The Whole is Better than the Sum: Using Aggregated Demonstrations in In-Context Learning for Sequential Recommendation

Anonymous ACL submission

Abstract

Large language models (LLMs) have shown excellent performance on various NLP tasks. To use LLMs as strong sequential recommenders, we explore the in-context learning approach to sequential recommendation. We investigate the effects of instruction format, task consistency, demonstration selection, and number of demonstrations. As increasing the number of demonstrations in ICL does not improve accuracy despite using a long prompt, we propose a novel method called LLMSRec-Syn that incorporates multiple demonstration users into one aggregated demonstration. Our experiments on three recommendation datasets show that LLMSRec-Syn outperforms state-of-the-art LLM-based sequential recommendation methods. In some cases, LLMSRec-Syn can perform on par with or even better than supervised learning methods. Our code will be publicly available.

1 Introduction

Motivation. Large language models (LLMs) are known to perform well as a zero-shot solution for many natural language processing tasks (Brown et al., 2020; Chowdhery et al., 2022; OpenAI, 2022; Qin et al., 2023). Recently, there are some works that focus on using LLMs to perform recommendation with promising accuracies (Hou et al., 2023; Wang and Lim, 2023; Liu et al., 2023a; Bao et al., 2023; Gao et al., 2023) and to provide explanations (Yang et al., 2023; Wang et al., 2023b). Most of these works developed LLM prompts for zero-shot sequential recommendation.

To investigate whether LLM can serve as a strong zero-shot sequential recommender, Hou et al. (2023) devised a prompt that is filled with historical items in chronological order, candidate items, and instruction to rank the candidate items. Wang and Lim (2023) proposed a three-step prompting method, where LLMs first summarizes the user preference based on the user's past interacted items. It then identifies representative items from the past interacted items that capture the user preference, and finally recommends items among the candidate items which are aligned with the representative items. Among the very few one-shot sequential recommendation works, Liu et al. (2023a) and Hou et al. (2023) explored in-context learning using the test user's second last item as the ground truth next-item and all earlier interacted items as input to create self-demonstrations. Nevertheless, previous experiments have shown that in-context learning (ICL) based sequential recommendation methods perform poorly compared with the supervised learning-based methods (e.g., SASRec) due to the complex recommendation task definition (Liu et al., 2023a; Hou et al., 2023; Wang and Lim, 2023). The illustrative comparison of these two methods is shown in Figure 1.

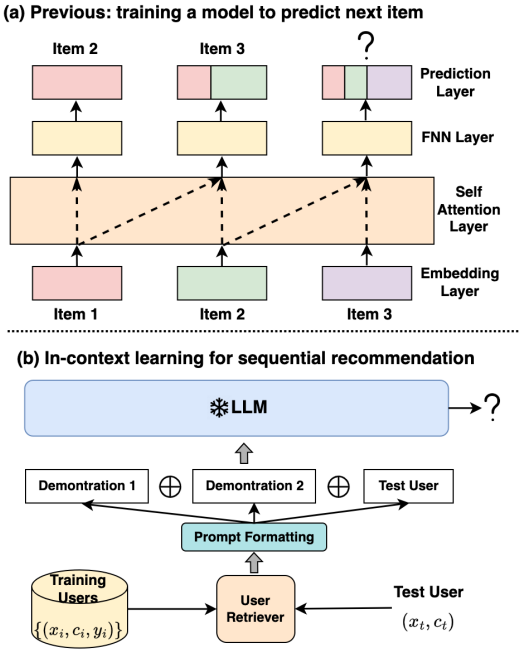

Figure 1: The illustrative comparison of (a) supervised sequential recommendation method and (b) in-context learning based sequential recommendation method.

To develop an effective in-context learning ap-

proach for LLMs to perform sequential recommendation, we first define the sequential recommendation problem as follows.

Problem definition. We denote each input user instance u_i to be a (x_i, c_i, y_i) tuple where x_i denotes the sequence of past interacted items (excluding y_i) by u_i, c_i denotes the candidate items to be recommended ($|c_i| = M$), and y_i denotes the ground truth next-item which is also the last item interacted by u_i. Note that y_i appears in c_i ($y_i \in c_i$). A LLM-based sequential recommendation method is required to assign a rank $rank(d) \in [1, M]$ to each item d in c_i. Our objective is to ensure that the method ranks y_i, i.e., $rank(y_i)$, as high as possible for all.

The above definition includes c_i as input as it is usually infeasible for LLMs to take all items as input due to limited prompt length. Moreover, having c_i does not introduce bias in the evaluation. The above definition is also adopted in Hou et al. (2023). We also assume a dataset of users' interacted item sequences from which we can construct demonstrations for ICL, and a LLM which is too large for pretraining or finetuning.

Overview of our study. Past works has shown that the effectiveness of ICL in adapting LLMs to new tasks is significantly influenced by instruction wording (Madaan and Yazdanbakhsh, 2022; Yang et al., 2023), label design (Yoo et al., 2022; Wei et al., 2023), selection of demonstrations (Liu et al., 2021; Shi et al., 2022; Zhang et al., 2023b), and number of demonstrations Chen et al. (2023); Zhao et al. (2023). Our study thus begins by systematically investigating how the instruction format, task consistency (between test and demonstration), demonstration selection, and the number of demonstrations affect ICL-based sequential recommendation. Through our preliminary experiments, we obtain four findings including the one that observes degradation of recommendation accuracy when the number of demonstrations increases. As each demonstration takes up significant length, it is also easy for multiple demonstrations to exceed the prompt limit of LLMs. Moreover, as LLMs are known to miss out relevant information in a long input prompt (Liu et al., 2023b), we thus embark on a follow-up study on designing a more efficient ICL scheme based on *aggregated demonstration*.

The key idea in aggregated demonstration is to combine multiple training users into one demonstration. This reduces the repetition of instruction text in the ICL prompt. It also seeks to summarize multiple training users relevant to the test instance in a compact manner. We also develop a novel ICL method using aggregated demonstration for sequential recommendation known as **LLMSRec-Syn**. The length of LLMSRec-Syn prompt increases only gradually with number demonstration users, LLMSRec-Syn can cope with more relevant information from the demonstration users within a concise input context. We finally show LLMSRec-Syn outperforms other zero-shot and one-shot ICL methods in an extensive set of experiments.

Contribution. Our contributions can be summarized as follows: (1) We systematically explore the ICL approach to sequential recommendation by empirically investigating the effect of instruction format, task consistency, demonstration selection, and number of demonstrations; (2) We propose a new in-context learning method for sequential recommendation called LLMSRec-Syn which leverages on an novel concept of aggregated demonstration; (3) We experiment on three popular recommendation datasets and show that LLMSRec-Syn outperforms previous LLM-based sequential recommendation methods.

