Sequentially Diversified and Accurate Recommendations in Chronological Order for a Series of Users

Jongjin Kim
Seoul National University
Seoul, Korea
j2kim99@snu.ac.kr

U Kang
Seoul National University
Seoul, Korea
ukang@snu.ac.kr

Abstract

When we sequentially recommend top-k items to users, how can we recommend them diversely while maintaining accuracy? Aggregate-level diversity is an important topic in recommender system since it is essential to maximize the potential profit of platforms by exposing a variety of items to users. However, previous studies do not consider the order of users receiving recommendations and assume that all users receive recommendations at once. In reality, users do not simultaneously receive recommendations so the preferences of the latter users are not given during recommending to the former users. In this work, we introduce the problem of sequentially diversified recommendation and propose SAPID, an accurate method to address the problem. SAPID removes the popularity bias from the model through a negative sampling mechanism based on temporal popularities. Then, SAPID collects candidate items to recommend based on the distribution of preference scores. Finally, SAPID decides which items to recommend immediately or later according to their estimated exposure opportunities. Extensive experiments show that SAPID shows the state-of-the-art performance in real-world datasets by achieving up to 61.0% increased diversity with 38.9% higher accuracy compared to the second-best competitor.

CCS Concepts

• **Information systems** → **Recommender systems**; **Information retrieval diversity**.

Keywords

Diversified Recommendation; Sequential Recommendation

ACM Reference Format:

Jongjin Kim and U Kang. 2025. Sequentially Diversified and Accurate Recommendations in Chronological Order for a Series of Users . In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining (WSDM '25), March 10–14, 2025, Hannover, Germany.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3701551.3703564

1 Introduction

When we sequentially recommend top-k items to users, how can we increase the diversity among the items while maintaining the accuracy? As the variety and abundance of products and information in market overwhelm the consumers, recommender systems have become an essential element for users to discover items to consume nowadays [13, 15, 16]. Recommender systems provide lists of items that the users are likely to be interested in [22, 23] and encourage the purchase from them. Hence, it is important to diversify the recommendations and expose as many products as possible to improve the total profit of platforms [2, 8]. However, this is challenging since the recommendation models are biased toward popular items during training because of the skewness in real-world data [26, 31]. Recently, studies about aggregately diversified recommender systems try to address this problem [1, 7, 14, 19, 20, 27]. They aim to increase aggregate-level diversity, which is the diversity among the recommendations for all users.

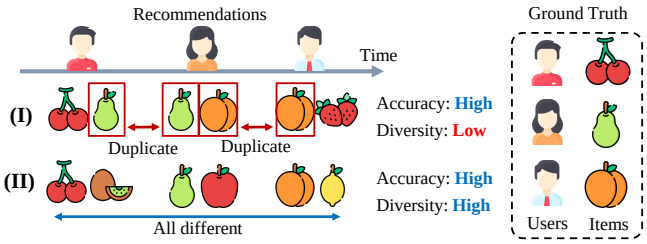


Figure 1: Examples of two sequential recommendation results (I) and (II). Both results (I) and (II) achieve the maximum accuracy. However, only result (II) achieves the maximum aggregate-level diversity since the result (I) recommends duplicate items to users.

However, existing works [7, 14, 19, 20, 27] do not consider the order of recommendations in the aggregately diversified recommendation field. In reality, the order of recommendations must be considered since users do not simultaneously receive recommendations. Thus, a recommender system cannot access the recommendation results for future users in advance to recommend to the current user. For instance, an e-commerce site shows recommended products to a customer before knowing which items will be browsed by tomorrow's users. This assumption fits well for the sequential recommendation setting, where time is the main consideration [32] so that the recommender system is strictly prohibited from using future information to recommend [18].

Hence, we introduce the problem of sequentially diversified recommendation to consider the order of recommendations in the aggregately diversified recommender system. Figure 1 shows the example of the sequentially diversified recommendation. Three users sequentially receive recommendations (I) and (II). Both results achieve the highest accuracy by recommending to each user their preferred items. However, result (I) exposes only four kinds of items and achieves poor diversity. On the other hand, result (II) achieves the highest diversity by recommending entirely different items for

WSDM '25, March 10–14, 2025, Hannover, Germany.

ACM ISBN 979-8-4007-1329-3/25/03
https://doi.org/10.1145/3701551.3703564

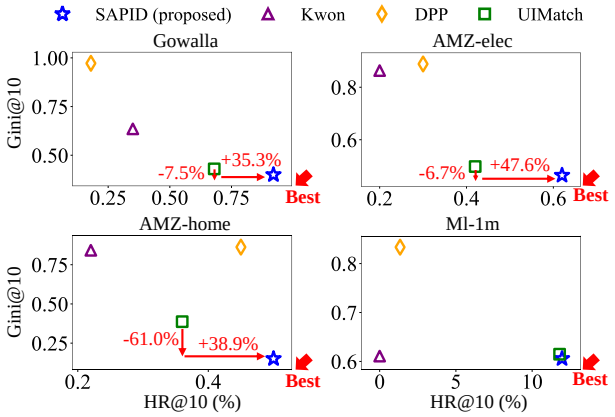


Figure 2: Performance of SAPID and competitors in real-world datasets. SAPID achieves the best accuracy and diversity, being the closest to the best point (red arrow).

all users. This is because the result (I) recommends items that latter users would prefer, leading to duplicate recommendations for the next user. In contrast, result (II) effectively distributes items by not recommending those intended for later use. The sequentially diversified recommendation aims to maximize both accuracy and diversity as in the result (II). As shown in Figure 1, it is important to distinguish between the items to be recommended now and those to be recommended later to achieve this goal.

In this paper, we propose SAPID (Sequentially Diversified Recommendation via Popularity Debiasing and Item Distribution), an accurate method for the sequentially diversified recommendation. SAPID removes the popularity bias from the base model by temporal popularity-based negative sampling scheme. Then, SAPID gathers the candidate pool of items to be recommended based on the distribution of preference scores.SAPID constructs the recommendation list from the candidate pool by considering whether each item would have a chance to be recommended later or not. Extensive experiments in real-world datasets show that SAPID achieves up to 61.0% increased diversity with 38.9% higher accuracy compared to the second best competitor as shown in Figure 2.

Our contributions are summarized as follows:

- **Problem Formulation.** We formally define the problem of sequentially diversified recommendation, an important task to maximize the potential profits of e-commerce platforms. We also suggest an evaluation metric that suits the problem.
- **Method.** We propose SAPID, an accurate method for the sequentially diversified recommendation. SAPID mitigates the popularity bias from the base model by popularity-based negative sampling in the training phase, and maximizes the diversity of recommendations by effectively distributing items in the reranking phase.
- **Experiments.** Extensive experiments in real-world datasets show that SAPID achieves the state-of-the-art performance compared to other methods for the aggregately diversified recommendation.

