) *(SIGIR '09)*. Association for Computing Machinery, New York, NY, USA, 758–759. https://doi.org/10.1145/1571941.1572114

[10] Bruce W. Croft, Stephen Cronen-Townsend, and Victor Lavrenko. 2001. Relevance Feedback and Personalization: A Language Modeling Perspective. In *DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries*. http://citeseer.ist.psu.edu/453602.html

[11] Ronan Cummins, Joemon Jose, and Colm O'Riordan. 2011. Improved query performance prediction using standard deviation. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Beijing, China) *(SIGIR '11)*. Association for Computing Machinery, New York, NY, USA, 1089–1090. https://doi.org/10.1145/2009916.2010063

[12] Suchana Datta, Debasis Ganguly, Mandar Mitra, and Derek Greene. 2022. A Relative Information Gain-based Query Performance Prediction Framework with Generated Query Variants. *ACM Trans. Inf. Syst.* 41, 2, Article 38 (dec 2022), 31 pages. https://doi.org/10.1145/3545112

[13] Shiran Dudy, Steven Bedrick, and Bonnie Webber. 2021. Refocusing on Relevance: Personalization in NLG. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 5190–5202. https://doi.org/10.18653/v1/2021.emnlp-main.421

[14] Susan T. Dumais. 2016. Personalized Search: Potential and Pitfalls. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management* (Indianapolis, Indiana, USA) *(CIKM '16)*. Association for Computing Machinery, New York, NY, USA, 689. https://doi.org/10.1145/2983323.2983367

[15] Mohamed Farah and Daniel Vanderpooten. 2007. An Outranking Approach for Rank Aggregation in Information Retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Amsterdam, The Netherlands) *(SIGIR '07)*. Association for Computing Machinery, New York, NY, USA, 591–598. https://doi.org/10.1145/1277741.1277843

[16] Lucie Flek. 2020. Returning the N to NLP: Towards Contextually Personalized Classification Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 7828–7838. https://doi.org/10.18653/v1/2020.acl-main.700

[17] Andrew Fowler, Kurt Partridge, Ciprian Chelba, Xiaojun Bi, Tom Ouyang, and Shumin Zhai. 2015. Effects of Language Modeling and Its Personalization on Touchscreen Typing Performance. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 649–658. https://doi.org/10.1145/2702123.2702503

[18] Markus Freitag and Yaser Al-Onaizan. 2017. Beam Search Strategies for Neural Machine Translation. In *Proceedings of the First Workshop on Neural Machine Translation*, Thang Luong, Alexandra Birch, Graham Neubig, and Andrew Finch (Eds.). Association for Computational Linguistics, Vancouver, 56–60. https://doi.org/10.18653/v1/W17-3207

[19] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised Dense Information Retrieval with Contrastive Learning. *Transactions on Machine Learning Research* (2022). https://openreview.net/forum?id=jKN1pXi7b0

[20] Gautier Izacard and Edouard Grave. 2021. Distilling Knowledge from Reader to Retriever for Question Answering. In *International Conference on Learning Representations*. https://openreview.net/forum?id=NTEz-6wysdb

[21] Aaron Jaech and Mari Ostendorf. 2018. Personalized Language Model for Query Auto-Completion. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Melbourne, Australia, 700–705. https://doi.org/10.18653/v1/P18-2111

[22] 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. *ArXiv* abs/2310.11564 (2023). https://api.semanticscholar.org/CorpusID:264289231

[23] Ekaterina Khramtsova, Shengyao Zhuang, Mahsa Baktashmotlagh, Xi Wang, and Guido Zuccon. 2023. Selecting which Dense Retriever to use for Zero-Shot Search. arXiv:2309.09403 [cs.IR]

[24] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014). https://api.semanticscholar.org/CorpusID:6628106

[25] Cheng Li, Mingyang Zhang, Qiaozhu Mei, Weize Kong, and Michael Bendersky. 2023. Automatic Prompt Rewriting for Personalized Text Generation. *ArXiv* abs/2310.00152 (2023). https://api.semanticscholar.org/CorpusID:263333908