2 Related Work

In-Context Learning. Several works show that LLMs can effectively adapt to different NLP and multimodal tasks, including machine translation (Agrawal et al., 2022), visual question answering (Yang et al., 2022), and foreground segmentation (Zhang et al., 2023b). This adaptation is achieved by learning from a few task-relevant demonstrations, commonly known as in-context learning (ICL) (Brown et al., 2020). Despite the above successes, ICL's performance is still significantly affected by the wording of instructions (Madaan and Yazdanbakhsh, 2022; Yang et al., 2023), label design (Yoo et al., 2022; Wei et al., 2023), demonstration selection (Liu et al., 2021; Shi et al., 2022; Zhang et al., 2023b), and number of demonstrations Chen et al. (2023); Zhao et al. (2023). ICL is much less studied in LLM-based sequential recommendation. As sequential recommendation is distinct from the pretraining tasks of LLMs and also different from the above-mentioned tasks, new designs of demonstration(s) and ICL prompt is necessary.

LLMs for Sequential Recommendation. Early sequential recommendation works adopt techniques such as Markov Chains (Rendle et al., 2010;

Table 1: Dataset statistics after removing duplicate interactions and users or items with fewer than 5 interactions.

Datasets	ML-1M	LastFM-2K	Games
# Users	6,040	1,143	50,547
# Items	3,706	11,854	16,859
# User-item Interactions	1,000,209	68,436	389,718
Avg. interacted items per user	165.59	59.92	7.71
Avg. interacted users per item	269.88	5.77	23.11

He and McAuley, 2016) and neural networks (e.g., RNN (Hidasi et al., 2015), CNN (Tang and Wang, 2018), Self-Attention (Kang and McAuley, 2018), and GNN (Chang et al., 2021)). To investigate if LLMs can be used as effective sequential recommenders without training, Hou et al. (2023) formulated sequential recommendation as conditional ranking, employing zero-shot LLM methods to reflect user preferences from past interactions and recency. Wang and Lim (2023) developed a three-step LLM prompting to summarize user preferences, while Hou et al. (2023) and Liu et al. (2023a) introduced a one-shot ICL method that utilizes the previous item interactions of the target user as a demonstration. To address position bias, Hou et al. (2023) proposed to randomize the candidate item order. In this work, we explore using training data demonstrations, not just user own history, and introduce aggregated demonstration for combining relevant users.

3 What Makes In-Context Learning Work for Sequential Recommendation

In this section, we conduct a preliminary empirical study to investigate the role of various aspects of demonstrations. These aspects include the wording of prompts, task consistency between demonstrations and test instances, selection of demonstrations, and number of demonstrations. While previous studies have explored the use of LLM as sequential recommenders in a zero-shot manner (Hou et al., 2023; Wang and Lim, 2023), this is the first study to comprehensively discuss how in-context learning can improve sequential recommendation.

3.1 Experiment Setup

We implement zero-shot, one-shot, and few-shot methods in this study, using three widely used recommendation datasets: the movie rating dataset *MovieLens-1M* (ML-1M) dataset, the category of *Games* from the Amazon Review dataset (McAuley et al., 2015), and the music artist listening dataset *LastFM-2K* (Cantador et al., 2011). The data statistics are summarized in Table 1. Taking into account cost-effectiveness of LLMs, we select 50 data examples from each of the three datasets to carry out all experiments for analysis in Section 3. Following the previous works (Hou et al., 2023; Wang and Lim, 2023), we use a leave-one-out strategy for evaluation, i.e., predicting the last interacted item of each user sequence and using the earlier interacted items as input. For each user sequence, we remove the last item, keeping it aside for testing. The rest of the sequence is used for training and validation. To evaluate the ranking results for each user u_i over a set of candidate items c_i, we adopt the widely used **NDCG@N** (N = 10, 20) as the evaluation metric. We use ChatGPT (`GPT-3.5-Turbo`) as the default LLM due to its excellent performance and cost-effectiveness. To ensure the reliability of findings, we repeat each experiment 9 times and report the average results. Without exception, we use ML-1M as an example for discussion.

3.2 In-Context Learning for Sequential Recommendation

In ICL for sequential recommendation, one or a few training users are used as demonstrations that are included in the LLM prompt. Each demonstration thus includes a training user i's historical item interactions x_i, a set of candidates c_i, and ground truth next-item y_i. We denote the prompt capturing the demonstration user i by $\mathcal{T}(x_i, c_i, y_i)$. The following shows the concatenation of n demonstrations $\mathcal{C}$ which is appended by the instruction prompt for the test user $\mathcal{T}(x_{test}, y_{test})$ for prediction.

$$\mathcal{C} = \mathcal{T}(x_1, c_1, y_1) \oplus \cdots \oplus \mathcal{T}(x_n, c_n, y_n) \quad (1)$$

$$y_{\text{test}} \sim \mathcal{P}_{LLM}(\cdot \mid \mathcal{C} \oplus \mathcal{T}(x_{\text{test}}, c_{\text{test},\cdot})) \quad (2)$$

3.3 Wording of Instructions

LLMs have been found to be sensitive to wording of the prompt (Madaan and Yazdanbakhsh, 2022; Yang et al., 2023). For example, prompts (or instructions) that are semantically similar may yield significantly different results (Kojima et al., 2022; Zhou et al., 2022; Wang et al., 2023a; Zhang et al., 2023a). To examine the impact of instruction wording and exclude the influence of other factors such as demonstration labels and selection, we employ LLM as a zero-shot solver for sequential recommendation.

We discuss four different options for the instruction format to investigate the sensitivity of the LLM to the wording of the instruction. Considering the prompts used in LLM-based zero-shot recommendation models (Hou et al., 2023; Wang and Lim,

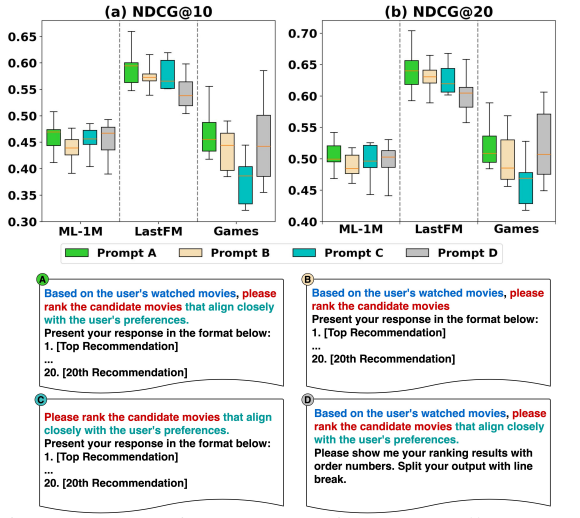

Figure 2: Instruction Format options: (A) Full, (B) w/o preference alignment, (C) w/o watched movie focus, (D) w/o rank result format

