Towards On-Device Personalization: Cloud-device Collaborative Data Augmentation for Efficient On-device Language Model

Anonymous ACL submission

Abstract

With the advancement of large language models (LLMs), significant progress has been achieved in various Natural Language Processing (NLP) tasks. However, existing LLMs still face two major challenges that hinder their broader adoption: (1) their responses tend to be generic and lack personalization tailored to individual users, and (2) they rely heavily on cloud infrastructure due to intensive computational requirements, leading to stable network dependency and response delay. Recent research has predominantly focused on either developing cloud-based personalized LLMs or exploring the on-device deployment of general-purpose LLMs. However, few studies have addressed both limitations simultaneously by investigating personalized on-device language models. To bridge this gap, we propose CDCDA-PLM, a framework for deploying personalized on-device language models on user devices with support from a powerful cloud-based LLM. Specifically, CDCDA-PLM leverages the server-side LLM's strong generalization capabilities to augment users' limited personal data, mitigating the issue of data scarcity. Using both real and synthetic data, A personalized on-device language models (LMs) is fine-tuned via parameter-efficient fine-tuning (PEFT) modules and deployed on users' local devices, enabling them to process queries without depending on cloud-based LLMs. This approach eliminates reliance on network stability and ensures high response speeds. Experimental results across six tasks in a widely used personalization benchmark demonstrate the effectiveness of CDCDA-PLM.

1 Introduction

Recently, Large Language Models (LLMs) have become a cornerstone of contemporary Natural Language Processing (NLP) research and industry applications due to their exceptional abilities in text understanding and generation (Radford and Narasimhan, 2018; Ray, 2023; Naveed et al., 2024). These models have achieved remarkable success and transformed numerous areas of NLP, such as translation, summarization, and conversational AI (Thirunavukarasu et al., 2023; Hu et al., 2024; Wang et al., 2024).

Despite their advancements, existing LLMs face two significant limitations that hinder their broader adoption: (1) **Lack of Personalization**. LLMs are designed as universal models, which limits their ability to generate responses tailored to users' personalized preferences and interests; (2) **Dependence on Cloud Infrastructure**. The powerful LLMs are typically trained and deployed on cloud servers due to their high computational demands. This architecture not only relies on stable and high-speed network connections to transmit user queries and deliver responses, but also requires a long time for LLM inference. However, these conditions are often unmet in real-world scenarios, particularly on mobile platforms such as smartphones and smart vehicles. For instance, LLM-based assistants are increasingly integrated into smart vehicle systems, but these vehicles frequently experience inconsistent network connectivity due to their mobility. In remote regions with weak or no network coverage, such cloud-based LLM services could become entirely inaccessible as the system goes offline. In addition, considering the time-sensitive applications, an LLM-powered service will lead to a significant inference latency overhead in reality, limiting the feasibility of the service. Consequently, there is a growing need for personalized LLMs that can run directly on user devices. Such models must address user-specific needs while operating efficiently on edge devices, free from the constraints of cloud connectivity.

Some recent efforts have explored techniques for enabling personalization in LLMs, which can be generally categorized into prompt-based methods and fine-tuning-based methods. The prompt-based

approaches format personalized prompts to leverage the in-context learning capabilities of LLMs. That is to say, all users share the same model, but personalized prompts are used to guide the generation process. For example, Christakopoulou et al. (2023) incorporates users' historical data into prompts to enhance generation performance. And to conquer the input length limitation when users' historical data are too long, some research employs retrieval-augmentation generation (RAG) to augment user's query by adding the most relevant history information into prompt (Richardson et al., 2023; Salemi et al., 2023; Li et al., 2024a,b). On the other hand, fine-tuning methods directly optimize the parameters of LLMs to adapt to users' personalized data distributions (Tan et al., 2024b,a; Park et al., 2024; Li et al., 2024b; Zhuang et al., 2024). However, these approaches face significant scalability issues, as the cloud server must fine-tune a separate model for each user. As the number of users grows, centralized computation becomes a major bottleneck, making the fine-tuning-based personalization method impractical.

While these methods show promise for personalization, they are primarily designed for cloud-based LLMs and face significant challenges in on-device settings. On-device language models (LMs) are constrained by the computational and storage limitations of edge devices, resulting in small model sizes. As demonstrated in many previous works (Richardson et al., 2023; Salemi et al., 2024), prompt-based personalization methods, including RAGs, cannot achieve satisfactory performance with these small-sized on-device LMs since these models have limited generalization and contextual understanding ability. Similarly, fine-tuning on-device LMs to adapt to users' local data distributions presents additional difficulties. For example, individual users typically possess limited data, which is insufficient for effective model fine-tuning.

In this paper, we take a first step toward developing a learning framework for personalized on-device language models (LMs). Our approach leverages cloud-based large language models (LLMs) to address the challenge of personal data scarcity and introduces a cloud-device collaborative framework to ensure scalability. Specifically, the cloud-based LLM generates synthetic data tailored to each user's limited local data, thereby augmenting the user's dataset and transferring relevant knowledge from the cloud LLM to the on-device LM. Once users receive the synthetic data, we apply parameter-efficient fine-tuning (PEFT) techniques to optimize their on-device LMs using both the synthetic and local personal data entirely on the user's device. This decentralized training strategy avoids the scalability bottleneck of requiring the cloud server to fine-tune models for all users. After training, the personalized on-device LM is deployed locally, allowing inference to be performed without network connectivity, which reduces both network dependency and inference latency. To evaluate the framework's effectiveness, we conduct extensive experiments on public datasets, and experimental results demonstrate that the proposed method can achieve promising performance in personalized classification and generation tasks.

Overall, the main contributions of this paper are summarized as follows:

- We take the first step in exploring the problem of LLM personalization in the context of small on-device LM deployment, where storage size and computational resources are constrained.
- We propose a personalized on-device LM framework, CDCDA-PLM. In this framework, we design a novel cloud-device collaboration mechanism in which the server model leverages data augmentation to transfer knowledge to the small on-device LM. Additionally, we develop a dedicated filtering method to enhance the robustness of the knowledge transfer process.
- We conduct extensive experiments across multiple tasks to demonstrate the effectiveness of CDCDA-PLM. Furthermore, we perform detailed ablation studies and hyperparameter analyses, followed by a case study, to further highlight the superiority of our proposed method.