In the rest of our paper, we provide the related works in Section 2, formally define the problem of sequentially diversified recommendation in Section 3, introduce our proposed method in Section 4, evaluate the proposed method and competitors on real-world datasets in Section 5, and conclude this work in Section 6. The code and datasets are available at https://github.com/snudatalab/SAPID.

2 Related Works

2.1 Aggregately Diversified Recommendation

Aggregately diversified recommendation aims to increase the diversity in the overall recommendation results of all users [1]. In recent years, this problem has attracted huge attention since it is essential to improve the potential profit of commerce platforms [2, 8] and user satisfaction [24]. Most previous studies try to modify the results of the base model to effectively reallocate items among recommendation lists and increase total diversity. Kwon et al. [1], Karakaya et al. [19], FairMatch [27], and UImatch [7] rerank the base model's recommendation results to achieve high diversity. DivMF [20] and PopCon [14] further regularize the base model to remove its popularity bias. However, they assume that recommendation lists are generated simultaneously, which is not the case for the sequentially diversified recommendation.

2.2 Individually Diversified Recommendation

Individually diversified recommendation aims to recommend more diverse items for each user [35]. Maximizing individual-level diversity enhances the user experience by showing novel items to users. Previous works of individually diversified recommendation utilize various methods such as GNN [36, 37] or DPP [34] to achieve high diversity. Recently, ComiRec [3] and IDSR [5] address the individual-level diversity in the sequential recommendation, trying to recommend novel items beyond the given session. However, an individually diversified recommender system may recommend already recommended items to new users [1], and thus individually diversified recommendation differs from sequentially diversified recommendation which cares about diversity over all users.

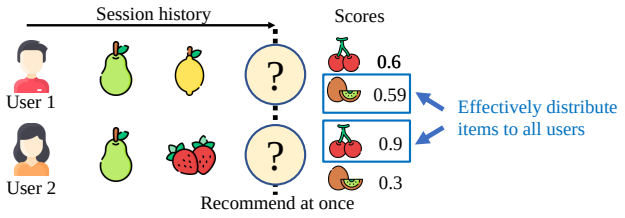
2.3 Long-tail Recommendation

Recommender systems tend to perform poorly for unpopular items due to the lack of information about long-tail items in skewed real-world datasets [17, 30]. Long-tail recommendation aims to obtain better representations of long-tail items from the lacking information to provide higher-quality recommendations with those items [12, 21, 25, 33]. However, the main focus of the long-tail recommendation differs from the aggregately diversified recommendation or sequentially diversified recommendation since they care about balancing the quantity of each item's recommendations rather than improving the quality.

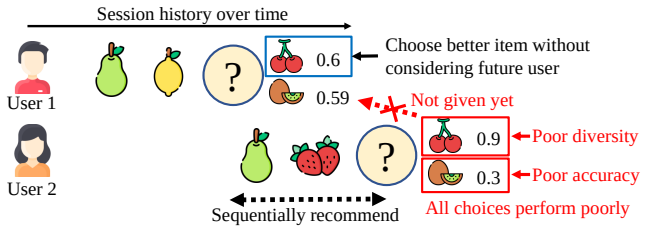
3 Problem Formulation

3.1 Sequentially Diversified Recommendation

***Problem* 1** (Sequentially Diversified Recommendation)**:** Assume that a session is a sequence of user-item interactions for a single user with timestamps. For the set $\mathbb{U}$ of users, the sessions $\mathbb{S}_1, \cdots, \mathbb{S}_{|\mathbb{U}|}$ are given and sorted in their last interactions' time order. Each session is provided sequentially to perform a recommendation, one at a time. The problem is to recommend a list $\mathbb{R}_u$ of k items for each session $\mathbb{S}_u$ that are most likely to appear next in the session while maximizing both the accuracy and the aggregate-level diversity. □



(a) Aggregately diversified recommendation



(b) Sequentially diversified recommendation

Figure 3: Illustrative comparison between the aggregately diversified recommendation and the sequentially diversified recommendation. See Section 3.1 for details.

Note that the recommender system needs to generate $\mathbb{R}_u$ as soon as $\mathbb{S}_u$ is provided. Thus, the information of $\mathbb{S}_{u+1}, \cdots, \mathbb{S}_n$ and $\mathbb{R}_{u+1}, \cdots, \mathbb{R}_n$ are not given while recommending for $\mathbb{S}_u$. We consider timestamps of user-item interactions as well as the timely order between users in formulating the problem to have consistency in considering the time. However, we do not consider further content information such as categories or genres in this paper.

Figure 3 illustrates the difference between the sequentially diversified recommendation and the conventional aggregately diversified recommendation. We are trying to recommend a cherry or a kiwi for each user. In aggregately diversified recommendation problem, we recommend to both users simultaneously as shown in Figure 3(a). Hence, the ideal strategy is to recommend a kiwi to user 1 and a cherry to user 2 since user 1 prefers both fruits while user 2 prefers only a cherry. In contrast, in the sequentially diversified recommendation problem, the recommender system does not have information about user 2 when recommending to user 1 as shown in Figure 3(b). In this case, if we naively recommend a cherry to user 1 based on user 1's preferences, we face a dilemma when recommending to user 2: to achieve diversity, the model would have to recommend a kiwi which user 2 does not like, or to maintain accuracy, the model would have to recommend a cherry which leads to poor diversity. Thus, it is not easy to simply use effective item allocation techniques from conventional aggregately diversified recommendation methods for sequentially diversified recommendation. Instead, it is crucial to consider the future demand for items and make decisions on which items to recommend immediately and which ones to reserve for the latter use.

3.2 Evaluation

There are two key criteria to evaluate the performance of sequentially diversified recommendation: accuracy and diversity.

3.2.1 Accuracy. The accuracy of a recommendation system refers to how well the model's recommendations align with the items that the user prefers. However, traditional accuracy metrics such as HR or nDCG do not reflect the consistency of a recommendation result. For instance, consider a recommender system that follows a strategy of recommending 1) the highest-scored items for the first 80% of users, and 2) previously unrecommended items for the last 20% to increase diversity. This recommender system would achieve at least 80% of the accuracy of the base model since the recommendation results for the first 80% of users would be the same. On the other hand, the latter 20% of users receive items that are entirely unrelated to their preferences, which is not acceptable for sales platform due to poor user experiences. Hence, we need new metrics to evaluate the consistency of the recommendation results as well as the overall accuracy to prevent a model from unfairly achieving high performance by ignoring latter users.

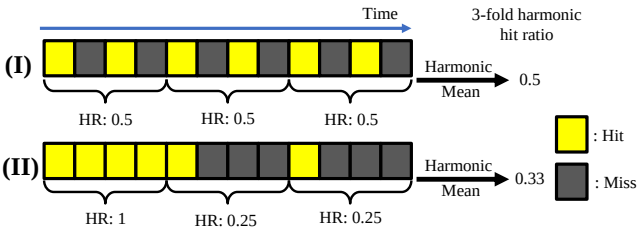


Figure 4: Examples of evaluating 3-fold harmonic hit ratio for two models (I) and (II). Yellow boxes indicate the correct recommendations (hit), and grey boxes indicate the wrong recommendations (miss) for each instance.