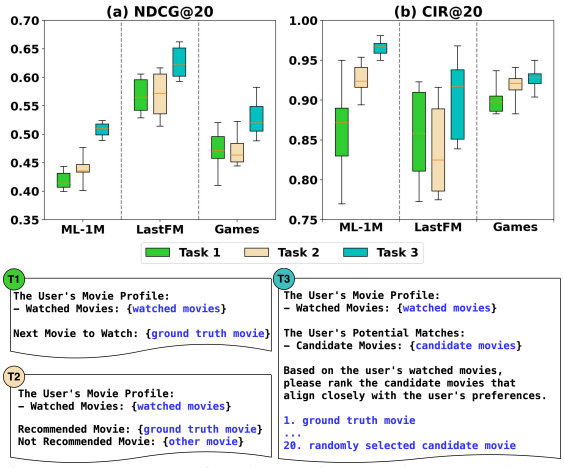

Figure 3: Impact of task consistency between demonstrations and test instances. CIR: Candidate Inclusion Ratio of Demonstration Templates: (T1) Next-Item option; (T2) Contrasting Item Pair option; (T3) Ranked Items option.

2023), we derive instructions with four possible mention components: (a) candidate item ranking, (b) user preference alignment, (c) historical interacted items, and (d) ranked result format. As recommendation is formulated as a ranking task, component (b) is mandatory. The *full* instruction covers all four components. To explore better instructions, we derive other instruction options by leaving out one of the remaining components. We thus have four instruction options: (A) full instruction $\mathcal{T}^{\text{A}}$, (B) full instruction without (b) $\mathcal{T}^{\text{B}}$, (C) full instruction without (c) $\mathcal{T}^{\text{C}}$, and (D) full instruction with (d) replaced by textual result table description $\mathcal{T}^{\text{D}}$ as shown in Figure 2.

As shown in Figure 2, we observe that ChatGPT's performance degrades when the instruction does not make reference to interacted items or user preferences across three datasets. This suggests that explicit inclusion of watched movies or user preferences can improve its ability to leverage the user's historical items effectively. While Instruction (A) shows similar average performance as Instruction (D) on ML-1M and LastFM, the former enjoys a smaller variance and outperforms the latter on LastFm. This suggests that LLM prefers explicit output formats over textual description of output format.

> ***Finding* 1.** For sequential recommendation, ChatGPT prefers explicit mentions of instructions and explicit mentions of interacted items, user preference alignment and ranked result format.

3.4 Task Consistency

LLMs are capable of learning new tasks at test time by understanding the relationship between the input of a demonstration and its corresponding output label (Yoo et al., 2022; Wei et al., 2023). In sequential recommendation, LLM is required to rank the ground truth target item at the top followed by other candidate items. However, in a demonstration example from the training set, we observe only one labeled next item but not the ranking of other candidate items. Hence, when constructing demonstrations for in-context learning, we have to answer the important questions: How to prepare the input-label correspondence for a demonstration to be consistent with the sequential recommendation task? To eliminate other factors that may influence the results, such as the number of demonstrations and instructions, we employ instruction (A) as it has proven to be the most effective and robust across three datasets in our previous experiments. We randomly select only one demonstration example for all experiments in this study.

In traditional sequential recommendation, next-item prediction (Song et al., 2021; Petrov and Macdonald, 2022), positive and negative item comparison (Rendle et al., 2012; Kang and McAuley, 2018; Xie et al., 2020), and reranking (Xu et al., 2023) are commonly utilized objectives to train models. Hence, we develop three different prediction tasks for demonstrations for in-context learning. These tasks include: (T1) predicting the next item, (T2) contrasting item pairs, and (T3) ranking candidate items. The prompts corresponding to these prediction tasks are shown in Figure 3. T1 uses the

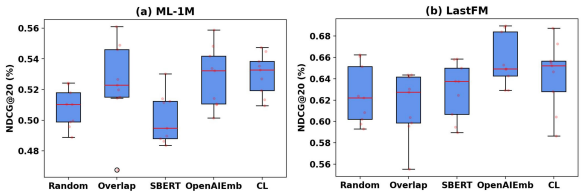

Figure 4: Demonstration selection: (1) random selection; (2) overlapping interacted items; (3) cosine similarity between the SBERT embeddings of interacted item sequences; (4) cosine similarity between the OpenAI embeddings of interacted item sequences; (5) cosine similarity using CL embeddings of interacted item sequences.

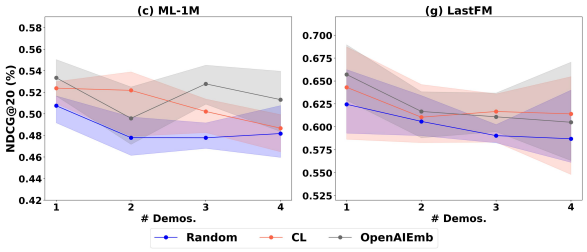

Figure 5: Varying number of demonstrations.

ground truth next-item directly in the demonstration. T2 uses the ground truth next item and another randomly selected item as the positive and negative items respectively. T3 ranks the ground truth next item at the first position and randomly shuffles the remaining candidate items to fill the other positions. Among the task prediction task options, T3 is the only one that aligns closely with the instruction for the test user, i.e., (A).

Figure 3 shows the results of these three tasks across three datasets. T3 consistently outperforms T1 and T2 on all three datasets, suggesting that task consistency between demonstration and test user benefits in-context learning for sequential recommendation. Additionally, As the recommended items may not be found among the provided candidates, we also report *candidate inclusion ratio* (CIR) which measures the proportion of the candidate items that appear in the ranked item results. As shown in Figure 3, we observe that the CIR generally correlates with the NDCG results. The inconsistent demonstration task options (e.g., T1 and T2 coupled with test instruction option (A)) are more likely to cause the LLM to generate non-candidate items in the results. This helps to understand why T3 achieves the best performance.

> ***Finding* 2.** Maintaining task consistency between demonstrations and test users is beneficial for in-context learning in sequential recommendation.

3.5 Selection of Demonstrations

It has been observed that the performance of in-context learning greatly depends on selecting suitable demonstrations (Liu et al., 2021). Utilizing examples that are semantically similar to the test sample can provide more informative and task-relevant knowledge to LLMs. Following Liu et al. (2021), there are several follow-up works (Rubin et al., 2021; Shi et al., 2022; Zhang et al., 2023b; Li et al., 2023) to develop methods for selecting better demonstrations. In this work, we evaluate five different demonstration selection methods to determine their impact to in-context learning for sequential recommendation. These methods include: (1) random selection; (2) overlapping historical items of demonstration user and test user; (3) text similarity scores using Sentence-BERT embedding (Reimers and Gurevych, 2019) (SBERT); (4) text similarity scores using OpenAI embedding[1] (OpenAIEmb); and (5) trained retriever using contrastive learning (Xie et al., 2020; Li et al., 2023) (CL). Other than random section, the other five options are also called the retrieval-based methods. In Option (5), positive examples are obtained by data augmentation applied to the anchor user sequence, while negative examples are randomly selected user item-interaction sequences.