2 Related Work

In this section, we review the literatures on LLM personalization and on-device deployment of LLMs.

2.1 Personalization of LLMs

Personalized LLM aims to better understand and generate text specific to match the user's interests and preferences. The existing research on LLM personalization could generally be divided into two categories: prompt design based personalization

and parameter-efficient fine-tuning (PEFT) based personalization (Salemi and Zamani, 2024).

Prompt-based Personalization. In the early development of personalized prompts, query prompts were formatted with user history as context to leverage the in-context and few-shot learning capabilities of large language models (LLMs). For instance, Christakopoulou et al. (2023) and Zhiyuli et al. (2023) demonstrate that incorporating long user history in prompts can enhance LLM generation performance. However, incorporating user history in prompts will increase the inference computational cost due to the lengthy input. To mitigate this issue, Salemi et al. (2023) proposed a strategy to shorten the user history length by using retrieval model to select relevant documents from user history based on the user query. Moreover, Salemi et al. (2024) optimizes and selects retrieval models based on LLM feedback from personalized tasks.

Fine-tuning based Personalization. Parameter-efficient fine-tuning (PEFT) offers an effective way to optimize LLMs for users' personal distributions by modifying only a small subset of parameters (Hu et al., 2021; Dettmers et al., 2023). For example, OPPU proposed a PEFT-based personalized LLM, which fine-tunes the LoRA adapter on user profiles for each user, to store user knowledge on PEFT parameters (Tan et al., 2024b). Building on this work, PER-PCS aggregates fine-tuned LoRA adapters from multiple users into a shared adapter pool, which can be leveraged to generate a personalized LLM for a target user by merging multiple LoRA adapters (Tan et al., 2024a). Reinforcement learning is also applied with PEFT to achieve better performance(Cheng et al., 2023; Li et al., 2024b; Park et al., 2024).

However, all the aforementioned personalization approaches have been developed for cloud-based LLMs, which possess formidable generalization and language understanding capabilities, lacking the exploration of weak on-device models.

2.2 On-device Deployment of LLMs

Due to their large size, deploying LLMs on edge devices presents critical challenges, including high computational overhead and significant memory demands. Current deployment methods can generally be categorized into two strategies.

The first strategy involves directly compressing the original large-scale model into a smaller one through quantization (Liu et al., 2023; Lin et al., 2025) and pruning (Ma et al., 2023; Frantar and Alistarh, 2023). Quantization maps high-precision values to lower precision, while pruning removes certain unimportant neurons. However, since the compressed model remains architecturally coupled with the original model, aggressive compression may lead to significant performance degradation.

The second strategy focuses on transferring knowledge from a large cloud-based model to a smaller on-device model. A widely used approach within this strategy is knowledge distillation (KD) (Hinton et al., 2015; Gou et al., 2021). Based on the accessibility of the teacher model, the KD process can be classified into white-box KD and black-box KD. In white-box KD, the student model learns from the teacher model's activations, hidden features, and output distribution (Xu et al., 2024; Ko et al., 2024; Wu et al., 2024; Agarwal et al., 2024; Gu et al., 2024). However, this approach requires the student model to share certain architectural similarities with the teacher model. In contrast, black-box KD allows the student model to access only the teacher model's responses to enhance training data (Dai et al., 2023; Ho et al., 2023; Tian et al., 2024; Jung et al., 2024). For instance, Qin et al. (2024) introduces an on-device LLM training framework by selecting the most representative user data to mitigate the data storage demands in the device. However, in their method, the on-device model is as large as the cloud-based model which is impractical. Our proposed method aligns closely with black-box KD, leveraging a cloud-based model to generate a synthetic dataset that transfers knowledge to the smaller on-device LM model.

3 Research Problem Formulation

This paper explores to fine-tune personalized on-device language models (LMs), incorporating two key concepts: personalized LMs and on-device LMs. Unlike general LMs, which produce output sequences solely based on the input sequence, a personalized LM generates responses by considering both the user's query x and their profile D_u. We define the user profile as a collection of the user's historical input-output pairs: i.e., $D_u = \{(x_{u1}, y_{u1}), (x_{u2}, y_{u2}), \ldots, (x_{ut_u}, y_{ut_u})\}$, where t_u indicates the history before query time t.

Compared to a cloud-based LLM M_{cloud}, an on-device LM M_{device} has a significantly smaller model size, as it must be deployed on a user's local device where computational resources are limited. Additionally, unlike server-based LLMs, which are

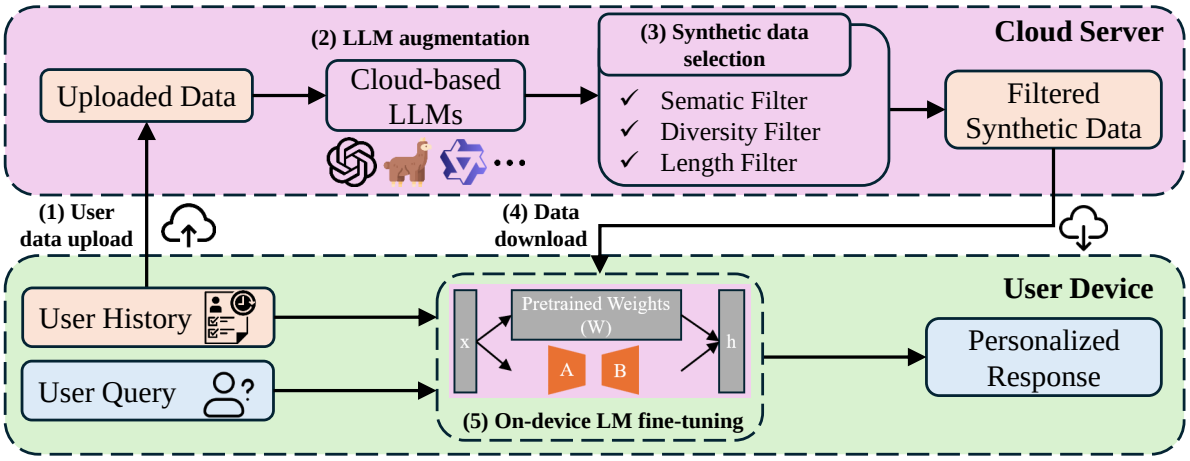

Figure 1: Overview of the proposed method.

trained on extensive datasets collected from various sources, on-device LMs are only trained on a single user's data, which is often insufficient. As a result, on-device LM has fast response ability, however, its does not have powerful language understanding ability compared to server-based LLMs. In this paper, we aim to exploit both the on-device LM's fast response merits and the strong inference ability of the server LLMs to build a high-performance on-device personalized LM.