[26] 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. *ArXiv* abs/2308.07968 (2023). https://api.semanticscholar.org/CorpusID:260926523

[27] Pan Li and Alexander Tuzhilin. 2019. Towards Controllable and Personalized Review Generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 3237–3245. https://doi.org/10.18653/v1/D19-1319

[28] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*. Association for Computational Linguistics, Barcelona, Spain, 74–81. https://aclanthology.org/W04-1013

[29] David E. Losada, Javier Parapar, and Alvaro Barreiro. 2018. A rank fusion approach based on score distributions for prioritizing relevance assessments in information retrieval evaluation. *Information Fusion* 39 (2018), 56–71. https://doi.org/10.1016/j.inffus.2017.04.001

[30] Bodhisattwa Prasad Majumder, Shuyang Li, Jianmo Ni, and Julian McAuley. 2019. Generating Personalized Recipes from Historical User Preferences. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 5976–5982. https://doi.org/10.18653/v1/D19-1613

[31] Pierre-Emmanuel Mazaré, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training Millions of Personalized Dialogue Agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 2775–2779. https://doi.org/10.18653/v1/D18-1298

[32] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS '13*. 3111–3119.

[33] Fatemehsadat Mireshghallah, Vaishnavi Shrivastava, Milad Shokouhi, Taylor Berg-Kirkpatrick, Robert Sim, and Dimitrios Dimitriadis. 2022. UserIdentifier: Implicit User Representations for Simple and Effective Personalized Sentiment Analysis. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Seattle, United States, 3449–3456. https://doi.org/10.18653/v1/2022.naacl-main.252

[34] Frederic Morin and Yoshua Bengio. 2005. Hierarchical Probabilistic Neural Network Language Model. In *AISTATS '05*. 246–252.

[35] 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. https://api.semanticscholar.org/CorpusID:265213422

[36] Maxim Naumov et al. 2019. Deep Learning Recommendation Model for Personalization and Recommendation Systems. arXiv:1906.00091 [cs.IR]

[37] Rabia Nuray and Fazli Can. 2006. Automatic ranking of information retrieval systems using data fusion. *Information Processing & Management* 42, 3 (2006),