Figure 4 compares the five selection methods on ML-1M and LastFM as they are used in one-shot sequential recommendation. The results show that selection methods (4) and (5) generally outperform the rest. As method (4) appears to be more robust than (5) and it does not require additional training, we thus use that as the default retriever model in the subsequent experiments.

> ***Finding* 3.** Retrieval-based methods are better than random selection, and stronger LLMs can serve as stronger retrievers without any training.

3.6 Number of Demonstrations

When training a model, having more training data examples usually leads to better model performance. However, it is interesting to note that Zhao et al. (2023) discover that increasing the number

[1] `text-embedding-ada-002` (https://platform.openai.com/docs/models/moderation)

of demonstrations for in-context learning does not necessarily result in improved performance. Similarly, Chen et al. (2023) finds that using only one demonstration may not perform worse than using more demonstrations. In our case, we evaluate the impact of the number of demonstrations on ML-1M and LastFM using random selection, OpenAIEmb, and CL demonstration selection methods. We conduct experiments with the number of demonstrations ranging from 1 to 4, as exceeding 4 demonstrations would exceed the input limit of ChatGPT (`GPT-3.5-Turbo`). Figure 5 demonstrates a clear trend of performance decreasing with number of demonstrations.

> ***Finding* 4.** Increasing demonstrations for in-context learning for sequential recommendation would result in performance degradation and exceed the input limit of LLMs.

4 In-Context Learning with Aggregated Demonstrations

Finding 4 suggests LLMs have difficulties coping with multiple demonstrations in sequential recommendation. A similar finding by Liu et al. (2023b) also suggests that the current language models often struggle to utilize information in long input contexts. In particular, their performance tends to significantly degrade when the relevant information is located in the middle of long contexts, also known as the "lost in the middle" phenomenon. The in-context learning prompts for sequential recommendation can easily exceed the prompt length limit of LLM when more than 4 demonstrations are to be accommodated. Such prompts not only suffer from "lost in the Middle", but also incur additional costs of calling LLM APIs.

To address the above challenge, we propose *aggregated demonstration* which combines K ($K > 1$) demonstration users into one for in-context learning. This simple yet effective in-context learning method for sequential recommendation is called **LLMSRec-Syn**. As the prompt length of aggregated demonstration only increases marginally when we increase K, LLMSRec-Syn can accommodate more member demonstration users.

Based on Finding 3, LLMSRec-Syn begins with selecting K demonstration users that are similar to the test user. We use similarity between the OpenAIEmb embeddings of demonstration and test users. We also follow Finding 1 and adopt instruction template (A) for the test user. Based on Finding 2, we also adopt demonstrate template (T3) for the aggregated demonstration to maintain consistency with the task for test user. Next, we construct the aggregated demonstration's historical item-interactions, candidate items, and the desired ranking of the candidate items from its member demonstrations.

Historical item-interactions. Let H denote the historical item-interactions and H is empty initially. We first rank the K selected demonstration users by similarity score. We then add the most recent interacted item from the most similar demonstration to H. We repeat the same step for the remaining demonstrations in their similarity order. When we run out of most recent interacted items from K selected demonstrations, we continue to add the next recent interacted items of these demonstrations to H until the number of historical items reaches MAX_H.

Candidate Items. Let C denote the candidate items of the aggregated demonstration and C is empty initially. We first gather all the ground truth next items from the K selected demonstrations and add them to C. Next, we randomly add other items from the item pool to C so as to meet the required number of candidate items.

Ranking of Candidate Items. To rank the candidate items in C, we place the ground truth next item of the most similar demonstration at rank 1, followed by that of next similar demonstration until we run out of the ground truth next items of all K selected demonstrations. Next, we assign random ranks to the remaining items in C.

Once the aggregated demonstration is constructed, it is added to the prompt the same way a training user is added as a demonstration. we add it to the corresponding test user and use them as input for the LLM.

$$\mathcal{C}^{\textbf{agg}} = \mathcal{T}^{\text{A}}\left(\text{Agg}^{\text{T3}}\left(x_{\sigma_1}, c_{\sigma_1}, y_{\sigma_1}, \cdots, x_{\sigma_n}, c_{\sigma_n}, y_{\sigma_n}\right)\right), \tag{3}$$

$$y_{\text{test}} \sim \mathcal{P}_{LLM}\left(\cdot \mid \mathcal{C}^{\textbf{agg}} \oplus \mathcal{T}^{\text{A}}\left(x_{\text{test}}, c_{\text{test}}\right)\right), \tag{4}$$

where σ_i represents the ith ranked selected users returned by the retrieval model. Finally, the LLM generates a ranked list of candidate items as the recommendation result.

There are several advantages of the proposed LLMSRec-Syn: 1) Standard demonstration only has one ground truth next item in the ranking list. In contrast, the aggregated demonstration includes more next items at high positions in the ranking list.

This approach can avoid sparse signals and provide more guidance to LLMs for recommending to the test user; 2) LLMSRec-Syn is less sensitive to the number of demonstrations; 3) Cost of LLMSRec-Syn does not increase much with the number of demonstrations; and 4) LLMSRec-Syn keeps to the prompt length limit of LLMs.

5 Experiments and Results

5.1 Methods for Comparison

To evaluate the performance of LLMSRec-Syn, we conduct an extensive set of experiments on ML-1M, Games, and LastFM-2K datasets. Following Hou et al. (2023), we select 200 data examples from each of the three datasets to carry out all experiments. We use an experiment setup similar to that mentioned in Section 3.1 except that we now uses more LLMs and reports the NDCG@N results where N=5,10, and 20. We compare LLMSRec-Syn with 10 methods categorized into 3 types:

Supervised methods: Most Popular (Recommending items based on their overall popularity among all users in the training data), GRU4Rec (Hidasi et al., 2015) (using GRUs to model user's item sequences), and SASRec (Kang and McAuley, 2018) (employing a self-attention mechanism to learn user preferences from their item sequences).

Zero-shot methods: BM25 (Robertson et al., 2009) (ranking candidate items based on their textual similarity with the test user's interacted items), LLMSeqSim (Harte et al., 2023) (ranking candidate items by semantically similarity using OpenAI embeddings (`text-embedding-ada-002`)), LLMRank-Seq (Hou et al., 2023) (using ChatGPT to rank candidate items with crafted prompts), and LLMSRec (a zero-shot version of the proposed LLMSRec-Syn using the instruction prompt $\mathcal{T}^{\text{A}}$).

