# Generalizable Recommender System During Temporal Popularity Distribution Shifts

Hyunsik Yoo
University of Illinois Urbana-Champaign
Urbana, IL, USA
hy40@illinois.edu

Ruizhong Qiu
University of Illinois Urbana-Champaign
Urbana, IL, USA
rq5@illinois.edu

Charlie Xu
Amazon.com, Inc.
Seattle, WA, USA
caizhx@amazon.com

Fei Wang
Amazon.com, Inc.
Sunnyvale, CA, USA
feiww@amazon.com

Hanghang Tong
University of Illinois Urbana-Champaign
Urbana, IL, USA
htong@illinois.edu

## Abstract

Many modern recommender systems represent user and item attributes as embedding vectors, relying on them for accurate recommendations. However, entangled embeddings often capture not only intrinsic property factors (e.g., user interest in item property) but also popularity factors (e.g., user conformity to item popularity) indistinguishably. These embeddings, influenced by popularity distribution, may face challenges when the popularity distribution at test time *differs* from historical distribution. Existing remedies in the literature involve disentangled embedding learning, which aims to separately capture intrinsic and popularity factors, demonstrating plausible generalization during popularity distribution shifts. However, we highlight that these methods often overlook a crucial aspect of popularity shifts—*their temporal nature*—in both training and inference phases. To address this, we propose Temporal Popularity distribution shift generalizABle recommender system (TPAB), a novel disentanglement framework incorporating temporal popularity. TPAB introduce a new (1) temporal-aware embedding design for users and items. Within this design, (2) popularity coarsening and (3) popularity bootstrapping are proposed to enhance generalization further. We also provide *theoretical analysis showing that the bootstrapping loss eliminates the effect of popularity on the learned model.* During inference, we infer test-time popularity and corresponding embeddings, using them alongside property embeddings for prediction. Extensive experiments on real-world datasets validate TPAB, showcasing its outstanding generalization ability during temporal popularity distribution shifts.[1]

## CCS Concepts

• **Information systems** → **Data mining**; • **Computing methodologies** → *Machine learning*.

## Keywords

recommender systems; temporal popularity distribution shifts; embedding disentanglement

**ACM Reference Format:**
Hyunsik Yoo, Ruizhong Qiu, Charlie Xu, Fei Wang, and Hanghang Tong. 2025. Generalizable Recommender System During Temporal Popularity Distribution Shifts. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3690624.3709299

**KDD Availability Link:** The source code of this paper has been made publicly available at https://doi.org/10.6084/m9.figshare.28236941.

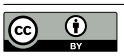
## 1 Introduction

To distill valuable insights from vast user-item interaction data, contemporary recommender systems often leverage embedding-based empirical risk minimization. They embed user preferences and item properties within their representations, and then use those embeddings to predict future interaction between users and items. However, the prevalent mechanism, which employs a single embedding for each entity (user or item) in the system, leads to the *entanglement* of an item's intrinsic property with its popularity derived from the overall popularity distribution. [2] These entanglement approaches may perform well under the assumption of an identical test distribution to historical training data, which unfortunately proves unrealistic [1, 18, 31, 35]. In reality, popularity distribution *continually evolves over time* [32, 36, 37].

An effective remedy to this challenge has emerged through *disentanglement* techniques [9, 27, 34, 35, 39], showcasing superiority over entangled counterparts, including popularity debiasing methods [11, 40], during popularity distribution shifts. The core concept involves disentangling intrinsic property factors and popularity factors by representing these distinct factors as separate embeddings. Generally, two additional losses are employed for disentanglement, focusing on (1) invariance learning and (2) geometric separation. Various techinques for the former objective have been proposed,

[1] The GitHub repository is available at https://github.com/hsyoo32/tpab_.

[2] Note that in the literature, the user-side counterparts of item property and popularity are often denoted as user's pure interest and conformity, respectively. For brevity, we omit these concepts throughout the introduction.




*KDD '25, Toronto, ON, Canada*

ACM ISBN 979-8-4007-1245-6/25/08
https://doi.org/10.1145/3690624.3709299

such as popularity-aware negative item sampling/learning [39], incorporating popularity as margins in the loss function [34], and popularity-intervened embedding learning [35]. For the latter objective, many methods aim to explicitly maximize distance/discrepancy metrics between property and popularity embeddings. They employ different metrics such as L1 [39], L2 [39], Pearson correlation coefficient [4], Maximum Mean Discrepancy (MMD) [22], and distance correlation [27, 35, 39].

Despite these strides, our contention in this paper is that existing disentanglement methods often overlook a crucial aspect of popularity shifts – *their temporal nature*. For example, in an e-commerce system, seasonal changes naturally affect the popularity distribution [35]. Also, while popular items may continue to gain popularity, they can also be surpassed by "new kids on the block" with better intrinsic quality over time [2, 7, 10]. Furthermore, some items may experience a natural fade in popularity over time, becoming outdated. We provide empirical validation using real-world datasets in Figure 1, which will be discussed further in Section 2.

However, such temporal dynamics of popularity are frequently neglected by existing methods in both *training* and *inference* phases. During training, popularity is viewed merely as static data derived from historical records, which likely lead to suboptimal performance of popularity embeddings for robustness in temporal popularity shifts [22, 34, 35, 39]. Moreover, in the complex temporal setting, their use of geometric separation techniques to maximize the distance between the two types of embeddings may compromise the expressiveness of embeddings [4, 22, 35, 39].

Besides that, during the inference phase, they often exclusively use property embeddings while disregarding popularity embeddings, which is suboptimal during temporal popularity shifts [34, 35]. This approach may be reasonable in case of uniform future distribution where there are no relative popularity difference affecting user behavior. However, in reality, it is reasonable to assume that item popularity persists at any given time and can distinctly influence user behavior, alongside the user's inherent interest in the item [36, 37]. For these reasons, a thoughtful approach is needed to intelligently learn and utilize popularity embeddings that accurately capture the temporal dynamics of popularity.