595–614. https://doi.org/10.1016/j.ipm.2005.03.023
[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. *PyTorch: An Imperative Style, High-Performance Deep Learning Library.* Curran Associates Inc., Red Hook, NY, USA.
[39] Hongjin Qian, Xiaohe Li, Hanxun Zhong, Yu Guo, Yueyuan Ma, Yutao Zhu, Zhanliang Liu, Zhicheng Dou, and Ji-Rong Wen. 2021. Pchatbot: A Large-Scale Dataset for Personalized Chatbot. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 2470–2477. https://doi.org/10.1145/3404835.3463239
[40] Chris Richardson, Yao Zhang, Kellen Gillespie, Sudipta Kar, Arshdeep Singh, Zeynab Raeesy, Omar Zia Khan, and Abhinav Sethy. 2023. Integrating Summarization and Retrieval for Enhanced Personalization via Large Language Models. *ArXiv* abs/2310.20081 (2023). https://api.semanticscholar.org/CorpusID:264805263
[41] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Text Retrieval Conference*. https://api.semanticscholar.org/CorpusID:3946054
[42] H. Roitman, S. Erera, and B. Weiner. 2017. Robust Standard Deviation Estimation for Query Performance Prediction. In *Proceedings of the 2017 International ACM SIGIR Conference on the Theory of Information Retrieval (ICTIR '17)*. 245–248.
[43] Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2023. LaMP: When Large Language Models Meet Personalization. arXiv:2304.11406 [cs.CL]
[44] A. Shtok, O. Kurland, D. Carmel, F. Raiber, and G. Markovits. 2012. Predicting Query Performance by Query-Drift Estimation. *ACM Transactions on Information Systems* 30, 2 (May 2012).
[45] Shayan A. Tabrizi, Azadeh Shakery, Hamed Zamani, and Mohammad Ali Tavallaei. 2018. PERSON: Personalized information retrieval evaluation based on citation networks. *Information Processing & Management* 54, 4 (2018), 630–656. https://doi.org/10.1016/j.ipm.2018.04.004
[46] Sebastian Vincent, Rowanne Sumner, Alice Dowek, Charlotte Blundell, Emily Preston, Chris Bayliss, Chris Oakley, and Carolina Scarton. 2023. Personalised Language Modelling of Screen Characters Using Rich Metadata Annotations. *arXiv preprint arXiv:2303.16618* (2023).
[47] Dingmin Wang, Qiuyuan Huang, Matthew Jackson, and Jianfeng Gao. 2023. Retrieve What You Need: A Mutual Learning Framework for Open-domain Question Answering. (March 2023). https://www.microsoft.com/en-us/research/publication/retrieve-what-you-need-a-mutual-learning-framework-for-open-domain-question-answering/
[48] Ronald J. Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.* 8, 3–4 (may 1992), 229–256. https://doi.org/10.1007/BF00992696
[49] Yuwei Wu, Xuezhe Ma, and Diyi Yang. 2021. Personalized Response Generation via Generative Split Memory Network. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Online, 1956–1970. https://doi.org/10.18653/v1/2021.naacl-main.157
[50] Joern Wuebker, Patrick Simianer, and John DeNero. 2018. Compact Personalized Models for Neural Machine Translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 881–886. https://doi.org/10.18653/v1/D18-1104
[51] Gui-Rong Xue, Jie Han, Yong Yu, and Qiang Yang. 2009. User Language Model for Collaborative Personalized Search. *ACM Trans. Inf. Syst.* 27, 2, Article 11 (mar 2009), 28 pages. https://doi.org/10.1145/1462198.1462203
[52] Sohee Yang and Minjoon Seo. 2020. Is Retriever Merely an Approximator of Reader? arXiv:2010.10999 [cs.CL]
[53] Hamed Zamani and W. Bruce Croft. 2017. Relevance-based Word Embedding. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Shinjuku, Tokyo, Japan) *(SIGIR '17)*. Association for Computing Machinery, New York, NY, USA, 505–514. https://doi.org/10.1145/3077136.3080831
[54] Hamed Zamani, W. Bruce Croft, and J. Shane Culpepper. 2018. Neural Query Performance Prediction using Weak Supervision from Multiple Signals. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 105–114. https://doi.org/10.1145/3209978.3210041
[55] 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* (Madrid, Spain) *(SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 2875–2886. https://doi.org/10.1145/3477495.3531722
[56] Hansi Zeng, Surya Kallumadi, Zaid Alibadi, Rodrigo Nogueira, and Hamed Zamani. 2023. A Personalized Dense Retrieval Framework for Unified Information Access. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Taipei, Taiwan) *(SIGIR '23)*. Association for Computing Machinery, New York, NY, USA, 121–130. https://doi.org/10.1145/3539618.3591626
[57] Kai Zhang, Fubang Zhao, Yangyang Kang, and Xiaozhong Liu. 2023. Memory-Augmented LLM Personalization with Short- and Long-Term Memory Coordination. *ArXiv* abs/2309.11696 (2023). https://api.semanticscholar.org/CorpusID:262083954
[58] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing Dialogue Agents: I have a dog, do you have pets too?. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 2204–2213. https://doi.org/10.18653/v1/P18-1205
[59] Shaoting Zhang, Ming Yang, Timothee Cour, Kai Yu, and Dimitris N. Metaxas. 2015. Query Specific Rank Fusion for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, 4 (2015), 803–815. https://doi.org/10.1109/TPAMI.2014.2346201
[60] Hanxun Zhong, Zhicheng Dou, Yutao Zhu, Hongjin Qian, and Ji-Rong Wen. 2022. Less is More: Learning to Refine Dialogue History for Personalized Dialogue Generation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Seattle, United States, 5808–5820. https://doi.org/10.18653/v1/2022.naacl-main.426
[61] Yun Zhou and W. Bruce Croft. 2007. Query performance prediction in web search environments. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Amsterdam, The Netherlands) *(SIGIR '07)*. Association for Computing Machinery, New York, NY, USA, 543–550. https://doi.org/10.1145/1277741.1277835