One-shot methods: LLMRank-His (Hou et al., 2023) (using historical items of the test user to form a demonstration), LLMSRec-Fixed (using a randomly selected demonstration for all test users), and LLMSRec-Nearest (finding the most similar training user as the demonstration).

As Section 3.6 shows that more than one demonstration in in-context learning for sequential recommendation does not yield better performance, we do not include few-shot methods in this set of experiments. We however will study how many member demonstrations K is ideal for aggregated demonstration (see Section 5.2).

We implement LLMSRec-Syn using three different LLMs, LLaMa2 (Touvron et al., 2023), ChatGPT (OpenAI, 2022) (LLMSRec-Syn), and GPT-4 (OpenAI, 2023) (LLMSRec-Syn-4). However, we found LLaMa2 unable to follow recommendation instructions and is prone to generating historical interacted items or in-context examples. As a result, we exclude the LLaMa2 results In LLMSRec-Syn, we set the number of member users in the aggregated demonstration as {1,2,3,4,5,6,7} and conduct a brute force search to determine the optimal number for each dataset. We set the number of historical items $MAX_H = 50$ and number of candidate items to 20. We analyse some specific test cases of LLMSRec-Syn-4 in the Appendix 8.2.

5.2 Main Results

The main experiment results are shown in Table 2, from which we obtain the following findings:

ICL one-shot methods with appropriate demonstrations out-perform zero-shot methods. As shown in Table 2, LLMRank-His, LLMSRec-Fixed, and LLMSRec-Nearest using one training user as demonstration outperform LLMRank-Seq on three datasets, except for LLMSRec-Fixed which performs slightly worse than LLMRank-Seq on LastFM-2K. This result suggests that ICL can enhance the LLM's ability to perform a complex task such as sequential recommendation.

Aggregated demonstration, combining multiple member users, allows LLM to effectively gather useful task specific information about the test user within a conise context. Compared to other ICL baselines (i.e., LLMRank-His, LLMSRec-Fixed, and LLMSRec-Nearest), LLMSRec-Syn achieves the superior one-shot performance across all datasets as shown in Table 2. While Figure 5 shows that having more demonstrations may hurt ICL for sequential recommendation, the idea of incorporating multiple demonstration users into an aggregated demonstration enhances the performance of LLMSRec-Syn. These results illustrate the advantage of aggregated demonstration in accommodating multiple training users within a limited prompt length.

LLMSRec-Syn is competitive against supervised methods when the amount of training data is limited. LLMSRec-Syn easily outperforms the simple supervised baseline, Most Popular. While it does not outperform GRU4Rec and SASRec on ML-1M and Games, LLMSRec-Syn surprisingly outperforms all supervised baselines on LastFM-2K. One possible reason is that LastFM-2K has

Table 2: Main results. We report NDCG@5, NDCG@10 and NDCG@20 on ML-1M, LastFM-2K and Games. (Best results in each group of methods are **boldfaced** and overall best results are underlined).

Setting	Method	ML-1M			LastFM-2K			Games		
		NDCG@5	NDCG@10	NDCG@20	NDCG@5	NDCG@10	NDCG@20	NDCG@5	NDCG@10	NDCG@20
Supervised	Most Popular	0.3673	0.4623	0.4748	0.4055	0.4205	0.4803	0.2746	0.3905	0.4496
	GRU4Rec	0.7205	0.7494	0.7610	0.3382	0.3971	0.4784	0.6747	0.7002	0.7278
	SASRec	**0.7322**	**0.7595**	**0.7702**	**0.4081**	**0.4680**	**0.5303**	**0.6828**	**0.7189**	**0.7311**
Zero-shot	BM25	0.1314	0.2053	0.3370	0.1215	0.1393	0.3354	0.2285	0.3108	0.4055
	LLMSeqSim	0.3250	0.4037	**0.4723**	0.4090	0.4662	0.5293	**0.4269**	**0.4830**	**0.5360**
	LLMRank-Seq	**0.3344**	0.3882	0.4612	0.5084	0.5545	**0.6070**	0.3063	0.3607	0.4074
	LLMSRec	0.3339	**0.4087**	**0.4723**	**0.5126**	**0.5602**	0.6057	0.4070	0.4555	0.5103
One-shot	LLMRank-His	0.3919	0.4444	0.5074	0.5318	0.5725	0.6212	0.4191	0.4667	0.5206
	LLMSRec-Fixed	0.3590	0.4193	0.4793	0.4961	0.5425	0.5984	0.3744	0.4400	0.4899
	LLMSRec-Nearest	0.3842	0.4382	0.5017	0.5249	0.5697	0.6197	0.3975	0.4388	0.4994
	LLMSRec-Syn	0.4267	0.4813	0.5334	0.5554	0.5918	0.6371	0.4989	0.5334	0.5869
	LLMSRec-Syn-4	**0.5112**	**0.5685**	**0.5936**	**0.6544**	**0.6799**	**0.7017**	**0.5647**	**0.6019**	**0.6277**

sparse information about items after removing duplicate user-item interactions and users/items with less than 5 interactions, making it challenging to train a good supervised model.

LLMSRec-Syn using more powerful LLMs may outperform supervised methods in the future. With rapid advancement of LLM research, LLMSRec-Syn can be further enhanced when more powerful LLM is used. Our results in Table 2 shows that LLMSRec-Syn-4 significantly outperforms LLMSRec-Syn on all the 3 datasets.

5.3 Analysis of Aggregated Demonstrations

In this section, we study the recommendation performance when varying the settings of aggregated demonstrations. Analysis of ordering of users and label in the aggregated demonstration can be found in the Appendix 8.1.

Impact of number of users in the aggregated demonstration. We evaluate the impact of K (the number of member users) in the aggregated demonstration on LLMSRec-Syn's performance. We empirically vary K from 2 to 7. As shown in Figure 6, an approximate inverted U-shaped relationship exists between K and NDCG@10/20 performance. Initially, as K increases, there is a noticeable performance increase, suggesting that LLMSRec-Syn benefits from aggregated demonstration. However, beyond some K value, more member users in aggregated demonstration leads to lower performance. This can be explained by more irrelevant training users being incorporated into the aggregated demonstration.

Impact of number of aggregated demonstrations. We evaluate the impact of the number of aggregated demonstrations to LLMSRec-Syn by varying the number of aggregated demonstrations from 1 to 4 such that each demonstration involves 2 users (see Figure 7(a)) and 3 users (see Figure 7(b)). For the Games dataset, experimentation with 3 aggregated demonstrations was not possible due to GPT-3.5-Turbo's input limit. The results show that a single aggregated demonstration outperforms multiple ones, except in the LastFM-2K dataset, where two demonstrations slightly excel.