In this work, we aim to integrate the dynamic aspect of popularity with disentanglement principles by designing a novel method, Temporal Popularity distribution shift generalizABle recommender system (TPAB). TPAB introduces three distinct technical aspects: First, a new **(1) temporal-aware embedding design** for users and items, where the popularity embeddings capture temporal popularity and the item property embeddings capture intrinsic properties. Within this embedding design, **(2) popularity coarsening** and **(3) popularity bootstrapping** are proposed to further enhance generalization ability of TPAB.

Popularity coarsening allows items with *similar*–not just identical–popularity levels to share the same popularity embeddings, reducing TPAB's sensitivity to minor fluctuations in popularity. Popularity bootstrapping involves an additional risk minimization loss that replaces the popularity embedding with a randomly sampled popularity embedding, enhancing the invariance of property embeddings to the temporal popularity. Along with empirical evidence, we provide *theoretical analysis showing that this bootstrapping loss effectively eliminates the effect of popularity on the learned model*

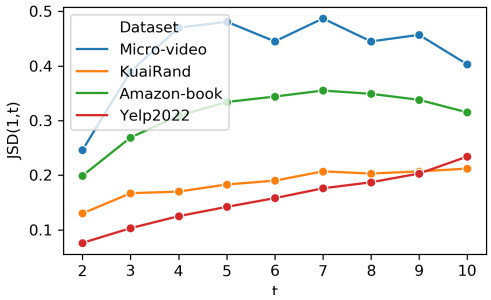



**Figure 1: Temporal popularity distribution shifts on four real-world datasets. The shifts (i.e., Jensen-Shannon Divergence (JSD)) between the initial and current stage (1,t) tend to increase over time.**

during training. Moreover, we observe that these ideas **(2&3)** reduce the variance in risks across different time stages, further supporting the rationale behind our algorithm design [15]. Finally, during inference, we predict the next-time popularity by classic time-series forecasting [36, 37], to derive test-time popularity embeddings for items. These embeddings are utilized alongside property embeddings for prediction. Via extensive experiments with real-world temporal datasets, we validate the strong generalization ability of TPAB during temporal popularity shifts, and the effectiveness of each of our algorithm designs.

In summary, the paper makes the following key contributions:

- **Theory.** We theoretically analyze the optimal solution for minimizing the bootstrapping loss. The analysis implies that the bootstrapping loss helps eliminate the effect of popularity on the learned model, and thus encourages property embeddings to be invariant to temporal popularity embeddings, as desired.
- **Algorithm.** We propose a new method named TPAB, which yields disentangled user/item embeddings with strong generalization ability under temporal popularity distribution shifts. TPAB incorporates three novel techniques: temporal-aware embedding design, popularity coarsening, and popularity bootstrapping.
- **Experiments.** Extensive experiments on real-world temporal recommendation datasets validate the effectiveness of TPAB, showing an average increase of 8.33% compared to the best-performing recent competitors.

## 2 Preliminary

In this section, we first present the key notations used throughout the paper. Next, we discuss temporal popularity distribution shifts, providing evidence of these shifts across four real-world datasets. We then explore generalization in temporal popularity shifts and the principles of disentanglement in recommender systems. Finally, we formally define the problem we aim to address in the paper.

**Notations.** Table 1 outlines the main symbols employed in this paper. Throughout the paper, we use bold upper-case letters for matrices (e.g., $\mathbf{Y}$), bold lower-case letters for vectors (e.g., $\mathbf{r}$) and calligraphic letters for sets (e.g., $\mathcal{U}$). We use conventions similar to NumPy [12] in Python for indexing. For example, $\mathbf{Y}[i, j]$ is the entry at the $i$-th row and the $j$-th column in matrix $\mathbf{Y}$.

For dataset representation at time stage $t$ $\forall t \in \{1, \ldots, T\}$, we denoted it as $\mathcal{D}_t = \{\mathcal{U}_t, \mathcal{I}_t, \mathcal{E}_t, \mathbf{Y}_t\}$, where $\mathcal{U}_t$ stands for the user set, $\mathcal{I}_t$ for the item set, $\mathcal{E}_t$ for the user-item interaction set, and $\mathbf{Y}_t$ for the user-item interaction matrix. The subscript $_t$ signifies the time stage $t$. [3] We consider binary user-item interaction in this work, where $\mathbf{Y}_t[u, i] = 1$ indicates interaction between user $u$ and item $i$ within the $t$-th time period, and 0 otherwise.

**Real-world temporal popularity distribution shifts.** We estimate item popularity at time stage $t$ using the number of interactions for each item: $p_i^t = |\mathcal{E}_t(i)|$. Each time stage $t$ is considered a distinct environment $e_t$ $\forall t \in \{1, \ldots, T\}$ with varying popularity distributions $[m_1^t, \cdots, m_{|\mathcal{I}_t|}^t]$, where $m_i^t = \frac{p_i^t}{\sum_{j \in \mathcal{I}_t} p_j^t}$, due to temporal shifts over time. Earlier work [36] highlights persistent popularity distribution shifts between consecutive stages, with the degree of shift tending to increase over time.

As an empirical validation, we analyze whether temporal popularity distribution shifts exist in the real-world datasets used in the paper. We examined the shift between the initial and current stage $(1, t)$ using the Jensen-Shannon Divergence (JSD) as shown in Figure 1. The results indicate that JSD$(1, t)$ tends to increase over time on all datasets, showing clear evidence of popularity distribution shifts over time.

**Generalization in temporal shifts.** Our objective is to ensure the generalization ability of a model trained on historical data to perform well in the future environment $e_{T+1}$, which is highly likely to exhibit a changed popularity distribution. Specifically, the model generates a top-$N$ recommendation list $[i_1, \ldots, i_N]$ for each user $u$, ranked by the predicted scores $s_{ui}^{T+1}, \forall i$, and this list should align closely with the user's actual decisions in the future environment.