4 Proposed Method

Different from previous work (Salemi et al., 2023), which implements personalization in a cloud server setting, this paper proposes a cloud-device collaborative data augmentation for on-device personalized LM deployment framework that enhances inference efficiency without relying on the server LLM. The basic idea of CDCDA-PLM is to use the powerful server LLM model to assist the on-device personalized model's fine-tuning. As shown in Figure 1, the proposed framework consists of the following five steps: (1) user data uploading, (2) data augmentation with server LLM, (3) synthetic data selection, (4) synthetic data downloading, and (5) on-device LM personalization fine-tuning. In the following parts, we provide a detailed description of each step.

User data uploading. A user's historical profile provides a unique data distribution. However, due to the limited data size, directly fine-tuning the on-device model on this local data cannot achieve satisfactory performance. Therefore, in CDCDA-PLM, user upload their personal data to a central server, where a powerful cloud-based LLM performs data augmentation.

Data augmentation with server LLM. On the server side, we use the following prompt to augment the uploaded data: *"Generate an Input and Response pairs semantically similar to the following example, no need to explain. Input: [], Response: []."* Then, for each pair of data in the uploaded user dataset D_u, the server LLM M_{cloud} generates k augmented samples:

$$D_u^{syn} = \{D_{ui} = \{(x_{ui}^j, y_{ui}^j)\}_{j=1}^k\}_{i \in I_u} \quad (1)$$

Synthetic data selection. Although M_{cloud} generates a large amount of data for the target user, the generated data can be noisy, and not all samples contribute useful information for personalization fine-tuning. Intuitively, high-quality augmented data should be similar to the original samples while still providing some diversity. Therefore, we apply three carefully designed filters to select useful data for personalized LM.

Filter 1: Semantic consistency filter. Reliable synthetic data should preserve the semantics of the original statement without introducing hallucinated content. Natural Language Inference (NLI) models are trained to determine whether a given "hypothesis" and "premise" entail, contradict, or are neutral to each other. Therefore, we employ a small NLI model M_{NLI} (Liu et al., 2022) as the semantic evaluator, which provides a semantic consistent score between the synthetic and original samples:

$$SCF = (M_{NLI}(x \Rightarrow x_{syn}) \geq \epsilon_{scf}) \wedge (M_{NLI}(x_{syn} \Rightarrow x) \geq \epsilon_{scf}) \quad (2)$$

where ϵ_{scf} is the threshold to filter out dissimilar

synthetic pairs, and $M_{NLI}(a \Rightarrow b)$ indicates the possibility of inferring b given a.

Filter 2: Token diversity filter. While the SCF filter ensures the consistency of semantics for synthetic data, it is also important to maintain diversity in the augmented data. Ideally, synthetic samples should convey the original meaning but with different wording. To measure this, we apply the ROUGE-L (Lin, 2004) metric to assess token overlap between original and generated sequences:

$$TDF = \text{ROUGE-L}(x, x_{syn}) \leq \epsilon_{tdf} \quad (3)$$

where ϵ_{tdf} is the threshold for the ROUGE-L score.

Filter 3: Length size filter. Finally, we ensure that synthetic samples have a reasonable length to avoid abnormal or redundant data. We discard data that are either too short or too long, using predefined minimum and maximum length thresholds ϵ_{min_len} and ϵ_{max_len}:

$$LSF = (len(x_{syn}) \geq \epsilon_{min\_len} \cdot len(x) \wedge (len(x_{syn}) \leq \epsilon_{max\_len} \cdot len(x)) \quad (4)$$

Specifically, we filter all generated samples whose length ratio (i.e., the length ratio of x_{syn} to x) is out of the pre-defined range $[\epsilon_{min\_len}, \epsilon_{max\_len}]$ to ensure the generated sample has a length similar to the input. By applying these three filters, we obtain a high-quality dataset $D_{filtered}$ from the synthetic data pool D_{syn}, which is then used for LM fine-tuning.

Synthetic data downloading. After selecting the high-quality augmented data $D_{filtered}$, the server sends these data back to the corresponding users. Users then download the data and combine it with their local datasets for on-device fine-tuning.

On-device LM personalization fine-tuning. We employ a pretraining and efficient fine-tuning approach for on-device personalization. Specifically, for a target task, we fine-tune a general LM on a public, standard dataset to enhance its general task understanding. Since this step does not involve personal data, it is executed on the cloud server to avoid using the constrained on-device resources. After optimization, we obtain a task-specific pretrained model M_{base} as the initialization point for personalized LMs fine-tuning.

On the device, we fine-tune an on-device LM M_{base} on the combined synthetic and user local datasets to learn both personalized information from the user and the insightful knowledge from the server LLM. The on-device LM is much smaller than the server LLM, ensuring low inference latency.

To reduce training costs, we implement parameter-efficient fine-tuning using LoRA (Dettmers et al., 2023). LoRA introduces trainable adapters ΔW_u into the original weights of M_{base}, forming the personalized LM M_{device}:

$$M_{device} = M_{base} + \Delta W_u \quad (5)$$

We then only optimize ΔW_u using the user's historical data D_u and the filtered LLM-generated data $D_u^{filtered}$.

$$\Delta W_u = argmin\, CE(M_{device} | D_u \cup D_u^{filtered}) \quad (6)$$

where $CE(\cdot)$ represents the cross-entropy loss function.