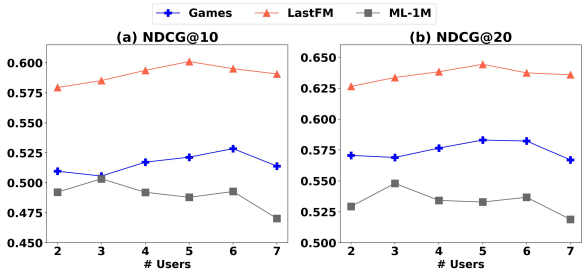

Figure 6: Varying number of users (K) in aggregated demonstration.

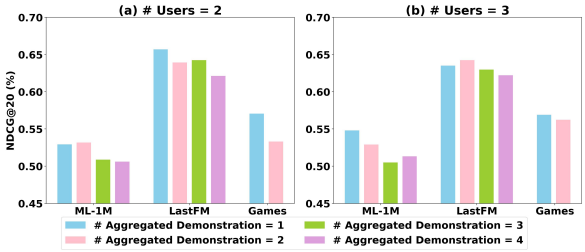

Figure 7: Varying number of aggregated demonstrations each with: (a) 2 member users, and (b) 3 member users.

6 Conclusion

This paper investigates in-context learning (ICL) for LLM-based sequential recommendation. Our study identifies key factors such as instruction format and demonstration selection that influence ICL's effectiveness. We further introduce the LLMSRec-Syn method which utilizes our proposed aggregated demonstration to efficiently incorporate relevant information from multiple training users. Tested on three datasets, LLMSRec-Syn consistently outperforms existing LLM-based sequential recommendation methods. Future work includes a detailed analysis of LLMSRec-Syn's unexpected success compared to some supervised methods and the optimization of aggregated demonstration strategies.

7 Limitations

While this paper considers several factors in applying LLMs to sequential recommendation and proposes a new demonstration concept known as aggregated demonstration, there are still some limitations yet to be addressed. Firstly, the wording of LLMSRec-Syn prompt is manually handcrafted and may not be optimal. Moreover, when a user's historical items are too many, LLMSRec-Syn may still suffer from the issue of long text. Furthermore, the aggregated demonstration method, while mitigating input length constraints, might oversimplify the user preferences, potentially resulting in less personalized recommendations. Moreover, the non-utilization of existing user datasets for pretraining or fine-tuning, due to LLM size constraints, limits the adaptability and fine-tuning of the model to specific recommendation contexts. These limitations highlight the need for further research in optimizing LLMs for complex, dynamic tasks such as sequential recommendation, where user context and historical data play crucial roles.

References

Sweta Agrawal, Chunting Zhou, Mike Lewis, Luke Zettlemoyer, and Marjan Ghazvininejad. 2022. In-context examples selection for machine translation. *arXiv preprint arXiv:2212.02437*.

Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. *arXiv preprint arXiv:2305.00447*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2011. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA. ACM.

Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2021. Sequential recommendation with graph neural networks. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 378–387.

Jiuhai Chen, Lichang Chen, Chen Zhu, and Tianyi Zhou. 2023. How many demonstrations do you need for in-context learning? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11149–11159.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. 2023. Chat-rec: Towards interactive and explainable llms-augmented recommender system. *arXiv preprint arXiv:2303.14524*.

Jesse Harte, Wouter Zorgdrager, Panos Louridas, Asterios Katsifodimos, Dietmar Jannach, and Marios Fragkoulis. 2023. Leveraging large language models for sequential recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 1096–1102.

Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 191–200. IEEE.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2023. Large language models are zero-shot rankers for recommender systems. *arXiv preprint arXiv:2305.08845*.

Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206. IEEE.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. 2023. Unified demonstration retriever for in-context learning. *arXiv preprint arXiv:2305.04320*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

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

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023b. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*.

Aman Madaan and Amir Yazdanbakhsh. 2022. Text and patterns: For effective chain of thought, it takes two to tango. *arXiv preprint arXiv:2209.07686*.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

OpenAI. 2022. Introducing chatgpt. https://openai.com/blog/chatgpt.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Aleksandr Petrov and Craig Macdonald. 2022. Effective and efficient training for sequential recommendation using recency sampling. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 81–91.

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. Bpr: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*.

Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 811–820.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*.

Peng Shi, Rui Zhang, He Bai, and Jimmy Lin. 2022. Xricl: Cross-lingual retrieval-augmented in-context learning for cross-lingual text-to-sql semantic parsing. *arXiv preprint arXiv:2210.13693*.

Wenzhuo Song, Shoujin Wang, Yan Wang, and Shengsheng Wang. 2021. Next-item recommendations in short sessions. In *Proceedings of the 15th ACM Conference on Recommender Systems*, pages 282–291.

Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*, pages 565–573.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Lei Wang and Ee-Peng Lim. 2023. Zero-shot next-item recommendation using large pretrained language models. *arXiv preprint arXiv:2304.03153*.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. *arXiv preprint arXiv:2305.04091*.

Lei Wang, Songheng Zhang, Yun Wang, Ee-Peng Lim, and Yong Wang. 2023b. Llm4vis: Explainable visualization recommendation using chatgpt. *arXiv preprint arXiv:2310.07652*.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. 2023. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*.

Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Bolin Ding, and Bin Cui. 2020. Contrastive learning for sequential recommendation. *arXiv preprint arXiv:2010.14395*.

Yue Xu, Hao Chen, Zefan Wang, Jianwen Yin, Qijie Shen, Dimin Wang, Feiran Huang, Lixiang Lai, Tao Zhuang, Junfeng Ge, et al. 2023. Multi-factor sequential re-ranking with perception-aware diversification. *arXiv preprint arXiv:2305.12420*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2023. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*.

Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. 2022. An empirical study of gpt-3 for few-shot knowledge-based vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3081–3089.

Kang Min Yoo, Junyeob Kim, Hyuhng Joon Kim, Hyunsoo Cho, Hwiyeol Jo, Sang-Woo Lee, Sang-goo Lee, and Taeuk Kim. 2022. Ground-truth labels matter: A deeper look into input-label demonstrations. *arXiv preprint arXiv:2205.12685*.

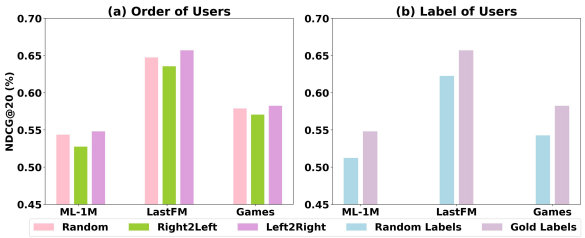

Figure 8: (a) Ordering of member users in the aggregated demonstration. (b) Ground truth vs random next-items in aggregated demonstrations.