**Disentanglement principles.** Drawing from prior disentanglement methods [34, 35, 39], we identify two factors influencing interaction label $Y$: (1) $Z_{\text{prop}}$: property factors related to user's pure interest in intrinsic item properties (2) $Z_{\text{pop}}$: popularity factors related to user's conformity influenced by item popularity. Typically, those factors are represented separately.

**Problem definition.** Our problem is formally defined as follows:

Problem 1 (Generalization in Temporal Popularity Distribution Shifts). ***Input:*** *a sequentially collected dataset* $\mathcal{D}_t = \{\mathcal{U}_t, \mathcal{I}_t, \mathcal{E}_t, \mathbf{Y}_t\}, \forall t \in \{1, \ldots, T\}$ *and the popularity* $p_i^t$ *of item* $i$ *at each time stage* $t$.

***Output:*** *an out-of-distribution generalizable model that delivers high-quality recommendations in a future time stage with a different popularity distribution from the past.*

## 3 Proposed Framework

In this section, we propose Temporal Popularity distribution shift generalizABle recommender system (TPAB), a novel recommender system designed to maintain robustness against temporal popularity distribution shifts. In summary, TPAB introduces three distinct technical aspects. First, it employs **(1) a temporal-aware disentangled embedding** framework for users and items, where the popularity embeddings capture temporal popularity (Section 3.1).

[3] Depending on the needs of the system or implementation, the time stage could be either a specific time frame (e.g., daily, weekly, monthly) or until a specific number of interactions has been collected.

**Table 1: Main symbols used in this paper.**

| Symbol | Description |
|---|---|
| $\mathcal{D}_t$ | Dataset collected at time stage $t$ |
| $\mathcal{U}_t, \mathcal{I}_t, \mathcal{E}_t$ | Sets of users, items, and their interactions at time stage $t$ |
| $U_{\text{int}}, U_{\text{conf}}$ | Interest and conformity embeddings for a user, respectively |
| $I_{\text{prop}}, I_{\text{pop}}$ | Property and popularity embeddings for an item, respectively |
| $s_{ui}$ | Final recommendation score between $u$ and $i$ |
| $\mathcal{L}_{\text{erm}}, \mathcal{L}_{\text{boot}}$ | Empirical risk minimization and bootstrapping loss |
| $p_i^t$ | Popularity of item $i$ at time stage $t$ |
| $C$ | Temporal popularity coarsening function |
| $K$ | The number of categories for popularity coarsening |
| $\lambda$ | Scaling parameter for $\mathcal{L}_{\text{boot}}$ |
| $\alpha$ | Controlling parameter for next-time popularity forecasting |

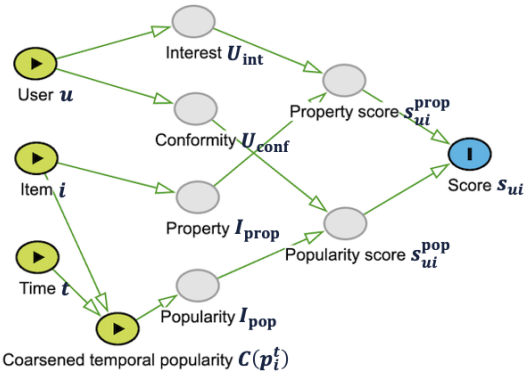



**Figure 2: Dependencies of embeddings and scores in TPAB.**

Then, within this framework, **(2) popularity coarsening** (Section 3.2) and **(3) popularity bootstrapping** (Section 3.3) are proposed to further enhance generalization. We also provide *theoretical analysis to demonstrate that the bootstrapping loss helps eliminate the effect of popularity on the learned model.* For training, TPAB jointly employs the standard empirical risk minimization loss and the bootstrapping loss (Section 3.4). During inference, TPAB first derives test-time popularity embeddings for items and integrates them with the property embeddings for future recommendations (Section 3.5). The detailed training and inference procedures of TPAB are elaborated in Algorithms 1 and 2, respectively.

### 3.1 Temporal-Aware Disentangled Embeddings

In most existing disentanglement methods, there is a prevailing design principle to separate an item's intrinsic *property* and its *popularity* into distinct embeddings. Similarly, user representations include counterparts for property and popularity: user *interest* in item property and user *conformity* to popularity, respectively. One of our key contributions extends beyond the conventional design that primarily addresses *static* popularity by proposing a new disentanglement mechanism that captures *temporal* popularity, separated from the invariant property.

We formally define our embeddings of user/item for each interaction $(u, i, t) \in \mathcal{E}_t$ $\forall t \in \{1, \ldots, T\}$ as follows:

- For an item $i$, two embedding functions, $I_{\text{prop}}$ and $I_{\text{pop}}$, capture the item's invariant property and changing popularity, respectively. To distinguish these features, we propose using the item's popularity $p_i^t$ at time stage $t$ as the categorical input for $I_{\text{pop}}$, while $I_{\text{prop}}$ takes the item ID as the input. Note that a popularity integer value is used as categorical input, similar to ID, as done in previous works [35]. Unlike those works, we use temporal popularity $p_i^t$ rather than global/static popularity $p_i$. The overall item representation is the concatenation of $I_{\text{prop}}$ and $I_{\text{pop}}$.
- For a user $u$, two embedding functions for user interest and conformity, $U_{\text{int}}$ and $U_{\text{conf}}$, correspond to $I_{\text{prop}}$ and $I_{\text{pop}}$, respectively. Both take the user ID as input, reflecting the exact same architecture but dependent on their item-side counterparts. We denote them by different names to emphasize their semantics. The overall user representation is concatenation of $U_{\text{int}}$ and $U_{\text{conf}}$.