After optimizing the personalized LM M_{device}, users can then process queries locally without relying on the cloud server, benefiting from lower latency and without relying on network connection.

5 Experiments

5.1 Experimental Settings

Datasets. To validate the effectiveness of the proposed method, we conduct extensive experiments on six personalization tasks in Large Language Model Personalization (LaMP) benchmark (Salemi et al., 2023).[1] In this study, we use the time-based separation data in LaMP benchmark. Note that our data usage slightly differs from the original LaMP benchmark setting. Specifically, they address multi-user personalization by training models on the combined histories of multiple users, which results in coarse-grained personalization. In contrast, our work focuses on fine-grained personalization by training a separate model for each individual user using only their own historical data. The core statistics of pre-processed datasets for each task are presented in Appendix A. To promote the personalization phenomenon, following (Tan et al., 2024b), we select the 100 most active users with the longest history logs as target users while using all remaining users for base model training. Our objective is to obtain on-device personalized LM for each user among these 100 users. In addition, to further investigate the effectiveness of our method,

[1]We exclude the LaMP-6: Email Subject Generation task as it relies on private data that we cannot access.

Table 1: The performance of CDCDA-PLM and baselines on LaMP benchmark. The best performance of personalized on-device model M_{device} is highlighted in **bold** and the second best is underlined.

Tasks	Metric	Non-Personalized		RAG				Speculative Decoding	Fine-tuning based M_{device}			
		M_{cloud}	M_{device}	M_{cloud} +BM25	M_{device} +BM25	M_{cloud} +Contriever	M_{device} +Contriever	M_{device}	Direct -FT	EDA -FT	RKD -FT	CDCDA -PLM
LaMP-1	Accuracy ↑	0.520	0.390	0.560	0.310	0.630	0.230	0.500	0.420	0.410	0.460	**0.530**
	F1 ↑	0.515	0.356	0.528	0.381	0.605	0.245	0.469	0.390	0.382	0.389	**0.483**
LaMP-2	Accuracy ↑	0.248	0.017	0.319	0.009	0.292	0.050	0.353	0.243	0.296	0.283	**0.391**
	F1 ↑	0.129	0.017	0.225	0.019	0.234	0.066	0.201	0.099	0.156	0.125	**0.224**
LaMP-3	MAE ↓	1.120	0.640	1.970	1.580	1.630	1.465	0.550	0.474	0.450	0.474	**0.400**
	RMSE ↓	1.371	1.131	2.508	2.191	2.252	2.117	1.034	0.946	**0.831**	0.912	0.834
LaMP-4	ROUGE-1 ↑	0.107	0.102	0.122	0.092	0.121	0.098	0.110	0.106	0.117	0.116	**0.120**
	ROUGE-L ↑	0.096	0.090	0.110	0.083	0.109	0.088	0.098	0.094	0.104	0.103	**0.107**
	BERT-F1 ↑	0.847	0.838	0.849	0.837	0.850	0.839	0.847	0.845	0.847	0.847	**0.849**
LaMP-5	ROUGE-1 ↑	0.427	0.360	0.457	0.328	0.453	0.346	0.341	0.375	0.370	0.375	**0.382**
	ROUGE-L ↑	0.362	0.309	0.379	0.292	0.387	0.292	0.279	0.314	0.316	0.307	**0.317**
	BERT-F1 ↑	0.894	0.885	0.896	0.882	0.896	0.878	0.879	**0.886**	0.885	0.884	**0.886**
LaMP-7	ROUGE-1 ↑	0.365	0.337	0.355	0.296	0.338	0.230	0.354	0.337	0.373	0.374	**0.383**
	ROUGE-L ↑	0.310	0.297	0.315	0.262	0.296	0.201	0.317	0.302	0.327	0.328	**0.336**
	BERT-F1 ↑	0.881	0.877	0.881	0.869	0.879	0.854	0.878	0.875	0.880	**0.882**	0.881

we also implement experiments for additional users from LaMP. The further experiment details are provided in Appendix D.

Evaluation Metrics. Following LaMP (Salemi et al., 2023), we use accuracy and F1-score for the LaMP-1 and LaMP-2, MAE and RMSE for the LaMP-3, and ROUGE-1, ROUGE-L (Lin, 2004) and BERTScore-F1 (BERT-F1) (Zhang et al., 2020) for LaMP-4, LaMP-5 and LaMP-7. Except for MAE and RMSE, where lower values are better, all other metrics with higher values indicate better performance.

Baselines. We compare CDCDA-PLM with the non-personalized models and other personalized baselines. In the non-personalized baselines, we select the cloud-based LLM (M_{cloud}) and on-device LM (M_{device}), which is fine-tuned only on the remaining users without the 100 target users.

The personalized baselines include RAG-based methods, fine-tuning based methods, and speculative decoding method, for fair comparison, these personalized methods are all implemented on on-device models: (1) Retrieval-Augmented Personalization (RAG): RAG incorporates relevant items from target user history to the prompt (Salemi et al., 2023) to achieve a personalized response. To showcase the deteriorated performance of RAG in on-device LM, we also present the performance of RAG in the cloud counterpart. Follow by (Salemi et al., 2023), in experiment, we implement a sparse retriever: BM25 (Trotman et al., 2014) and a dense retriever: Contriever (Izacard et al., 2021). (2) Speculative Decoding: An optimization technique accelerates models' inference by using a non-personalized LLM on cloud to assist a personalized LM on device. We evaluate personalized tasks on Google's speculative decoding (Leviathan et al., 2023) implemented by HuggingFace (Joao Gante, 2023). (3) Direct-FT (Tan et al., 2024b): Directly LoRA fine-tuning M_{device} uses the target user's local historical data. This method cannot be satisfied due to limited local data size. (4) EDA-FT (Wei and Zou, 2019): EDA (Easy Data Augmentation) is a traditional text data augmentation method including synonym replacement, random insertion, random swap, and random deletion. (5) RKD-FT: An LLM knowledge distillation method uses reverse KL divergence (Gu et al., 2024). For EDA-FT, RKD-FT, and CDCDA-PLM, they augment local knowledge based on users' data.