We propose the k-fold harmonic metric to jointly evaluate the accuracy and the consistency of recommendation results.

Definition 3.1. Let $\mathbf{A}(f, \mathbb{S})$ be an accuracy metric evaluating the recommender system f for sessions $\mathbb{S}$ of length n. Then, the k-fold harmonic version $\mathbf{A}_H$ of $\mathbf{A}$ is as follows:

$$\mathbf{A}_H(f, \mathbb{S}) = H(\mathbf{A}(f, [\mathbb{S}_1 : \mathbb{S}_{\frac{n}{k}}]), \cdots, \mathbf{A}(f, [\mathbb{S}_{\frac{(k-1)n}{k}+1} : \mathbb{S}_n])), \quad (1)$$

where $H(\cdot)$ is the harmonic mean function. □

Figure 4 illustrates the evaluation process of 3-fold harmonic hit ratios of two models (I) and (II). To calculate 3-fold harmonic hit ratio, we first divide session instances into three portions and calculate the hit ratio in each portion. Then, we calculate the harmonic means of hit ratios.

Note that the maximum harmonic hit ratio or nDCG is equal to the average hit ratio or nDCG because of the arithmetic-harmonic mean inequality. Since the equality condition holds when all values are equal, these new metrics give the highest score to the model with the most uniform accuracy among models with the same average accuracy. For instance, even if two models (I) and (II) shown in Figure 4 have the same average hit ratios, model (I) achieves a higher 3-fold harmonic hit ratio than model (II). This is because model (I) consistently performs well across all periods while model (II) performs poorly in the latter sessions.

3.2.2 Diversity. The diversity of the sequentially diversified recommendation represents how diverse the item distribution is in the overall recommendation results. It is calculated by entropy or Gini index both of which measure the evenness in distributions. Entropy gives a positive real value, and a larger entropy indicates better diversity. Gini index ranges from 0 to 1, and a lower Gini index indicates better diversity.

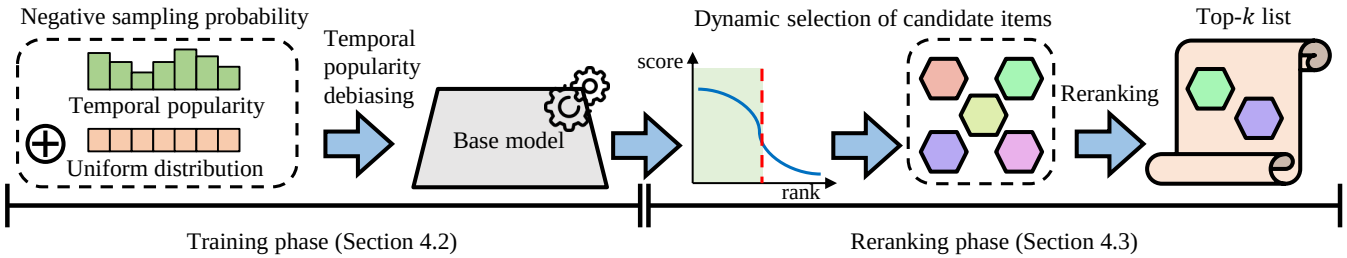


Figure 5: The overall architecture of SAPID.

4 Proposed Method

In this section, we propose SAPID (Sequentially Diversified Recommendation via Popularity Debiasing and Item Distribution), an effective approach for the sequentially diversified recommendation.

4.1 Overview

We address the following challenges to achieve high performance in the sequentially diversified recommendation:

C1. **Skewed real-world data.** Long tail items are underrated during the training process because of the skewed real-world data. How can we prevent the model from being biased toward the skewness of the real-world data?
C2. **Diversity maximization.** How can we increase the diversity of recommendations while maintaining the accuracy?
C3. **Accuracy consistency.** If we repeatedly recommend a few items to the earlier users, we have to recommend the remaining items to the latter users to improve diversity. However, those items may not be preferred by the latter users. How can we decide which items to recommend now or later?

The main ideas of SAPID are summarized as follows:

I1. **Temporal popularity debiasing.** SAPID considers the item frequencies during a given session when sampling negative items to prevent the popularity bias in the model.
I2. **Candidate selection.** SAPID constructs the candidate pool for each user considering their preference score predicted by the base model. Then, SAPID chooses diverse items from the candidate pool to increase diversity while preserving accuracy.
I3. **Effective item distribution.** SAPID judges whether the item would have opportunities to be recommended later or not to decide which items to recommend immediately and which items to preserve.

Figure 5 shows the overall process of SAPID. SAPID is composed of two phases: training phase and reranking phase. In the training phase, SAPID trains a sequential recommendation model such as SASRec [18] or GRU4Rec [11] as a base model with a temporal popularity-based negative sampling scheme to prevent the model from being biased toward popular items. In the reranking phase, SAPID checks how many times each item is recommended and will be recommended. Then, SAPID generates the recommendation lists by selecting the best items to increase the diversity from the candidate set generated by the base model.

4.2 Training Phase

How can we eliminate the popularity bias introduced by the skewed distribution during the model training process? The BPR loss in recommendation model training aims to maximize the difference between the recommendation scores of positive samples, which are items a user has interacted with, and negative samples, which are randomly selected items the user has not interacted with. The assumption behind this loss function is that a user would prefer items they have interacted with over those that they have not interacted with [29]. However, this approach leads to the popularity bias in the model, as lesser-known items are underestimated. This is because a user may be unaware of the lesser-known items even if those items meet the user's taste. Hence, lesser-known items are overly chosen as negative samples if the sampling probability is uniform. This leads to the poor diversity of the model's recommendation results.

For instance, if a user session was established before 2020 which is before the release of "Queen's Gambit", the absence of interactions with the series is because of the unavailability of the content at that time rather than the user's disinterest. In this case, choosing "Queen's Gambit' as a negative sample unfairly decreases the recommendation score of the series. Conversely, if a user session was formed after the release and the user did not engage with "Queen's Gambit," it indicates a lack of interest in the series. It is appropriate to choose "Queen's Gambit' as a negative sample for this case.

Our approach to address this issue is to consider temporal item popularity when selecting negative samples. SAPID finds which items were popular and which were not when the user session was formed. Then, SAPID reduces the frequency of lesser-known items being chosen as negative samples. Thus, unpopular items are not unfairly chosen as negative samples, mitigating the popularity bias.