All aforementioned embedding functions (i.e., $I_{\text{prop}}$, $I_{\text{pop}}$, $U_{\text{int}}$, and $U_{\text{conf}}$) are based on the same backbone recommendation model, but only $I_{\text{pop}}$ takes temporal popularity $p_i^t$ as input, which simplifies our method. The backbone models can be any that encode embedding vectors, such as MF [20] or LightGCN [8], and their embedding dimensions are all $d$. Then, we compute the user-item recommendation score $s_{ui}$ as follows:

$$s_{ui} := \langle U_{\text{int}}(u) || U_{\text{conf}}(u), I_{\text{prop}}(i) || I_{\text{pop}}(C(p_i^t)) \rangle, \tag{1}$$

where $||$ is a concatenation operator, $\langle \cdot, \cdot \rangle$ is the dot product, and $C(p_i^t)$ denote the *coarsened* temporal popularity. We will introduce popularity coarsening in Section 3.2. Note that this final recommendation score can also be interpreted as the sum of the property factor score $s_{ui}^{\text{prop}}$ and the popularity factor score $s_{ui}^{\text{pop}}$ as follows:

$$s_{ui} := s_{ui}^{\text{prop}} + s_{ui}^{\text{pop}}, \tag{2}$$

$$s_{ui}^{\text{prop}} := \langle U_{\text{int}}(u), I_{\text{prop}}(i) \rangle, \quad s_{ui}^{\text{pop}} := \langle U_{\text{conf}}(u), I_{\text{pop}}(C(p_i^t)) \rangle. \tag{3}$$

$s_{ui}^{\text{prop}}$ and $s_{ui}^{\text{prop}}$ represent how much the corresponding factors influence the user-item interaction. Refer to Figure 2 for a graphical representation illustrating the dependencies between embeddings and scores in TPAB.

## 3.2 Popularity Coarsening

Based on such embedding design leveraging the temporal popularity, we propose a new technique of *popularity coarsening* with exponential bucketing for $I_{\text{pop}}$. This technique offers two key advantages: (1) reducing the model's sensitivity to item popularity and (2) mitigating size differences between different buckets.

First, we define the coarsening function that maps each popularity $p_i^t$ to $K$ categories (i.e., bucketing), where $K$ is a hyperparameter:

$$C(p_i^t) := k \quad \text{if} \quad p_{\max}^{\frac{k-1}{K}} < p_i^t \le p_{\max}^{\frac{k}{K}}, \ \forall k \in \{1, \dots, K\}, \tag{4}$$

where $p_{\max}$ represents the maximum temporal popularity (i.e., $p_{\max} = \max(p_i^t)$). Then, we use this coarsened popularity ID (i.e., $k$) as an input for the popularity embedding as $I_{\text{pop}}(C(p_i^t))$. This coarsening encourages TPAB to be less sensitive to minor fluctuations in item popularity, thus enhancing generalizability on unseen data. Without coarsening, items with "same" popularity share the same popularity embeddings. However, with coarsening, items with a "similar" level of popularity share the same popularity embeddings.

**Algorithm 1** Training procedure of TPAB

**Input:** Coarsening parameter $K$, scaling parameter $\lambda$, popularity trend drift parameter $\alpha$, the number of negative items $n$ for $\mathcal{L}_{\text{bpr}}$ and $\mathcal{L}_{\text{boot}}$, input dataset $\mathcal{D}_t = \{\mathcal{U}_t, \mathcal{I}_t, \mathcal{E}_t, \mathrm{Y}_t\}_{t=1}^T$, initialized model parameters $\mathcal{W} = \{U_{\text{int}}, U_{\text{conf}}, I_{\text{prop}}, I_{\text{pop}}\}$
**Output:** Updated model parameters $\mathcal{W}$
Set $p_i^t \leftarrow |\mathcal{E}_t(i)|, \forall i \in \mathcal{I}_t, \forall t = 1, \cdots, T$
**for** each epoch **do**
  **for** mini-batch $\mathcal{B}$ obtained from $\mathcal{E}$ **do**
    Negative set $\mathcal{N} \leftarrow \{\}$;
    **for** user-item interaction $(u, i) \in \mathcal{B}$ **do**
      Sample $n$ negative items as $\mathcal{N}_{ui}$;
      Update $\mathcal{N} \leftarrow \mathcal{N} \cup \{(u, i')\}_{i' \in \mathcal{N}_{ui}}$;
    **end for**
    Compute the ERM loss $\mathcal{L}_{\text{bpr}}$ with $\mathcal{B} \cup \mathcal{N}$ by Eq. (9);
    Randomly sample $\mathcal{R} \leftarrow \{(u, i, p')\}_{(u,i) \in \mathcal{B} \cup \mathcal{N}}$;
    Compute the bootstrapping loss $\mathcal{L}_{\text{boot}}$ with $\mathcal{R}$ by Eq. (10);
    Update $\mathcal{W}$ based on $\mathcal{L}_{\text{bpr}} + \lambda \mathcal{L}_{\text{boot}}$ via gradient descent;
  **end for**
**end for**

**Algorithm 2** Inference procedure of TPAB

**Input:** Temporal item popularity $p_i^t$, trained parameters $\mathcal{W}$
**Output:** Top-$N$ recommendation lists
Predict $\hat{p}_i^{T+1} \leftarrow p_i^T + \alpha(p_i^T - p_i^{T-1})$;
**for** user $u \in \mathcal{U}$ **do**
  $S_{ui}^{T+1} \leftarrow \langle U_{\text{int}}(u) || U_{\text{conf}}(u), I_{\text{prop}}(i) || I_{\text{pop}}(C(\hat{p}_i^{T+1})) \rangle, \forall i \in \mathcal{I}$;
  Generate top-$N$ recommendation list for $u$ based on $S_{ui}^{T+1}$;
**end for**

Second, we choose exponential bucketing over uniform bucketing (i.e., $C(p_i^t) := k$ if $\frac{p_{\max} \cdot (k-1)}{K} < p_i^t \le \frac{p_{\max} \cdot k}{K}$, $\forall k \in \{1, \dots, K\}$) to address the size difference across different popularity buckets. This size difference is a noticeable problem, even without coarsening, due to the typical long-tail distribution of item popularity, where there are many unpopular items and few popular items [4, 23] (e.g., the number of items with $p = 5$ is much larger than the number of items with $p = 50$). Our coarsening addresses this size difference by exponentially grouping popularity levels (e.g., $p = 5 \sim 10$ is grouped by $k = 2$, and $p = 10 \sim 50$ is grouped by $k = 3$).