Table 2: Ablation studies results with respect to server LLM data augmentation (LDA) and data selection (DS) components. The best are highlighted in **bold**.

Methods	LaMP-2		LaMP-7	
	Acc	F1	R-1	R-L
Full model (DS)	**0.391**	**0.224**	**0.383**	**0.336**
Full model (RS)	0.324	0.157	0.344	0.302
-DS	0.360	0.187	0.354	0.314
-DS -LDA	0.243	0.099	0.337	0.302
-DS -LDA -FT	0.017	0.017	0.337	0.297

Implementation. For all baselines in our study, we choose models from one of the most widely adopted open-source LLM series *Qwen2.5* [2] (Yang et al., 2024). Specifically, we use *Qwen2.5-3B-Instruct* as the cloud-based model and *Qwen2.5-0.5B-Instruct* as the on-device model for each user. To ensure efficiency, we choose one retriever item

[2] https://github.com/QwenLM/Qwen2.5

for all retrieval-based methods.

By default, we set the LLM generation samples k to 5 in all experiments. We apply the LoRA adapter on all linear layers of the on-device model, and set the LoRA rank r to 16 and scaling factor α to 8. We quantize the on-device model weight in NF4 data type and use bfloat 16 for computation. Followed by the Qwen2.5 technique report (Yang et al., 2024), we used the multinomial sampling decoding with temperature $\tau_{temp} = 0.7$. We implement all the experiments using Pytorch (Paszke et al., 2017) and HuggingFace library (Wolf et al., 2020) on an NVIDIA RTX A5000 GPU.

5.2 Overall Results

To validate our proposed method's effectiveness, we compare it with several baselines and show the results in Table 1. From the results, we have some interesting observations as follows.

First, by comparing the cloud model M_{cloud} with the device model M_{device}, we observe that the cloud model performs significantly better than the corresponding device model in both personalized and non-personalized settings. This is because cloud-based models have a much larger number of parameters, approximately six times more in our experiments, making them unsuitable for deployment on edge devices. This finding highlights the necessity of transferring knowledge from the cloud-based LLM to support the weaker on-device LM.

Furthermore, when comparing RAG-based personalization methods, we find that the performance of the small on-device model actually declines after incorporating RAG. This aligns with our argument that on-device models are too small to effectively support prompt-based personalization.

By comparing speculative decoding with other baselines, we observe that it achieves relatively strong performance. However, as discussed in the baseline section, speculative decoding relies on frequent interaction with a cloud-based LLM, which introduces additional latency and requires a stable network connection. In addition, our proposed CDCDA-PLM outperforms speculative decoding by a clear margin, demonstrating its superiority in terms of performance. Moreover, CDCDA-PLM operates entirely on-device without the need for network connectivity and enables fast decoding, offering further advantages over the speculative decoding approach.

Among fine-tuning-based personalization approaches, Direct-FT yields the worst performance due to the limited availability of local user data, which is typically insufficient for effective personalized fine-tuning. The baseline methods, EDA-FT and RKD-FT, improve upon direct fine-tuning in some tasks, but their enhancements are limited. In some cases, their performance even deteriorates, likely due to the simplistic knowledge augmentation techniques they employ.

Our proposed CDCDA-PLM consistently outperforms all on-device baselines across all tasks. Additionally, CDCDA-PLM achieves performance comparable to cloud models, demonstrating its effectiveness and strong generalization ability.

5.3 Ablation Study

In this part, on LaMP-2 and LaMP-7, we demonstrate the effectiveness of our delicately designed modules in CDCDA-PLM, including LLM data augmentation (LDA) and data selection (DS) components. As shown in Table 2, when we replace our carefully designed filters in DS with random selection (RS), the accuracy of the full model with DS drops from 0.391 to 0.324 on LaMP-2. When we remove the DS (i.e., -DS), i.e., the on-device model is directly trained on all augmented data, the ROUGE-1 score also decreases from 0.391 to 0.360 on LaMP-2. Furthermore, when we fine-tune on-device models without LLM data augmentation (i.e., -LDA), the model performance further drops to 0.243 accuracy and 0.337 ROUGE-1 score on LaMP-2 and LaMP-7. Overall, the results support the effectiveness of all the proposed components.

5.4 Hyper-parameter Analysis

In this part, we investigate the impact of the hyper-parameters, synthetic data augmentation size k, associated with our proposed method. To better understand the impact of cloud-based LLM augmentation, we vary the number of LLM-generated samples k for both the classification (LaMP-2) and generation (LaMP-7) tasks, as shown in Figure 2. Overall, increasing k leads to improvements in both tasks LaMP-2 and LaMP-7. Specifically, in both tasks, performance stabilizes on generating 1 and 3 samples and achieves greater improvement when increasing generated samples to 5.

5.5 Efficiency Analysis

To further evaluate the CDCDA-PLM's efficiency on real-world edge devices, we deploy personalized on-device LM on two devices: (1) a work-

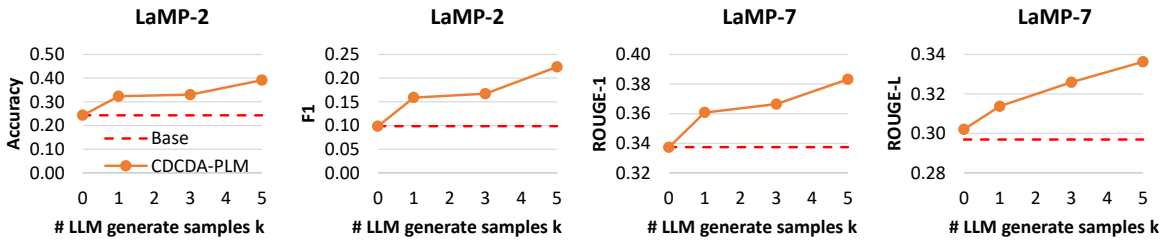

Figure 2: The impact of hyperparameter in LLM data augmentation. k controls the number of samples generated by server-sided LLM.