Tianjun Zhang, Xuezhi Wang, Denny Zhou, Dale Schuurmans, and Joseph E Gonzalez. 2023a. Tempera: Test-time prompt editing via reinforcement learning. In *The Eleventh International Conference on Learning Representations*.

Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. 2023b. What makes good examples for visual in-context learning? *arXiv preprint arXiv:2301.13670*.

Fei Zhao, Taotian Pang, Zhen Wu, Zheng Ma, Shujian Huang, and Xinyu Dai. 2023. Dynamic demonstrations controller for in-context learning. *arXiv preprint arXiv:2310.00385*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

8 Appendix

8.1 More In-Depth Analysis

Impact of user order in aggregated demonstrations. We experiment with 3 possible orders of member users: (i) Random (randomly selects historical items and next-items from the selected users to construct the aggregated demonstration), (ii) Right2Left (the reverse order of demonstration users in constructing an aggregated demonstration in LLMSRec-Syn in Section 4), and (iii) Left2Right (the user order used in the LLMSRec-Syn). Figure 8(a) illustrates that Left2Right and Right2Left are the most and least ideal orders respectively. The performance of Random is naturally sandwiched in between.

Impact of labeled next-items in the aggregated demonstration. According to Min et al. (2022), ground truth labels are not important for in-context learning. To investigate this claim for ICL-based sequential recommendation, we compare LLMSRec-Syn using ground truth next-items in the aggregated demonstration (referred to as "Gold Labels") with that using random non-ground truth next-items (referred to as "Random Labels"). Our results in Figure 8(b) clearly indicate that ground truth next-items are required to yield better performance contradicting the claim by Min et al. (2022). This could possibly be explained by the complexity of sequential recommendation task.

8.2 Case Study

Table 3: Example of LLMSRec-Syn-4 on the ML-1M dataset. LLMSRec-Syn-4 would leverage patterns in movie preferences, like genre diversity, popularity, era, and theme loyalty, from the demonstration example to inform rankings in the test example. This user profile includes a mix of action, comedy, and drama, with movies like **"Fight Club"**, **"The Sixth Sense"**, and **"Terminator 2: Judgment Day"**. Ground truth next movie **"10 Things I Hate About You"** fits well into this mix, being a popular comedy-drama from a similar era.

Aggregated Demonstration Example:
The User's Movie Profile:
- Watched Movies: ['0. The Perfect Storm', '1. Toy Story 2', '2. Dunston Checks In', '3. Sleepy Hollow', '4. X-Men', ..., '49. Being John Malkovich']
The User's Potential Matches:
- Candidate Movies: ['0. Toy Story 2', '1. The Toxic Avenger Part III: The Last Temptation of Toxie', ..., '19. Lovers of the Arctic Circle, The (Los Amantes del Círculo Polar)']
Based on the user's watched movies, please rank the candidate movies that align closely with the user's preferences.
- You ONLY rank the given Candidate Movies.
- You DO NOT generate movies from Watched Movies.
Present your response in the format below: 1. [Top Recommendation (Candidate Movie)] 2. [2nd Recommendation (Candidate Movie)] ... 20. [20th Recommendation (Candidate Movie)]
Answer: 1. Gone in 60 Seconds 2. Wild Wild West 3. Casper 4. Stigmata 5. Toy Story 2 6. Andre 7. The Way of the Gun 8. Friday the 13th Part VI: Jason Lives 9. Drowning Mona 10. Carlito's Way 11. Duel in the Sun 12. Priest 13. The Candidate 14. Class Reunion 15. Cat People 16. The Toxic Avenger Part III: The Last Temptation of Toxie 17. Force 10 from Navarone 18. Lovers of the Arctic Circle, The (Los Amantes del Círculo Polar) 19. Time Regained (Le Temps Retrouvé) 20. The War Room

---

Test User:
Learn from the above demonstration examples to solve the following test example.
The User's Movie Profile:
- Watched Movies: ['0. Buffy the Vampire Slayer', '1. Bad Girls', ..., '12. **Fight Club**', ..., '32. **Terminator 2: Judgment Day**', ..., '36. **The Sixth Sense**', ..., '49. 200 Cigarettes']
The User's Potential Matches:
- Candidate Movies: ['0. Deterrence', '1. My Fellow Americans', '2. The Longest Day', '3. The Naked Man', '4. Terminal Velocity', '5. Police Story 4: Project S (Chao ji ji hua)', '6. The Original Kings of Comedy', '7. Speed', '8. Forces of Nature', '9. Jerry Maguire', '10. Inspector Gadget', '11. 10 Things I Hate About You', '12. Bad Company', '13. Police Academy 3: Back in Training', '14. Communion (a.k.a. Alice, Sweet Alice/Holy Terror)', '15. In & Out', '16. Five Easy Pieces', '17. Species II', '18. What Happened Was...', '19. Shallow Grave']
Based on the user's watched movies, please rank the candidate movies that align closely with John's preferences.
- You ONLY rank the given Candidate Movies.
- You DO NOT generate movies from Watched Movies.
Present your response in the format below: 1. [Top Recommendation (Candidate Movie)] 2. [2nd Recommendation (Candidate Movie)] ... 20. [20th Recommendation (Candidate Movie)]
Answer:
1. Jerry Maguire 2. Speed 3. **10 Things I Hate About You** 4. Forces of Nature 5. Inspector Gadget 6. My Fellow Americans 7. The Longest Day 8. Police Academy 3: Back in Training 9. In & Out 10. Police Story 4: Project S (Chao ji ji hua) 11. The Original Kings of Comedy 12. Bad Company 13. Terminal Velocity 14. Species II 15. Deterrence 16. Shallow Grave 17. Five Easy Pieces 18. What Happened Was... 19. Communion (a.k.a. Alice, Sweet Alice/Holy Terror) 20. The Naked Man

---

Table 4: Example of LLMSRec-Syn-4 on the LastFM dataset. LLMSRec-Syn-4 would leverage patterns in music preferences from the demonstration example to inform rankings in the test example. The user's previously listened music artists, like **'Foo Fighters'**, **'The Kooks'**, **'Tool'**, **'Editors'**, and **'Black Rebel Motorcycle Club'**, indicate a preference for alternative rock and post-punk bands. the ground truth label **Manson**, known for his unique style that blends elements of industrial and alternative rock, fits within this broader genre preference.