Based on our scoring mechanism with coarsened temporal popularity (Eq. (1)), which reduces TPAB's sensitivity to minor popularity fluctuations, we can use any empirical risk minimization (ERM) loss such as Bayesian Personalized Ranking (BPR) or Sampled Softmax loss. In this work, owing to its efficiency and consistency with our theory on bootstrapping loss, we use BPR loss, denoted as $\mathcal{L}_{\text{bpr}}(\{s_{ui}\}_{u \in \mathcal{U}, i \in \mathcal{I}})$, which will be detailed in Section 3.4

## 3.3 Popularity Bootstrapping

While the embedding design with coarsened temporal popularity enhances the generalizability of TPAB on future data, the disentanglement between property and popularity factors may still be suboptimal when only using the standard empirical risk minimization

loss ($\mathcal{L}_{\text{bpr}}$). To address this, we propose the concept of *popularity bootstrapping*, which enhances the invariance of property embeddings to the popularity. Specifically, we design an additional risk minimization loss that replaces the temporal popularity embedding with a randomly sampled temporal popularity embedding, defined as follows:

$$\mathcal{L}_{\text{boot}} \coloneqq \mathcal{L}_{\text{bpr}}(\{\tilde{s}_{ui}|p'\}_{u\in\mathcal{U},i\in\mathcal{I}}), \tag{5}$$

$$\tilde{s}_{ui} \coloneqq \langle U_{\text{int}}(u)||U_{\text{conf}}(u), I_{\text{prop}}(i)||I_{\text{pop}}(C(p'))\rangle. \tag{6}$$

where $p'$ denotes a randomly sampled popularity of an arbitrary item at an arbitrary time stage, and thus, $C(p')$ represents randomly sampled coarsened popularity from a sample size of $K$. Optimizing $\mathcal{L}_{\text{boot}}$ encourages $I_{\text{prop}}$ to more accurately learn intrinsic item properties that remain invariant to the temporal popularity $I_{\text{pop}}$. Beyond intuition, we provide theoretical analysis on the effect of the bootstrapping loss in the training of TPAB.

Theorem 3.1 (Bootstrapping Loss). *For simplicity, let $p$ denote $C(p)$, which represents coarsened popularity from a sample size of $K$. Given a triplet for the BPR loss $(u, i_1, i_2)$, where $i_1$ is a positive item and $i_2$ is a negative item, let $s_1(p_1) = s_1^{prop} + s_{p_1}^{pop}$ be the true score based on the true popularity $p_1$ of the positive pair $(u, i_1)$, and $\hat{s}_1(p'_1) = \hat{s}_1^{prop} + \hat{s}_2^{prop}$ be the predicted score based on randomly sampled popularity $p'_1$. Similarly, we define $s_2(p_2)$ and $\hat{s}_2(p'_2)$ for negative pair $(u, i_2)$. Assuming that the user chooses either $i_1$ and $i_2$ with probability $\sigma(s_1(p_1) - s_2(p_2))$ and $\sigma(s_2(p_2) - s_1(p_1))$, respectively, the optimal solution for minimizing the bootstrapping loss is given by:*

$$\hat{s}_1^{prop} - \hat{s}_2^{prop} = \log \frac{\mathbb{E}_{p_1,p_2}\, \sigma(s_1(p_1) - s_2(p_2))}{\mathbb{E}_{p_1,p_2}\, \sigma(s_2(p_2) - s_1(p_1))}; \tag{7}$$

$$\hat{s}_{p'_1}^{pop} = \hat{s}_{p'_2}^{pop}, \quad \forall p'_1, p'_2. \tag{8}$$

Proof is in Section A. Theorem 3.1 presents two conclusions. First, Eq. (7) shows that only property scores affect the equation, with no influence from popularity scores. Second, Eq. (8) shows that popularity scores are the same for every possible popularity value, implying that the bootstrapping loss effectively eliminates the effect of popularity on the learned model. Thus, when jointly used with our ERM loss, which aims to accurately learn both property and popularity embeddings, and when employing temporal popularity, the bootstrapping loss promotes invariance of the property embeddings to temporal popularity changes. In Section 4.3, we empirically verify the effectiveness of the bootstrapping loss.

## 3.4 Training Protocol

Our BPR loss [20], which we use as ERM loss, is defined as follows:

$$\mathcal{L}_{\text{bpr}} \coloneqq -\frac{1}{|\mathcal{E}|} \sum_{(u,i,t)\in\mathcal{E}} \frac{1}{|\mathcal{N}_{ui}|} \sum_{i'\in\mathcal{N}_{ui}} \log(\sigma(s_{ui} - s_{ui'})), \tag{9}$$

where $\sigma(\cdot)$ is the sigmoid function, and $\mathcal{N}_{ui}$ is a set of sampled negative items for $(u, i)$. By optimizing $\mathcal{L}_{\text{bpr}}$, $I_{\text{prop}}$ and $I_{\text{pop}}$ become adept at capturing intrinsic item characteristics and temporal popularity, respectively. We let the user representations ($U_{\text{int}}$ and $U_{\text{conf}}$) naturally adapt to their corresponding item-side counterparts.