Table 3: Inference efficiency results. Storage size is the required memory for deploying models. TTFT represents the time to generate the first token (sec), and Decode represents the average number of output tokens generated per second (tokens/s).

Models	Storage Size (GB)	Workstation		Android	
		TTFT	Decode	TTFT	Decode
Cloud-based LLM	2.36	4.8	234	31.7	2.3
On-device LM	0.42	0.9	688	4.9	8.6
	(-82%)	(-81%)	(2.9x)	(-85%)	(3.8x)

Table 4: Comparative results with FlanT5-base model on 6 tasks. In each task, the best result is marked in **bold**.

Tasks	FlanT5-base		Qwen2.5-0.5B
	+BM25	CDCDA-PLM	CDCDA-PLM
LaMP-1 (Acc)	**0.6**	0.570	0.530
LaMP-2 (Acc)	0.3499	0.283	**0.391**
LaMP-3 (MAE)	0.4632	0.484	**0.400**
LaMP-4 (R-1)	0.0834	0.115	**0.120**
LaMP-5 (R-1)	0.2867	**0.406**	0.382
LaMP-7 (R-1)	0.2933	0.362	**0.383**

station with GPU NVIDIA RTX A5000 and (2) a Samsung Galaxy Tab A8. We first train all the on-device personalized LMs on a workstation with a GPU, and then they are encapsulated by MLC-LLM (MLC team, 2025), which is a compiler and high-performance deployment engine for LLMs. To compare the efficiency in on-device deployment, We evaluate on-device model deployment's efficiency from two perspectives: model storage memory, and inference efficiency (TTFT and decoding speed). We record the inference efficiency for 5 runs of each model on the randomly selected queries from 6 tasks.

The results are reported in Table 3. Our on-device models are much smaller and faster than the cloud-based models on two devices. Specifically, the memory footprint of our on-device model is more than five times smaller than that of the server-based model. The time-to-first-token (TTFT) for our on-device model is just 0.9 seconds on a workstation and 4.9 seconds on an Android phone—both lower than the TTFT of the server model. In terms of throughput, our on-device model generates output tokens at a rate three times faster than the original server model. Notably, according to Rayner et al. (2016), the average human reading speed is approximately 4 to 6 words per second. Our on-device model, even on a low-end Android device, achieves a decoding speed of 8.6 tokens per second, which is sufficient for real-world applications. Overall, the efficiency analysis proves that our developed personalized LM is able to achieve more practical memory and response latency for on-device services.

5.6 Generalization of our framework

In this part, we investigate the effectiveness of our framework on another language model, FlanT5-base (Chung et al., 2022). As shown in Table 4, FlanT5-base with CDCDA-PLM achieves better performance than FlanT5-base with BM25 RAG on 4 tasks due to limited historical data on a single user, indicating that our framework is able to be implemented on different LM architectures to improve the on-device personalized performance.

6 Conclusion

This paper introduces CDCDA-PLM, a personalized on-device LM deployment framework designed to close the performance and efficiency gap between cloud-based LLM and on-device LM by augmenting user historical data. Specifically, CDCDA-PLM first uses a server LLM to construct a synthetic dataset containing similar samples as user data to assist the on-device personalized model's fine-tuning. The experimental results demonstrate that CDCDA-PLM achieves better performance on personalized content generation.

7 Limitations

Several limitations are concerned with our work. Firstly, due to dataset constraints, our study aims to deploy a personalized model to generate responses on a specific task for each user, ignoring the user behaviors from other tasks and domains. For example, for the user who engages in news headline generation and scholarly title generation tasks, both tasks could provide the user's stylistic pattern preference. Nevertheless, in the future, we believe CDCDA-PLM can be applied to any NLP task across different domains. Secondly, the data quality of LLM augmentation can be affected by the cloud-based LLM. Exploring a larger LLM or multiple LLMs to augment user data remains an area for future investigation.

8 Ethical Considerations

Training a personalized model heavily relies on personal data, which may leak sensitive or private information of users. Sharing user data with server LLM for user personal data augmentation also leads to privacy concerns. Therefore, it is important to investigate further robust methods for privacy protection in cloud-server LLM data augmentation. In addition, a personalized model aims to generate content aligning with user preferences and interests shown in user data. However, personalization models may be trained with user data consisting of biased and unfair information, leading to harmful responses. Within CDCDA-PLM, the biased data is uploaded to server LLM for augmentation, which further negatively affects the on-device model. Future works may explore strategies to avoid sharing or augmenting harmful data on the server LLM.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. *Preprint*, arXiv:2306.13649.

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

Konstantina Christakopoulou, Alberto Lalama, Cj Adams, Iris Qu, Yifat Amir, Samer Chucri, Pierce Vollucci, Fabio Soldo, Dina Bseiso, Sarah Scodel, Lucas Dixon, Ed H. Chi, and Minmin Chen. 2023. Large language models for user interest journeys. *Preprint*, arXiv:2305.15498.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, 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. 2022. Scaling instruction-finetuned language models. *arXiv preprint*.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, Hongmin Cai, Lichao Sun, Quanzheng Li, Dinggang Shen, Tianming Liu, and Xiang Li. 2023. Auggpt: Leveraging chatgpt for text data augmentation. *Preprint*, arXiv:2302.13007.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Preprint*, arXiv:2305.14314.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: massive language models can be accurately pruned in one-shot. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. Minillm: Knowledge distillation of large language models. *Preprint*, arXiv:2306.08543.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *Preprint*, arXiv:1503.02531.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers. *Preprint*, arXiv:2212.10071.

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

Yuchen Hu, Chen Chen, Chao-Han Yang, Ruizhe Li, Dong Zhang, Zhehuai Chen, and EngSiong Chng. 2024. GenTranslate: Large language models are generative multilingual speech and machine translators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 74–90, Bangkok, Thailand. Association for Computational Linguistics.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning.

Joao Gante. 2023. Assisted generation: a new direction toward low-latency text generation.