To implement this idea, SAPID calculates the temporal popularity $\mathbf{P}^{\mathbb{S}}_{pop}$ of negative items for a session $\mathbb{S}$ by counting all interactions of users and items not included in $\mathbb{S}$ that happened between the first interaction of the session and the last interaction of the session. Let $\mathbf{P}^{\mathbb{S}}_{uni}$ be the uniform distribution for all items not included in $\mathbb{S}$. Then, the probability distribution $\mathbf{P}^{\mathbb{S}}$ for each item to be selected as a negative sample for the session $\mathbb{S}$ is defined as follows:

$$\mathbf{P}^{\mathbb{S}} = \alpha \mathbf{P}^{\mathbb{S}}_{pop} + (1-\alpha)\mathbf{P}^{\mathbb{S}}_{uni}, \tag{2}$$

where α is a hyperparameter which ranges from 0 to 1.

4.3 Reranking Phase

How can we maintain accuracy while increasing diversity? When striving to enhance diversity by recommending items that have not been recommended before, there is a trade-off as it may lead to recommending items that the user does not prefer, thereby potentially reducing the accuracy of recommendations. SAPID adopts the reranking method to address this problem. The reranking method is a technique to choose items to recommend from the candidate

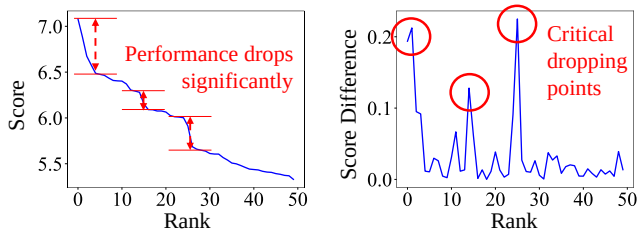


Figure 6: Preference scores of the first user on Ml-1m dataset for highly ranked 50 items predicted by SASRec. Preference scores sharply drop at a few points. Hence, it is ideal to compose the candidate set based on these points.

pool generated by a base model [1, 7, 14, 27]. To achieve high performance in this process, we must consider 1) how to compose a candidate set of recommendable items and 2) which items to recommend from the candidate set to maximize the diversity.

4.3.1 Constructing Candidate Pool. How can we accurately gather recommendable items for each user? Previous studies [14, 27] simply collect highly ranked c items through the base model where c is a hyperparameter greater than the length k of a recommendation list. However, this approach leads to a sharp decline in item preference within the candidate set. For example, if we collect top-2 items as the candidate set for each user in Figure 3, user 1's candidate set has two recommendable items that have comparable preference scores. On the other hand, a kiwi would be included in the candidate set of user 2, which is significantly less preferred compared to a cherry and thus not recommendable. Hence, fixing the size of the candidate pool allows lesser preferred items to get mixed into the pool and harms the quality of a recommendation.

In Figure 6, we visualize the preference scores and differences with consecutively ranked items for the top 50 items predicted by SASRec for the first user in the Ml-1m dataset to analyze how the item qualities of the candidate set drop in the real-world dataset. Note that there are distinct drop-off points where item preference sharply declines, indicating a significant quality gap between the items before and after the points. Hence, it is ideal to compose the candidate set based on these drop-off points.

SAPID refers to the difference of preference scores to compose the candidate set based on drop-off points. First, SAPID examines the score differences between consecutively ranked items for the top c items evaluated by the base model. Then, SAPID identifies the drop-off points where the score differences fall within the top 25%. Finally, SAPID collects items above the point closest to the c-th ranked item as a candidate pool. This ensures that all candidate items are preferred by the user.

4.3.2 Selecting Items to Recommend. How can we choose items from the candidate pool to maximize the diversity? SAPID judges how much an item contributes to diversity by counting its expected number of being recommended. To achieve better diversity, the recommender system needs to recommend less frequently recommended items. For instance, assume that there are a popular item A and a lesser-known item B in the candidate pool. Then, A is more likely to be recommended later for other users since it is a more popular one. Thus, A has a higher expected number of being recommended so it is better to recommend B for better diversity. On the other hand, assume that there are items C and D in the candidate

Algorithm 1 Reranking phase of SAPID

Input: The base model's recommendation lists $\mathbb{R}^c_1, \cdots, \mathbb{R}^c_{|\mathbb{U}|}$ of length c each, vectors denote their preference scores $\mathbf{G}^c_1, \cdots, \mathbf{G}^c_{|\mathbb{U}|}$, popularity distribution $\mathbf{P}_{pop} = [p_1, \cdots, p_{|\mathbb{I}|}]$
Output: Recommendation lists $\mathbb{R}_1, \cdots, \mathbb{R}_{|\mathbb{U}|}$ of length k each

1: Initialize a vector $\mathbf{C}$ of length $|\mathbb{I}|$ to $[0, \cdots, 0]$
2: **for** u in $[1, \cdots, |\mathbb{U}|]$ **do**
3: /* Constructing Candidate Pool */
4: Initialize a vector $\mathbf{D}$ of length $c-1$
5: $\mathbf{D} \leftarrow \mathbf{G}^c_u[:-1] - \mathbf{G}^c_u[1:]$
6: Let $\mathbf{d}$ be the indices of the top 25% values in $\mathbf{D}$
7: Compose the candidate pool $\mathbb{R}^{cand}_u$ as $\mathbb{R}^c_u[:\mathbf{d}[-1]]$
8: /* Selecting Items to Recommend */
9: Initialize a vector $\mathbf{E}$ of length c
10: **for** i_r, $item$ in $enumerate(\mathbb{R}^{cand}_u)$ **do**
11: $\mathbf{E}[i_r] \leftarrow \mathbf{C}[item] + p_{item} \cdot (|\mathbb{U}| - u) \cdot k$
12: **end for**
13: Let a set $\mathbb{K}$ be the indices of the smallest k values in $\mathbf{E}$
14: $\mathbb{R}_u \leftarrow []$
15: **for** i_t in $\mathbb{K}$ **do**
16: $\mathbb{R}_u.append(\mathbb{R}^{cand}_u[i_t])$
17: $\mathbf{C}[\mathbb{R}^{cand}_u[i_t]] \leftarrow \mathbf{C}[\mathbb{R}^{cand}_u[i_t]] + 1$
18: **end for**
19: **end for**
20: **return** $\mathbb{R}_1, \cdots, \mathbb{R}_{|\mathbb{U}|}$

pool where currently C is more recommended for previous users. In this case, C has a higher expected number of being recommended since it has already been recommended a lot. Hence, it is better to recommend D for better diversity.

SAPID calculates the expected number of an item being recommended by summing 1) the number of times the item has been recommended for the previous sessions and 2) the estimated number of times it will be recommended for the remaining sessions. We assume that the recommendation rate of each item for future users is proportional to its popularity. Thus, the total expected number $E_u(i)$ of recommendations for item i when recommending to the u-th user is as follows:

$$E_u(i) = C(i) + (|\mathbb{U}| - u)k \times p_i, \tag{3}$$

where $C(i)$ is the current count of recommendations for item i, $\mathbb{U}$ is the set of all users, k is the length of a recommendation list, and p_i is the ratio of item i's occurrence count to the total number of interactions in the training data. Note that $(|\mathbb{U}| - u)k$ indicates the remaining recommendation slots for future users.