To be specific, $\mathcal{L}_{\text{boot}}$ (Eq. (5)) is formulated as follows:

$$\mathcal{L}_{\text{boot}} \coloneqq -\frac{1}{|\mathcal{E}|} \sum_{(u,i,t)\in\mathcal{E}} \frac{1}{|\mathcal{N}_{ui}|} \sum_{i'\in\mathcal{N}_{ui}} \log(\sigma(\tilde{s}_{ui} - \tilde{s}_{ui'})), \tag{10}$$

where the only difference from $\mathcal{L}_{\text{bpr}}$ is the use of bootstrapped scores for both positive and negative items.

For our final loss, we jointly use $\mathcal{L}_{\text{erm}}$ and $\mathcal{L}_{\text{boot}}$ as follows:

$$\mathcal{L} \coloneqq \mathcal{L}_{\text{bpr}} + \lambda\mathcal{L}_{\text{boot}}, \tag{11}$$

where $\lambda$ is a scaling parameter that controls the degree of the bootstrapping loss. By training the final loss in Eq. (11), TPAB achieves the disentangled user/item embeddings that generalize well during temporal popularity distribution shifts.

**Remark (variance in risks).** Note that [15] demonstrated the benefits of equalizing training risks (e.g., loss value) across different environments for enhanced generalization. Guided by these insights, in Section 4.4, we present evidence of the effectiveness of our popularity coarsening (Eq. (4)) and bootstrapping (Eq. (5)) not only in the overall accuracy but also in the reduction of variance in risks. In essence, both techniques encourage items across different time stages to share the popularity embeddings, by imposing coarsening and bootstrapping (randomness), which may lead to smaller variances and thereby indicate enhanced generalization ability.

## 3.5 Inference Protocol

For inference, we first infer the next-time popularity $\hat{p}_i^{T+1}$ for every item $i$ using classic time-series forecasting, following previous works [36, 37]:

$$\hat{p}_i^{T+1} \coloneqq p_i^T + \alpha(p_i^T - p_i^{T-1}), \tag{12}$$

where $\alpha$ is a hyperparameter that controls the popularity trend drift. With this predicted popularity, we get the next-time popularity embedding of $i$ as follows: $I_{\text{pop}}(C(\hat{p}_i^{T+1}))$. Note that if the predicted popularity exceeds the maximum popularity $p_{\max}$, we approximate it to $p_{\max}$.

Then, the top-$N$ recommendation list for each user is generated based on predicted scores as follows:

$$s_{ui}^{T+1} \coloneqq \langle U_{\text{int}}(u)||U_{\text{conf}}(u), I_{\text{prop}}(i)||I_{\text{pop}}(C(\hat{p}_i^{T+1}))\rangle. \tag{13}$$

By uniquely and accurately capturing item property and temporal popularity in representations, our model achieves strong out-of-distribution generalization in the temporal popularity distribution shifts, as we will demonstrate in Section 4.

**Complexity analysis.** Our bootstrapping loss introduces only a marginal increase in the time complexity to the ERM loss. Assuming we employ MF [20] as the base recommendation model with user/item sub-embeddings of dimensionality $d$, the time complexity of minimizing $\mathcal{L}_{\text{bpr}}$ is $O(|\mathcal{E}|nd)$, where $n$ represents the number of negative items. Given that the bootstrapping loss $\mathcal{L}_{\text{boot}}$ shares the same structure as $\mathcal{L}_{\text{bpr}}$, differing only in the randomly sampled popularity $p'$, the time complexity of minimizing it remains $O(|\mathcal{E}|nd)$. The time complexity for computing the temporal popularity $p_i^t \coloneqq |\mathcal{E}_t(i)|, \forall i \in \mathcal{I}_t, \forall t = 1, \cdots, T$, is $O(|\mathcal{I}|T)$, and computing the next-time popularity $\hat{p}_i^{T+1}$ is $O(|\mathcal{I}|)$. Both are negligible compared to that of the losses and require only a one-time pre-computation before model training.

# 4 Experiments

We design experiments to answer the key research questions (RQs):

**RQ1.** To what extent does TPAB outperform its competitors?
**RQ2.** How does each design in TPAB enhance generalization?
**RQ3.** How does geometric separation affect embeddings of TPAB?
**RQ4.** How sensitive is TPAB to its hyperparameter?
**RQ5.** How time-efficient is TPAB compared to its competitors?

## 4.1 Experimental Settings

*4.1.1 Datasets.* We use the following real-world temporal datasets.

- **Micro-video** [37]: This dataset is an industrial dataset collected from a micro-video APP. It contains 210, 550 interactions between 25, 871 users and 44, 503 micro-videos over one month. Specifically, it sets $\mathbf{Y}[u, i] = 1$ if user $u$ played a micro-video $i$ for durations exceeding 8 seconds, and $\mathbf{Y}[u, i] = 0$ otherwise.
- **KuaiRand**[4]: This popular video recommendation dataset contains 621, 064 click interactions on 7, 076 movies by 22, 128 users over one month.

To simulate temporal popularity distribution shifts, we first sort the interactions chronologically and then partition them into training, validation, and testing sets with a ratio of 8:1:1. We then further split the training set into 8 time stages, each with an equivalent time range. This process yields $\{\mathcal{D}_1, \cdots, \mathcal{D}_T, \mathcal{D}_{\text{val}}, \mathcal{D}_{\text{test}}\}$, where $T = 8$. As Figure 1 shows, the test and training popularity distributions are not identical, reflecting temporal distribution shifts.

Additionally, to test the model's generalizability over longer time periods, we show results on larger/longer datasets **Amazon-book** and **Yelp2022**, which span 17 years and 10 years, in Appendix B.

*4.1.2 Compared methods.* To ensure the independence of TPAB's effectiveness from the base recommender system used, we use two base models, Matrix Factorization (MF) [20] and LightGCN [8], both with the Bayesian Personalized Ranking (BPR) loss [20].