Jaehun Jung, Peter West, Liwei Jiang, Faeze Brahman, Ximing Lu, Jillian Fisher, Taylor Sorensen, and Yejin Choi. 2024. Impossible distillation for paraphrasing and summarization: How to make high-quality lemonade out of small, low-quality model. 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 4439–4454, Mexico City, Mexico. Association for Computational Linguistics.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. 2024. Distillm: Towards streamlined distillation for large language models. *Preprint*, arXiv:2402.03898.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19274–19286. PMLR.

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 Web Conference 2024*, WWW '24, page 3367–3378. ACM.

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

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

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Guangxuan Xiao, and Song Han. 2025. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *GetMobile: Mobile Comp. and Comm.*, 28(4):12–17.

Alisa Liu, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. 2022. WANLI: Worker and AI collaboration for natural language inference dataset creation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6826–6847, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2023. Llm-qat: Data-free quantization aware training for large language models. *Preprint*, arXiv:2305.17888.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. In *Advances in Neural Information Processing Systems*, volume 36, pages 21702–21720. Curran Associates, Inc.

MLC team. 2025. MLC-LLM.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2024. A comprehensive overview of large language models. *Preprint*, arXiv:2307.06435.

Chanwoo Park, Mingyang Liu, Dingwen Kong, Kaiqing Zhang, and Asuman Ozdaglar. 2024. Rlhf from heterogeneous feedback via personalization and preference aggregation. *Preprint*, arXiv:2405.00254.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Ruiyang Qin, Jun Xia, Zhenge Jia, Meng Jiang, Ahmed Abbasi, Peipei Zhou, Jingtong Hu, and Yiyu Shi. 2024. Enabling on-device large language model personalization with self-supervised data selection and synthesis. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*, DAC '24, New York, NY, USA. Association for Computing Machinery.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Partha Pratim Ray. 2023. Chatgpt: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems*, 3:121–154.

Keith Rayner, Elizabeth R. Schotter, Michael E. J. Masson, Mary C. Potter, and Rebecca Treiman. 2016. So much to read, so little time: How do we read, and can speed reading help? *Psychological Science in the Public Interest*, 17(1):4–34. PMID: 26769745.

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.

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

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

Alireza Salemi and Hamed Zamani. 2024. Comparing retrieval-augmentation and parameter-efficient fine-tuning for privacy-preserving personalization of large language models. *Preprint*, arXiv:2409.09510.

Zhaoxuan Tan, Zheyuan Liu, and Meng Jiang. 2024a. Personalized pieces: Efficient personalized large language models through collaborative efforts. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6459–6475, Miami, Florida, USA. Association for Computational Linguistics.

Zhaoxuan Tan, Qingkai Zeng, Yijun Tian, Zheyuan Liu, Bing Yin, and Meng Jiang. 2024b. Democratizing large language models via personalized parameter-efficient fine-tuning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6476–6491, Miami, Florida, USA. Association for Computational Linguistics.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. 2023. Large language models in medicine. *Nature medicine*, 29(8):1930–1940.

Yijun Tian, Yikun Han, Xiusi Chen, Wei Wang, and Nitesh V. Chawla. 2024. Beyond answers: Transferring reasoning capabilities to smaller llms using multi-teacher knowledge distillation. *Preprint*, arXiv:2402.04616.

Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to bm25 and language models examined. In *Proceedings of the 19th Australasian Document Computing Symposium*, ADCS '14, page 58–65, New York, NY, USA. Association for Computing Machinery.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. 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)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Taiqiang Wu, Chaofan Tao, Jiahao Wang, Runming Yang, Zhe Zhao, and Ngai Wong. 2024. Rethinking kullback-leibler divergence in knowledge distillation for large language models. *Preprint*, arXiv:2404.02657.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. 2024. A survey on knowledge distillation of large language models. *Preprint*, arXiv:2402.13116.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. *Preprint*, arXiv:1904.09675.

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

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

Table 5: Statistics of the preprocessed datasets..

Task	Target Users				Filtered Synthetic Dataset		
	# Q	# History	L_{in}	L_{out}	# Q	L_{in}	L_{out}
LaMP-1	100	317.57	78.43	3.0	15928	161.76	19.15
LaMP-2	2752	54.58	129.55	2.24	8962	121.21	2.20
LaMP-3	100	959.02	244.79	1.00	15721	193.19	1.00
LaMP-4	955	269.08	31.49	15.60	12736	27.40	14.27
LaMP-5	100	443.03	222.60	15.52	23473	156.92	18.63
LaMP-7	100	120.15	40.90	27.66	16490	39.56	0.00

A Datasets

In Table 5, #Q and #History represent the total number of user queries and history, respectively, in the target users test dataset and synthetic selected training dataset. L_{in} and L_{out} are the average tokens of inputs and outputs.

B Prompt Details

In this part, we show the prompt used in our experiment.

B.1 LaMP-1: Personalized Citation Identification

<|im_start|>system
With the given examples, which reference is related?<|im_end|>
<|im_start|>user
{RETRIEVED USER HISTORY}
For an author who has written the paper with the title {PAPER TITLE}, which reference is related? Just answer with [1] or [2] without explanation. [1]: {OPTION_1} [2]: {OPTION_2}<|im_end|>

B.2 LaMP-2: Personalized Movie Tagging

<|im_start|>system
With the given examples, generate a tag for the given movie. <|im_end|>
<|im_start|>user
{RETRIEVED USER HISTORY}
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: {MOVIE DESCRIPTION} <|im_end|>

B.3 LaMP-3: Personalized Product Rating

<|im_start|>system
With the given examples, generate a score for the given review. <|im_end|>
<|im_start|>user
{RETRIEVED USER HISTORY}
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} <|im_end|>

B.4 LaMP-4: Personalized News Headline Generation

<|im_start|>system
With the given examples, generate a title for the given article. Only output the title and nothing else. <|im_end|>
<|im_start|>user
{RETRIEVED USER HISTORY}
Generate a headline for the following article: {ARTICLE} <|im_end|>

B.5 LaMP-5: Personalized Scholarly Title Generation

<|im_start|>system
With the given examples, generate a title for the given article. Only output the title and nothing else. <|im_end|>
<|im_start|>user
{RETRIEVED USER HISTORY}
Generate a title for the following abstract of a paper: {PAPER} <|im_end|>

B.6 LaMP-7: Personalized Tweet Paraphrasing

<|im_start|>system
With the given examples, paraphrase the following tweet without any explanation before or after it. <|im_end|>
<|im_start|>user
{RETRIEVED USER HISTORY}
Paraphrase the following tweet without any explanation before or after it: USER TWEET <|im_end|>

C Case Study

To further intuitively understand the personalization effectiveness of CDCDA-PLM, we conduct a case study for a user on the Personalized Scholarly Title Generation (LaMP-5) task, which tests the ability of models to capture stylistic patterns when generating scholarly titles based on the abstract of an article.