Aggregated Demonstration Example:
User's Previously Listened Music Artists:
- Recently Listened Music Artists (in sequential order): ['0. Cheap Mondays', '1. Skillet', ... , '48. The Strokes']
Candidate Music Artists for Recommendation:
- Candidate Music Artists: ['0. Die Fantastischen Vier', '1. Anugama', ... , '19. UGK']
Given the user's listening history, please arrange the candidate music artists in order of relevance to the user's preferences. It is important to:
- Rank ONLY the music artists listed in the candidates.
- Avoid introducing any music artists not included in the candidate list..
Recommendations should be formatted as follows: 1. [Most Recommended Artist] 2. [Second Most Recommended Artist] ...
Answer: 1. Die Fantastischen Vier 2. Staind 3. MGMT 4. Mark Ronson 5. Lasgo 6. Hey 7. Maxïmo Park 8. Defeater 9. Oddział Zamknięty 10. Alexander O'Neal 11. White Apple Tree 12. Anugama 13. Jane Birkin 14. Senser 15. Sam & Dave 16. Court Yard Hounds 17. Unkle Bob 18. Before Their Eyes 19. Zahara 20. UGK

Test User:
Learn from the above demonstration examples to solve the following test example.
User's Previously Listened Music Artists:
- Recently Listened Music Artists (in sequential order): ['0. **Foo Fighters**', '1. **The Kooks**', '2. A Perfect Circle', '3. **Tool**', '4. **Editors**', '5. The Magnetic Fields', '6. Them Crooked Vultures', '7. Snow Patrol', '8. Jimmy Eat World', '9. The Verve', '10. Vampire Weekend', '11. **Black Rebel Motorcycle Club**', ... , '49. The Waterboys']
Candidate Music Artists for Recommendation:
- Candidate Music Artists: ['0. Dulce María', '1. Cyber Whore meets Krachmaschine', '2. Billy Bragg and The Blokes', '3. CjClip', '4. Paul Oakenfold', '5. Hi:BRiD', '6. Fikret Kızılok', '7. ', '8. The Pioneers', '9. Apollo Brown', '10. Wheatus', '11. Aghora', '12. Flotsam and Jetsam', '13. Doubting Thomas', '14. NEVERSMILE', '15. ', '16. Wast3', '17. Zezé Di Camargo & Luciano', '18. Marilyn Manson', '19. Aggro Santos']
Given the user's listening history, please arrange the candidate music artists in order of relevance to the user's preferences. It is important to:
- Rank ONLY the music artists listed in the candidates.
- Avoid introducing any music artists not included in the candidate list..
Recommendations should be formatted as follows: 1. [Most Recommended Artist] 2. [Second Most Recommended Artist] ...
Answer: 1. **Manson** 2. Billy Bragg and The Blokes 3. Paul Oakenfold 4. Wheatus 5. Flotsam and Jetsam 6. Apollo Brown 7. Doubting Thomas 8. The Pioneers 9. Aghora 10. Cyber Whore meets Krachmaschine 11. Hi:BRiD 12. NEVERSMILE 13. 14. Wast3 15. CjClip 16. Dulce María 17. Fikret Kızılok 18. 19. Zezé Di Camargo & Luciano 20. Aggro Santos

---

Table 5: Example of LLMSRec-Syn-4 on the Games dataset. LLMSRec-Syn-4 would leverage patterns in game product preferences from the demonstration example to inform rankings in the test example. the user's purchase history, which includes multiple PlayStation-related items such as "**PS3 250 GB Uncharted 3 and PS Plus Bundle**" and "**Rocksmith 2014 Edition - Playstation 3 (Cable Included)**". This indicates a clear preference for PlayStation-related products, such as the ground truth game product **Playstation Plus: 3 Month Membership [Digital Code]**. The recommendation of a PlayStation Plus membership aligns with this preference.

Aggregated Demonstration Example:
User's Game Product Purchase History:
- Previously Purchased Game Products (in order of purchase): ['0. My Fitness Coach - Nintendo Wii', '1. Wii Messenger Bag', ... , '45. Rocksmith 2014 [Online Game Code]']
Potential Game Product Recommendations:
- List of Candidate Game Products for Consideration: ['0. Ryse: Son of Rome XBOX one', '1. Dark - Xbox 360', ... , '19. Battletoads']
Considering the user's past game product purchases, please prioritize the game products from the provided list that best match the user's preferences. Ensure that:
- You ONLY rank the given candidate game products.
- You DO NOT generate game products outside of the listed candidate game products..
Present your response in the format below: 1. [Top Recommendation] 2. [2nd Recommendation] ...
Answer: 1. Nintendo DS Lite Polar White 2. Wii G-pak Console Organizer Case ... 20. Ninja Reflex - Nintendo Wii

---

Test User:
Learn from the above demonstration examples to solve the following test example.
User's Game Product Purchase History:
- Previously Purchased Game Products (in order of purchase): ['0. Wii Fit Game with Balance Board', '1. EA Sports Active - Nintendo Wii', '2. **Rocksmith 2014 Edition - Playstation 3 (Cable Included)**', '3. DISNEY INFINITY Starter Pack PS3', '4. **PS3 250 GB Uncharted 3 and PS Plus Bundle**', '5. dreamgear DGPS3-1394 Radium Wireless Controller for PS3', '6. Personal Trainer', '7. Rocksmith 2014 [Online Game Code]', ...]
Potential Game Product Recommendations:
- List of Candidate Game Products for Consideration: ['0. Resident Evil 6 - Xbox One', "1. Tom Clancy's Ghost Recon Advanced Warfighter 2 - Xbox 360", '19. New Interchangeable Power Charging Adapter & Cable for Nintendo Wii U GamePad']
Considering the user's past game product purchases, please prioritize the game products from the provided list that best match the user's preferences. Ensure that:
- You ONLY rank the given candidate game products.
- You DO NOT generate game products outside of the listed candidate game products..
Present your response in the format below: 1. [Top Recommendation] 2. [2nd Recommendation] ...
Answer: 1. **Playstation Plus: 3 Month Membership [Digital Code]** 2. Call of Duty: Ghosts - PlayStation 4 3. eForCity Premium Vehicle Car Charger for Game Boy Advance SP/Nintendo DS 4. Angry Birds Star Wars - Nintendo 3DS 5. Monster High: Skultimate Roller Maze - Nintendo Wii 6. PDP Afterglow Karga Xbox One Headset - Green 7. New Interchangeable Power Charging Adapter & Cable for Nintendo Wii U GamePad 8. Call of Duty: World at War Platinum Hits - Xbox 360 9. Resident Evil 6 - Xbox One 10. ReCore - Xbox One 11. Tom Clancy's Ghost Recon Advanced Warfighter 2 - Xbox 360 12. Sniper: Ghost Warrior - Xbox 360 13. The Outfit - Xbox 360 14. Xbox 360 Street Fighter IV FightPad - Ken 15. Star Ocean: Second Evolution - Sony PSP 16. Teenage Mutant Ninja Turtles: Tournament Fighters 17. WinBack: Covert Operations - PlayStation 2 18. Stubbs The Zombie in Rebel Without a Pulse - Xbox 19. Theatrhythm Final Fantasy Curtain Call - Limited Edition 20. Collective Minds XBOX ONE Strike Pack

---