We compare TPAB with other competitors designed to generalize well in unseen popularity distribution shifts. The competitors fall into two categories. First, the (entangled) competitors leveraging temporal popularity include:

- PDA (Popularity-bias deconfounding/adjusting) [36]: This method estimates future popularity for each item to recalibrate user-item matching scores in both the training and inference phases.
- PDRO (Popularity-aware distributionally robust optimization) [37]: Building on the PDA approach, this method incorporates temporal popularity into the group-DRO [29] for recommendations.

Second, the disentangled competitors are all based on two embeddings for users and items, similar to TPAB. However, they only consider the static popularity, while TPAB considers temporal the dynamics in popularity. Those competitors are listed as follows:

- DICE [39]: This method aims for disentanglement based on popularity-aware negative item sampling/learning.
- MARGIN [34]: This method achieves disentanglement by incorporating popularity factors as margins in the loss function based on property factors. It only uses invariant interest/property embeddings for inference.
- INVCF [35]: This method aims for disentanglement based on representation augmentation and disentanglement modules. It only uses invariant interest/property embeddings for inference.
- DCCL [38]: This method achieves disentanglement based on popularity-aware augmentation and contrastive learning.

Additionally, we conduct an ablation study of TPAB, which utilizes three distinct technical aspects: (1) temporal-aware disentangled embeddings, (2) popularity coarsening, and (3) popularity bootstrapping. Each of these techniques is orthogonal to the others, meaning that any can be independently toggled on or off within TPAB. [5] We introduce three variants of TPAB as follows:

- TPAB-T: TPAB uses static/global popularity $p_i$ for each item, instead of temporal popularity $p_i^t$.
- TPAB-C: TPAB without the popularity coarsening ($C(\cdot)$).
- TPAB-B: TPAB without the popularity bootstrapping ($\mathcal{L}_{\text{boot}}$).

*4.1.3 Evaluation.* We employ the all-item-ranking method (i.e., using all items that a user did not interact with as candidate items for recommendation) to assess the top-$N$ recommendation accuracy [14]. Our evaluation metrics include Recall@$N$ and NDCG@$N$ (normalized discounted accumulated gain), where $N = 10$ or 20.

*4.1.4 Implementation details.* For all compared methods, we ensure consistency by employing one negative user-item pair for each positive user-item pair in BPR loss, setting the learning rate to 0.001, and applying L2 regularization of 0.0001. Model parameters are updated using the Adam optimization algorithm [13]. For the implementation details of all the competitors, refer to Appendix D.

For TPAB, we choose $K$ values from [20, 40, 60] and $\lambda$ values from [0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0]. Specifically, we set $K = 20$ and $\lambda = 1.0$ for Micro-video in all cases, $K = 60$ and $\lambda = 3.0$ for MF-based models on KuaiRand, and $K = 20$ and $\lambda = 2.0$ for LightGCN-based models on KuaiRand (See Section 4.5). We also maintain $\alpha = 0.2$ in all scenarios, in line with previous research [37]. To ensure reproducibility, we utilize a random seed during experimentation and report the mean and standard deviation values from five runs. We will release the source code upon the publication of the paper.

## 4.2 Main Results

To answer RQ1, we compare TPAB with seven competitors, evaluating their recommendation performance in temporal popularity distribution shifts. Table 2 shows the results across various metrics, base recommendation models, and datasets. Boldface and underlined values signify the best and second-best performances in each column for each base model.

First, no single competitor consistently outperforms others; the best one varies based on metrics, base models, and datasets. However, TPAB consistently outperforms all competitors, with an average improvement of 8.33% over the *best* competitor in each case (spec., 12.43%, 9.38%, 10.84% over PDA, DICE, DCCL, respectively). These results underscore TPAB as effective in tackling the out-of-distribution generalization challenge posed by temporal popularity distribution shifts.

Temporal popularity-aware competitors (PDA and PDRO) leverage the concept of calibrating original recommendation scores with

[4] https://kuairand.com/.

[5] For instance, even without coarsening, TPAB still employs temporal popularity as a categorical input for popularity embedding functions, as detailed in Section 3.1.

**Table 2: Recommendation performance of TPAB and seven competitors using MF and LightGCN (LGCN) base models. TPAB consistently outperforms all competitors across all metrics on both datasets, showcasing its superior generalization ability.**