4.3.3 Algorithm. Algorithm 1 shows the reranking phase of SAPID. SAPID first initializes the current count $\mathbf{C}$ of each item to zero in line 1. Then, SAPID processes each user through a for loop. From lines 3 to 7, SAPID collects the candidate pool based on the score differences $\mathbf{D}$. From lines 9 to 12, SAPID calculates the expected total number $\mathbf{E}$ of recommendations for each item at the time. In line 13, SAPID finds the set $\mathbb{K}$ of indices of the least recommended k items from the candidate pool. From lines 14 to 18, SAPID generates the recommendation list for the processing user and updates $\mathbf{C}$. SAPID returns recommendation lists in line 20.

Table 1: Summary of datasets.

Dataset	Users	Items	Interactions	Avg. Length
Gowalla [1]	34,688	63,279	2,438,708	70.30
AMZ-elec [2]	33,602	16,448	788,143	23.46
AMZ-home [2]	12,000	8,445	291,560	24.30
Ml-1m [3]	6,040	3,706	1,000,209	165.60

[1] https://snap.stanford.edu/data/loc-gowalla.html
[2] https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/
[3] https://grouplens.org/datasets/movielens/1m/

4.4 Complexity Analysis

We analyze the time complexity of SAPID once the recommendation lists of the base model are given.

Theorem 4.1. *Time complexity of Algorithm 1 is* $O(|\mathbb{U}|(c+k\log c))$, *where* $\mathbb{U}$ *is the set of all users,* c *is the size of a candidate pool, and* k *is the length of a recommendation list.* □

Proof. Note that Algorithm 1 repeats the same process for each user to generate recommendation lists. Hence, we need to consider only the time complexity to generate a recommendation list for a single user (from lines 3 to 18). The time complexity to collect the candidate pool from lines 3 to 7 and calculate $\mathbf{E}$ from lines 10 to 12 is $O(c)$ since they are simple vector operations. In line 13, finding the smallest k values in $\mathbf{E}$ requires $O(c + k\log c)$ time with min-heap structure. The time complexity to construct the recommendation list and update $\mathbf{C}$ is $O(k)$. Thus, the time complexity to process a user in Algorithm 1 is $O(c + k\log c)$ and the time complexity to process $|\mathbb{U}|$ users is $O(|\mathbb{U}|(c + k\log c))$. □

5 Experiment

We perform experiments to address the following questions:

Q1. **Performance (Section 5.2).** Does SAPID achieve higher diversity while sacrificing lesser accuracy compared to competitors?
Q2. **Hyperparameter sensitivity (Section 5.3).** How does the performance of SAPID change with variations in hyperparameters?
Q3. **Case study (Section 5.4).** Why is considering the latter users important to achieve better performance in the sequentially diversified recommendation?

5.1 Experimental Setup

Datasets. We use four real-world datasets for sequential recommendation as summarized in Table 1. Gowalla [6] contains users' checking-in data on a location-based social networking platform, Gowalla. AMZ-elec and AMZ-home [10, 28] are product review datasets collected from the e-commerce platform, Amazon. AMZ-elec consists of reviews for electronic products, while AMZ-home consists of reviews for home and kitchen products. Ml-1m [9] is a movie rating dataset collected by the GroupLens research group. Each dataset consists of user-item interaction history along with timestamps. We remove users and items with fewer than 15 interactions since we focus on recommending items to users with session history, rather than addressing the cold start problem.

Base models. We use the following two widely used sequential recommendation models as our base models for experiments.

- **SASRec [18].** SASRec is a Transformer-based sequential recommendation model. It encodes a user session with the directional Transformer encoder to extract the session owner's preference and recommend the next item.
- **GRU4Rec [11].** GRU4Rec is an RNN-based sequential recommendation model. It utilizes GRU cells to process each interaction of the given session and extracts the session's hidden states to predict the next item of the session.

Baselines. We use aggregately diversified recommendation methods as baselines to compare with SAPID. Note that most aggregately diversified recommendation methods do not apply to the sequentially diversified recommendation since they require recommendation results for all users at once. For instance, FairMatch [27] requires the recommendation lists for all users to construct a user-item graph. Karakaya et al. [19] need to count the frequencies of each item within the entire recommendation results. PopCon [14] simultaneously generates recommendation lists for all users at once. However, recommendation results of the latter users are not given when recommending items to the former users in the problem of sequentially diversified recommendation. Thus, the baselines have to generate a recommendation list for each user based only on information about the user without knowing the recommendation results of the base model for other users. Hence, we use the following three diversified recommendation methods as our baselines.

- **Kwon et al [1].** Kwon et al. modify the recommendation scores given by a base model to trade-off accuracy and diversity of recommendation results.
- **DPP [4].** DPP utilizes a determinantal point process to find diverse items which are lesser recommended. DPP also increases aggregate-level diversity despite focusing mainly on individual-level diversity.
- **UImatch [7].** UImatch assigns the maximum recommendation frequencies for each item and greedily selects items to recommend based on their recommendation scores.

Evaluation protocols. We employ a leave-one-out protocol and remove the last interaction of each user to construct the training data. We use the removed interaction as the ground truth of the test data and the rest as the test input for each user's session history during the evaluation. We assume that the user receives a recommendation when they interact with an item. Hence, we provide each session in the time order of the user's last interaction to evaluate the performance. We use 5-fold harmonic hit ratio and 5-fold harmonic nDCG to measure the accuracy. We use entropy and the Gini index to measure the diversity.

Training details. We set the dimensionality of embeddings to 50 for all models. We use two self-attention blocks and a drop-out ratio of 0.5 for SASRec. We use one GRU layer for GRU4Rec. All the models are trained for 200 epochs with Adam optimizer with learning rate 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.98$. We use a reverse prediction scheme and $T_H \in [0.5, 1)$ for Kwon et al. We use the implementation on https://github.com/laming-chen/fast-map-dpp for DPP.

5.2 Performance (Q1)

We measure the accuracy and the diversity of SAPID and baselines. We train each base model twice: without and with temporal

Table 2: Performance of SAPID and competitors compared to the base model. Bold texts and underlined texts indicate the best and the second-best performances among competitors, respectively. SAPID outperforms all competitors in most cases.