Figure 3 presents an example of a specific user. Note that, according to this user's historical data, they prefer to directly include the proposed method's name from the abstract as part of the title.

User Device (11003279)

User Query: Generate a title for the following abstract of a paper:

Abstract: Because of the large number of online games available nowadays, online game recommender systems are necessary for users and online game platforms. The former can discover more potential online games of their interests, and the latter can attract users to dwell longer in the platform. This paper investigates the characteristics of user behaviors with respect to the online games on the Steam platform. Based on the observations, we argue that a satisfying recommender system for online games is able to characterize: personalization, game contextualization and social connection. However, simultaneously solving all is rather challenging for game recommendation. … To this end, **we propose a Social-aware Contextualized Graph Neural Recommender System (SCGRec)**, which harnesses three perspectives to improve game recommendation. We conduct a comprehensive analysis of users 2019 online game behaviors, which motivates the necessity of handling those three characteristics in the online game recommendation.

Method:	**Response:**	**R-1**
Golden Answer	Large-scale Personalized Video Game Recommendation via Social-aware Contextualized Graph Neural Network	
M_{device}	A Comprehensive Analysis of User Online Game Behaviors for Satisfying Recommender Systems: Personalization, "Game Contextualization, and Social Connection"	0.13
+ **RAG**	Characterizing User Behaviors in Online Games through Social-Awareness	0.09
+ **Direct-FT**	A novel recommender system combining user behavior, context and social knowledge for online games.	0.07
CDCDA-PLM	Social-aware Contextualized Graph Neural Recommender System for Online Games	**0.43**

Figure 3: A case study in LaMP-5, which is the task of Personalized Scholarly Title Generation.

Table 6: The performance of CDCDA-PLM and baselines on LaMP benchmark for other users. The best performance of personalized on-device model M_{device} is highlighted in **bold** and the second best is underlined.

Tasks	**Metric**	**Non-Personalized**		**RAG**				**Speculative Decoding**	**Fine-tuning based M_{device}**			
		M_{cloud}	M_{device}	M_{cloud} +BM25	M_{device} +BM25	M_{cloud} +Contriever	M_{device} +Contriever	M_{device}	Direct -FT	EDA -FT	RKD -FT	CDCDA -PLM
LaMP-1	Accuracy ↑	0.480	0.500	0.530	0.480	0.550	0.500	0.480	0.520	0.520	0.470	**0.580**
	F1 ↑	0.466	0.469	0.487	0.324	0.502	0.333	0.442	0.473	0.513	0.435	**0.566**
LaMP-2	Accuracy ↑	0.155	0.010	0.224	0.035	0.255	0.052	0.169	0.072	0.103	0.110	**0.217**
	F1 ↑	0.139	0.013	0.233	0.044	0.258	0.081	0.114	0.065	0.076	0.070	**0.169**
LaMP-3	MAE ↓	1.100	0.790	1.540	1.410	0.950	1.010	0.580	0.580	0.480	0.610	**0.460**
	RMSE ↓	1.386	1.330	2.159	1.972	1.584	1.559	1.149	1.105	**0.980**	1.171	1.000
LaMP-4	ROUGE-1 ↑	0.133	0.128	0.151	0.122	0.159	0.122	0.145	0.136	0.111	**0.149**	0.148
	ROUGE-L ↑	0.117	0.113	0.134	0.109	0.143	0.110	0.125	0.124	0.099	0.131	**0.133**
	BERT-F1 ↑	0.843	0.841	0.848	0.842	0.850	0.841	0.844	0.844	0.833	0.846	**0.847**
LaMP-5	ROUGE-1 ↑	0.438	0.365	0.448	0.356	0.464	0.343	0.354	0.363	0.343	0.371	**0.376**
	ROUGE-L ↑	0.370	0.301	0.375	0.286	0.386	0.289	0.313	0.311	0.288	0.318	**0.324**
	BERT-F1 ↑	0.892	0.885	0.896	0.883	0.897	0.881	0.885	0.880	0.884	0.883	**0.889**
LaMP-7	ROUGE-1 ↑	0.352	0.257	0.323	0.214	0.326	0.225	0.330	0.294	0.321	0.332	**0.354**
	ROUGE-L ↑	0.295	0.213	0.271	0.182	0.276	0.196	0.269	0.241	0.272	0.277	**0.295**
	BERT-F1 ↑	0.884	0.862	0.882	0.856	0.884	0.858	0.881	0.877	0.881	0.882	**0.884**

In this case, they favor using the bold text "Social-aware Contextualized Graph Neural Recommender System" as indicated in the Golden Answer. However, all baseline models fail to capture this preference and instead generate titles by summarizing the abstract's semantics. Only our CDCDA-PLM successfully identifies this pattern, producing a title most similar to the Golden Answer.

D Performance of CDCDA-PLM and Baselines On Additional Users

This section reports the results of experiment on the additional users. In the previous experiment, Table 1, we selected the top 100 most active users followed by OPPU (Tan et al., 2024b) and Personalized Piece (Tan et al., 2024a). To better validate the effectiveness of our method, due to the limited computational resources, we conducted an additional experiment on all baselines using 100 randomly selected users from each task. Since these users were chosen at random, the results are expected to be representative of a broader user base. The experimental results, Table 6, demonstrate that our method outperforms nearly all baselines across all tasks on additional users.