Base model : SASRec		Without Debiasing				With Debiasing			
Dataset	**Method**	**HR@10 (↑)**	**nDCG@10 (↑)**	**Entropy@10 (↑)**	**Gini@10 (↓)**	**HR@10 (↑)**	**nDCG@10 (↑)**	**Entropy@10 (↑)**	**Gini@10 (↓)**
Gowalla	SASRec (Base)	0.0288	0.0150	8.4082	0.9351	0.0208	0.0101	8.9862	0.8982
	Kwon et al.	0.0044	0.0021	10.1034	0.6939	0.0035	0.0015	10.3057	0.6354
	DPP	**0.0129**	**0.0084**	6.8514	0.9767	0.0018	0.0014	7.1015	0.9721
	UImatch	0.0050	0.0023	10.5467	0.4487	0.0068	0.0040	10.5737	0.4300
	SAPID (Proposed)	0.0097	0.0050	**10.6361**	**0.4193**	**0.0092**	**0.0049**	**10.6517**	**0.3999**
AMZ-elec	SASRec (Base)	0.0268	0.0137	6.0685	0.9762	0.0179	0.0098	7.3271	0.9145
	Kwon et al.	0.0018	0.0008	6.7966	0.9532	0.0020	0.0009	7.8894	0.8635
	DPP	0.0018	0.0011	6.7193	0.9459	0.0030	0.0016	7.6090	0.8887
	UImatch	0.0017	0.0008	9.1471	0.5047	0.0042	0.0030	9.1598	0.4978
	SAPID (Proposed)	**0.0042**	**0.0023**	**9.2526**	**0.4758**	**0.0062**	**0.0035**	**9.2896**	**0.4642**
AMZ-home	SASRec (Base)	0.0511	0.0350	6.0015	0.9487	0.0371	0.0266	6.9828	0.8862
	Kwon et al.	0.0025	0.0010	6.5884	0.9130	0.0022	0.0009	7.3245	0.8419
	DPP	0.0040	0.0023	6.7288	0.9003	0.0045	0.0025	7.1502	0.8623
	UImatch	0.0000	0.0000	8.7149	0.3933	0.0036	0.0030	8.7261	0.3869
	SAPID (Proposed)	**0.0067**	**0.0046**	**8.8911**	**0.2555**	**0.0050**	**0.0037**	**8.9682**	**0.1508**
Ml-1m	SASRec (Base)	0.1805	0.0914	6.8847	0.7958	0.1577	0.0790	7.2376	0.7124
	Kwon et al.	0.0000	0.0000	7.2282	0.7177	0.0000	0.0000	7.5405	0.6109
	DPP	0.0128	0.0070	6.5724	0.8364	0.0134	0.0082	6.6078	0.8335
	UImatch	0.0774	0.0374	7.4725	0.6406	0.1180	**0.0628**	7.5428	0.6148
	SAPID (Proposed)	**0.0820**	**0.0423**	**7.7248**	**0.5123**	**0.1196**	0.0620	**7.5494**	**0.6057**

Base model : GRU4Rec		Without Debiasing				With Debiasing			
Dataset	**Method**	**HR@10 (↑)**	**nDCG@10 (↑)**	**Entropy@10 (↑)**	**Gini@10 (↓)**	**HR@10 (↑)**	**nDCG@10 (↑)**	**Entropy@10 (↑)**	**Gini@10 (↓)**
Gowalla	GRU4Rec (Base)	0.0182	0.0092	9.4498	0.8374	0.0120	0.0054	9.4255	0.8468
	Kwon et al.	0.0049	0.0022	10.2539	0.6477	0.0053	0.0023	10.2281	0.6638
	DPP	0.0052	0.0036	8.7888	0.9010	0.0014	0.0011	7.9300	0.9496
	UImatch	0.0066	0.0035	10.4368	0.5057	0.0090	0.0049	10.4624	0.5039
	SAPID (Proposed)	**0.0083**	**0.0044**	**10.6866**	**0.3780**	**0.0105**	**0.0056**	**10.5885**	**0.4459**
AMZ-elec	GRU4Rec (Base)	0.0094	0.0047	7.5910	0.8984	0.0058	0.0027	8.0572	0.8401
	Kwon et al.	0.0030	0.0013	8.8817	0.6695	0.0020	0.0009	9.2119	0.5380
	DPP	0.0024	0.0014	7.2973	0.9239	0.0041	0.0017	7.1595	0.9284
	UImatch	0.0021	0.0011	8.8608	0.6372	0.0036	0.0023	9.1897	0.5064
	SAPID (Proposed)	**0.0049**	**0.0028**	**9.1767**	**0.5222**	**0.0059**	**0.0034**	**9.3573**	**0.4168**
AMZ-home	GRU4Rec (Base)	0.0165	0.0081	7.4512	0.8321	0.0119	0.0060	7.9003	0.7492
	Kwon et al.	0.0050	0.0024	8.3049	0.6368	0.0044	0.0022	8.6279	0.4933
	DPP	0.0045	0.0023	7.0759	0.8735	0.0047	0.0022	7.4284	0.8279
	UImatch	0.0051	0.0026	8.6107	0.4644	0.0080	0.0045	8.7071	0.4070
	SAPID (Proposed)	**0.0067**	**0.0046**	**8.8911**	**0.2555**	**0.0083**	**0.0060**	**8.8827**	**0.2572**
Ml-1m	GRU4Rec (Base)	0.1716	0.0860	6.9456	0.7837	0.1508	0.0736	7.3581	0.6778
	Kwon et al.	0.0000	0.0000	7.4264	0.6536	0.0000	0.0000	**7.6081**	**0.5832**
	DPP	0.0153	0.0094	6.7176	0.8184	0.0202	0.0111	6.5236	0.8455
	UImatch	0.0773	0.0393	7.4676	0.6428	0.1075	0.0586	7.5251	0.6209
	SAPID (Proposed)	**0.0820**	**0.0423**	**7.7248**	**0.5123**	**0.1133**	**0.0587**	7.5776	0.5929

Table 3: Value of hyperparameter c for each experiment.

Model	Gowalla	AMZ-elec	AMZ-home	Ml-1m
SASRec	200	600	500	50
Debiased SASRec	200	200	500	24
GRU4Rec	250	500	500	50
Debiased GRU4Rec	150	250	275	26

popularity-based debiasing. SAPID without debiasing shows the performance when only the reranking phase of SAPID is applied. SAPID with debiasing shows the performance when all of our main ideas are applied. We also apply debiasing to competitors to confirm that SAPID performs better even if they also utilize the debiasing mechanism of SAPID. We set the debiasing hyperparameter α to 0.5. The hyperparameter c for each experiment is given in Table 3.

Table 2 shows the results. SAPID shows the best accuracy and diversity among competitors in most caseswhich proves the effectiveness of the reranking phase of SAPID. This is because SAPID finds items that improve the diversity considering their future demands. Figure 2 summarizes the results with SASRec as the base model. Note that SAPID shows the best accuracy and diversity.

5.3 Hyperparameter Sensitivity (Q2)

Debiasing hyperparameter α. To examine the effect of the debiasing hyperparameter α,we evaluate the performance of SAPID with SASRec as the base model in Ml-1m dataset while changing α. Note that popular items when the session is formed are more likely to be selected as negative samples as α increases.

Table 4 shows the results. Note that higher values of HR, nDCG, and Entropy are better, while a lower value of Gini is preferable. As α increases, accuracy gradually decreases while diversity increases. This trend proves the effectiveness of the temporal popularity debiasing method, as it encourages the model to recommend a wider range of items beyond the popular ones by assigning a higher probability of being negative samples for popular items.

Table 4: Analysis on debiasing hyperparameter α in Ml-1m dataset with SASRec as the base model. Note that accuracy decreases and diversity increases as α increases.

α	Accuracy HR@10(↑)	Accuracy nDCG@10(↑)	Diversity Ent.@10(↑)	Diversity Gini@10(↓)
0.05	0.2020	0.1052	6.9500	0.7842
0.25	0.1712	0.0889	7.0411	0.7648
0.5	0.1577	0.0790	7.2376	0.7124
0.75	0.1458	0.0729	7.2714	0.7015
0.95	0.1409	0.0693	7.4745	0.6316

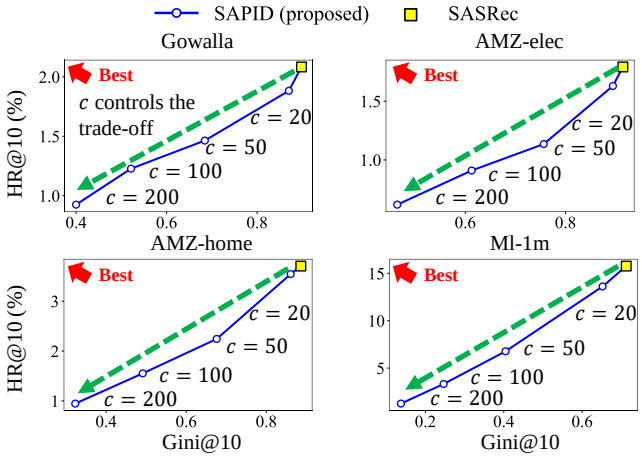


Figure 7: Change in the performance of SAPID while the size c of the candidate pool varies. SAPID further improves diversity by sacrificing accuracy as c increases.

Meanwhile, examining the performance of the original SASRec which is equivalent to $\alpha = 0$ as reported in Table 2, the accuracy is higher when $\alpha = 0.05$. This proves that the traditional negative sampling strategy not only hinders diversity but also damages the accuracy by making the model biased and over-recommends popular items. Hence, mitigating bias is important for not only the diversified recommendation but also accurate recommendation.

Initial candidate pool size c. We change the number c of the candidate pool and observe the change in the performance of SAPID to analyze its effect. Figure 7 shows the performance change of SAPID using SASRec as the base model on four real-world datasets. In all datasets, SAPID shows a clear pattern that it improves diversity by sacrificing accuracy as c increases. Hence, it is important to choose a proper c to control the trade-off and achieve desired performance.

5.4 Case Study (Q3)

To investigate how the consideration of future users affects the performance,we analyze the recommendation qualities of SAPID and UImatch on Gowalla dataset. We use a debiased SASRec as the base model. Note that UImatch does not consider future users and greedily constructs recommendation lists. Meanwhile, SAPID considers future users and preserves popular items for later use.

Figure 8 shows the average rank of items received by each user in their chronological order. The rank of items is determined by the base model. A higher average rank indicates that the user received items they are more likely to prefer, while a lower average rank denotes that the items recommended are irrelevant to the user.

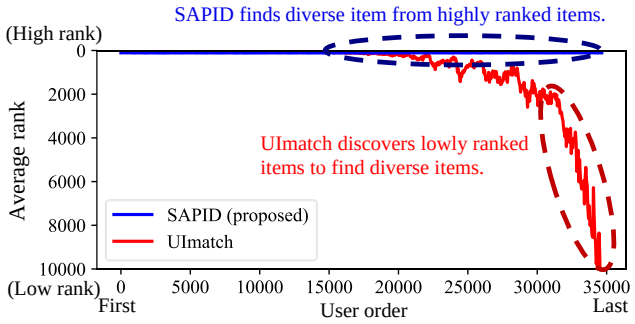


Figure 8: The average rank of recommended items of SAPID and UImatch on Gowalla dataset. Unlike UImatch, SAPID consistently recommends highly ranked items for all users.

We observe that SAPID consistently recommends highly ranked items throughout all recommendations, whereas UImatch's recommendation quality significantly drops for the latter users. This is because the greedy approach exhausts all popular items early on, leaving less relevant items to be recommended to the latter users to achieve higher diversity as shown in Figure 3(b). In contrast, SAPID forms a candidate pool by considering the distribution of item preferences for each user and then prioritizes items that are expected to be recommended less in the future. Thus, SAPID maintains high recommendation quality while also enhancing diversity.

6 Conclusion

In this paper, we introduce the problem of sequentially diversified recommendation, an important problem to maximize the potential profit of e-commerce platforms. We also suggest an evaluation metric that aligns with the problem. We propose SAPID, an effective method for the problem to achieve high accuracy and diversity. SAPID removes the popularity bias of the base model by adjusting the negative sampling probability based on each item's temporal popularity. Then, SAPID collects a set of recommendable items for each user based on their preference scores. Finally, SAPID generates a recommendation list considering which items to recommend immediately and which items to recommend later based on their potential chances to be exposed later. SAPID shows the state-of-the-art performance in real-world datasets by improving up to 61.0% diversity with 38.9% higher accuracy compared to the second best competitor. Future works include extending this work to achieve diversity and accuracy for sequential bundle recommendation.

Acknowledgments

This work was supported by Jung-Hun Foundation. This work was also supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) [No.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)], and [NO.RS-2021-II212068, Artificial Intelligence Innovation Hub (Artificial Intelligence Institute, Seoul National University)]. The Institute of Engineering Research at Seoul National University provided research facilities for this work. The ICT at Seoul National University provides research facilities for this study. U Kang is the corresponding author.

References

[1] Gediminas Adomavicius and YoungOk Kwon. 2012. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. *IEEE Trans. Knowl. Data Eng.* 24, 5 (2012), 896–911.

[2] Erik Brynjolfsson, Yu (Jeffrey) Hu, and Duncan Simester. 2011. Goodbye Pareto Principle, Hello Long Tail: The Effect of Search Costs on the Concentration of Product Sales. *Manag. Sci.* 57, 8 (2011), 1373–1386.

[3] Yukuo Cen, Jianwei Zhang, Xu Zou, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. Controllable Multi-Interest Framework for Recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*. ACM, 2942–2951.

[4] Laming Chen, Guoxin Zhang, and Eric Zhou. 2018. Fast Greedy MAP Inference for Determinantal Point Process to Improve Recommendation Diversity. (2018), 5627–5638.

[5] Wanyu Chen, Pengjie Ren, Fei Cai, Fei Sun, and Maarten de Rijke. 2020. Improving End-to-End Sequential Recommendations with Intent-aware Diversification. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*. ACM, 175–184.

[6] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011*. ACM, 1082–1090.

[7] Qiang Dong, Shuang-Shuang Xie, and Wen-Jun Li. 2021. User-Item Matching for Recommendation Fairness. *IEEE Access* 9 (2021), 130389–130398.

[8] Daniel G Goldstein and Dominique C Goldstein. 2006. Profiting from the long tail. *Harvard Business Review* 84, 6 (2006), 24–28.

[9] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4 (2016), 19:1–19:19.

[10] Ruining He and Julian J. McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*. ACM, 507–517.

[11] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. (2016).

[12] Seongwon Jang, Hoyeop Lee, Hyunsouk Cho, and Sehee Chung. 2020. CITIES: Contextual Inference of Tail-item Embeddings for Sequential Recommendation. In *20th IEEE International Conference on Data Mining, ICDM 2020, Sorrento, Italy, November 17-20, 2020*. IEEE, 202–211.

[13] Hyunsik Jeon, Jun-Gi Jang, Taehun Kim, and U Kang. 2023. Accurate bundle matching and generation via multitask learning with partially shared parameters. *Plos one* 18, 3 (2023), e0280630.

[14] Hyunsik Jeon, Jongjin Kim, Jaeri Lee, Jong-eun Lee, and U Kang. 2023. Aggregately Diversified Bundle Recommendation via Popularity Debiasing and Configuration-Aware Reranking. In *Advances in Knowledge Discovery and Data Mining - 27th Pacific-Asia Conference on Knowledge Discovery and Data Mining, PAKDD 2023, Osaka, Japan, May 25-28, 2023, Proceedings, Part III (Lecture Notes in Computer Science, Vol. 13937)*. Springer, 348–360.

[15] Hyunsik Jeon, Jongjin Kim, Hoyoung Yoon, Jaeri Lee, and U Kang. 2022. Accurate Action Recommendation for Smart Home via Two-Level Encoders and Commonsense Knowledge. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*. ACM, 832–841.

[16] Hyunsik Jeon, Bonhun Koo, and U Kang. 2019. Data Context Adaptation for Accurate Recommendation with Additional Information. In *2019 IEEE International Conference on Big Data (IEEE BigData), Los Angeles, CA, USA, December 9-12, 2019*. IEEE, 800–809.

[17] Hyunsik Jeon, Jong-eun Lee, Jeongin Yun, and U Kang. 2024. Cold-start Bundle Recommendation via Popularity-based Coalescence and Curriculum Heating. In *Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, May 13-17, 2024*. ACM, 3277–3286.

[18] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*. IEEE Computer Society, 197–206.

[19] Mahmut Özge Karakaya and Tevfik Aytekin. 2018. Effective methods for increasing aggregate diversity in recommender systems. *Knowl. Inf. Syst.* 56, 2 (2018), 355–372.

[20] Jongjin Kim, Hyunsik Jeon, Jaeri Lee, and U Kang. 2023. Diversely Regularized Matrix Factorization for Accurate and Aggregately Diversified Recommendation. In *Advances in Knowledge Discovery and Data Mining - 27th Pacific-Asia Conference on Knowledge Discovery and Data Mining, PAKDD 2023, Osaka, Japan, May 25-28, 2023, Proceedings, Part III (Lecture Notes in Computer Science, Vol. 13937)*. Springer, 361–373.

[21] Yejin Kim, Kwangseob Kim, Chanyoung Park, and Hwanjo Yu. 2019. Sequential and Diverse Recommendation with Long Tail. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. ijcai.org, 2740–2746.

[22] Bonhun Koo, Hyunsik Jeon, and U Kang. 2020. Accurate News Recommendation Coalescing Personal and Global Temporal Preferences. In *Advances in Knowledge Discovery and Data Mining - 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11-14, 2020, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 12084)*. Springer, 78–90.

[23] Bonhun Koo, Hyunsik Jeon, and U Kang. 2021. PGT: news recommendation coalescing personal and global temporal preferences. *Knowl. Inf. Syst.* 63, 12 (2021), 3139–3158.

[24] Jaeri Lee, Jeongin Yun, and U Kang. 2024. Towards True Multi-interest Recommendation: Enhanced Scheme for Balanced Interest Training. In *2024 IEEE International Conference on Big Data (IEEE BigData), Washington DC, USA, December 15-18, 2024*. IEEE.

[25] Siyi Liu and Yujia Zheng. 2020. Long-tail Session-based Recommendation. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020*. ACM, 509–514.

[26] Yudan Liu, Kaikai Ge, Xu Zhang, and Leyu Lin. 2019. Real-time Attention Based Look-alike Model for Recommender System. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*. ACM, 2765–2773.

[27] Masoud Mansoury, Himan Abdollahpouri, Mykola Pechenizkiy, Bamshad Mobasher, and Robin Burke. 2020. FairMatch: A Graph-based Approach for Improving Aggregate Diversity in Recommender Systems. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2020, Genoa, Italy, July 12-18, 2020*. ACM, 154–162.

[28] Julian J. 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, Santiago, Chile, August 9-13, 2015*. ACM, 43–52.

[29] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*. 452–461.

[30] Harald Steck. 2011. Item popularity and recommendation accuracy. In *Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011*. ACM, 125–132.

[31] Hao Wang, Zonghu Wang, and Weishi Zhang. 2018. Quantitative analysis of Matthew effect and sparsity problem of recommender systems. In *2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*. IEEE, 78–82.

[32] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet A. Orgun. 2019. Sequential Recommender Systems: Challenges, Progress and Prospects. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. ijcai.org, 6332–6338.

[33] Tianxin Wei, Fuli Feng, Jiawei Chen, Ziwei Wu, Jinfeng Yi, and Xiangnan He. 2021. Model-Agnostic Counterfactual Reasoning for Eliminating Popularity Bias in Recommender System. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*. ACM, 1791–1800.

[34] Mark Wilhelm, Ajith Ramanathan, Alexander Bonomo, Sagar Jain, Ed H. Chi, and Jennifer Gillenwater. 2018. Practical Diversified Recommendations on YouTube with Determinantal Point Processes. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*. ACM, 2165–2173.

[35] Qiong Wu, Yong Liu, Chunyan Miao, Yin Zhao, Lu Guan, and Haihong Tang. 2019. Recent Advances in Diversified Recommendation. *CoRR* abs/1905.06589 (2019). arXiv:1905.06589 http://arxiv.org/abs/1905.06589

[36] Liangwei Yang, Shengjie Wang, Yunzhe Tao, Jiankai Sun, Xiaolong Liu, Philip S. Yu, and Taiqing Wang. 2023. DGRec: Graph Neural Network for Recommendation with Diversified Embedding Generation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM 2023, Singapore, 27 February 2023 - 3 March 2023*. ACM, 661–669.

[37] Yu Zheng, Chen Gao, Liang Chen, Depeng Jin, and Yong Li. 2021. DGCN: Diversified Recommendation with Graph Convolutional Networks. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*. ACM / IW3C2, 401–412.