| Models | | Micro-video | | | | KuaiRand | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Recall@10 | NDCG@10 | Recall@20 | NDCG@20 | Recall@10 | NDCG@10 | Recall@20 | NDCG@20 |
| MF | Vanilla | 0.0755±0.0023 | 0.0532±0.0021 | 0.1139±0.0021 | 0.0656±0.002 | 0.0642±0.001 | 0.0426±0.0005 | 0.1111±0.0016 | 0.0581±0.0007 |
| | PDA | 0.0757±0.0022 | 0.0531±0.0016 | 0.1143±0.0023 | 0.0657±0.0019 | 0.065±0.0014 | 0.0431±0.0008 | 0.1126±0.0014 | 0.0588±0.0009 |
| | PDRO | 0.073±0.0016 | 0.0516±0.0008 | 0.112±0.0013 | 0.0642±0.0008 | 0.0586±0.0021 | 0.0386±0.0015 | 0.1016±0.0028 | 0.0529±0.0017 |
| | DICE | 0.0792±0.0027 | **0.0579±0.0022** | 0.1144±0.003 | 0.0693±0.0024 | 0.0713±0.0013 | 0.0471±0.0006 | 0.1212±0.0025 | 0.0637±0.0009 |
| | Margin | 0.0611±0.0021 | 0.0426±0.0007 | 0.0926±0.0035 | 0.0528±0.001 | 0.0558±0.001 | 0.0378±0.001 | 0.0937±0.0009 | 0.0503±0.0008 |
| | InvCF | 0.0759±0.0025 | 0.0534±0.0016 | 0.116±0.0021 | 0.0665±0.0016 | 0.0576±0.0007 | 0.0384±0.0007 | 0.0987±0.0016 | 0.0522±0.0009 |
| | DCCL | 0.0768±0.0017 | 0.0545±0.0014 | 0.1163±0.0019 | 0.0674±0.001 | 0.0683±0.0013 | 0.0454±0.0009 | 0.118±0.0017 | 0.0619±0.0011 |
| | TPAB | **0.0829±0.0021** | 0.0578±0.0023 | **0.1238±0.0036** | **0.0712±0.0027** | **0.0827±0.001** | **0.0549±0.0004** | **0.138±0.0019** | **0.0732±0.0005** |
| | Imp. / best | 4.67% | -0.17% | 6.45% | 2.74% | 15.15% | 15.92% | 13.86% | 14.76% |
| LGCN | Vanilla | 0.0857±0.0024 | 0.0591±0.0018 | 0.1274±0.0029 | 0.0724±0.0019 | 0.0746±0.0003 | 0.0497±0.0003 | 0.1281±0.0012 | 0.0672±0.0005 |
| | PDA | 0.0865±0.0026 | 0.0593±0.0019 | 0.1278±0.0032 | 0.0726±0.002 | 0.0768±0.0006 | 0.0513±0.0003 | 0.1306±0.001 | 0.069±0.0003 |
| | PDRO | 0.0867±0.0011 | 0.0599±0.0008 | 0.1264±0.0021 | 0.0728±0.0011 | 0.0749±0.0006 | 0.05±0.0003 | 0.1272±0.0005 | 0.0672±0.0002 |
| | DICE | 0.0855±0.0009 | 0.0591±0.0013 | 0.1283±0.0025 | 0.0728±0.0016 | 0.0751±0.0006 | 0.05±0.0002 | 0.1283±0.0005 | 0.0674±0.0002 |
| | Margin | 0.0788±0.0018 | 0.0537±0.0004 | 0.1152±0.0019 | 0.0656±0.0004 | 0.0647±0.001 | 0.0432±0.0005 | 0.1114±0.0006 | 0.0585±0.0003 |
| | InvCF | 0.0875±0.0014 | 0.0593±0.0014 | 0.1278±0.0014 | 0.0723±0.0013 | 0.0742±0.0003 | 0.0496±0.0002 | 0.1271±0.0006 | 0.067±0.0002 |
| | DCCL | 0.0877±0.002 | 0.0599±0.0012 | 0.1301±0.0019 | 0.0737±0.001 | 0.0743±0.0002 | 0.0495±0.0001 | 0.1277±0.0006 | 0.067±0.0002 |
| | TPAB | **0.0896±0.0018** | **0.0616±0.0013** | **0.1336±0.0017** | **0.0759±0.0013** | **0.0864±0.0003** | **0.0569±0.0003** | **0.147±0.0008** | **0.0769±0.0004** |
| | Imp. / best | 2.17% | 2.84% | 2.69% | 2.99% | 12.50% | 10.92% | 12.56% | 11.45% |

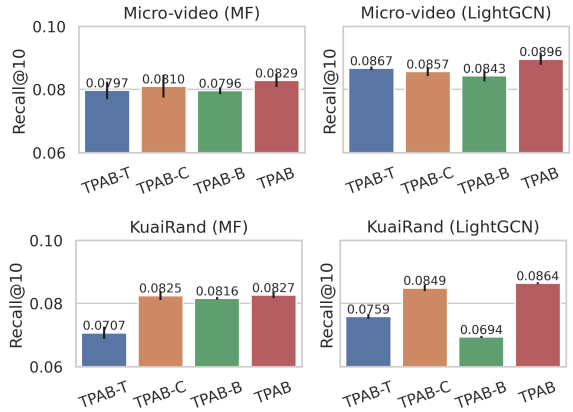



**Figure 3: Ablation study of TPAB w.r.t. recommendation accuracy. Each component—temporal popularity, popularity coarsening, and popularity bootstrapping—is effective. When employed together, TPAB achieves the best accuracy.**

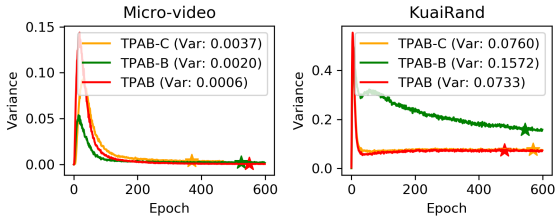



**Figure 4: Ablation study of TPAB w.r.t. variance in risks across time stages. Popularity coarsening and bootstrapping reduces variance, supporting their effectiveness in enhancing generalization of TPAB.**

predicted next-time popularity, and we empirically observe that such concept contributes significantly to their performance. However, their entangled embeddings of user/item may indiscriminately prioritize potential popular items, lacking a clear distinction between user interest in item property and user conformity to item popularity. This limitation hampers their generalization ability in scenarios with unseen popularity distribution shifts.

Conversely, disentanglement-based competitors (DICE, Margin, InvCF, and DCCL), while attempting to separate intrinsic property factors from popularity factors, neglect temporal dynamics in popularity, leading to suboptimal generalization during temporal distribution shifts. TPAB successfully integrates temporal popularity with disentanglement principles, based on temporal-aware embeddings, popularity coarsening, and bootstrapping.

## 4.3 Ablation Studies

To answer RQ2, we conduct a detailed exploration of the impact of the use of temporal popularity embeddings ($p_i^t$), popularity coarsening (Eq. (4)) and popularity bootstrapping (Eq. (5)). Specifically, we compare TPAB with its three variants (TPAB-T, TPAB-C, TPAB-B), as outlined in Section 4.1.2, evaluating both recommendation accuracy and variance in risks across different time stages.

*4.3.1 Accuracy.* Figure 3 shows that TPAB consistently and significantly outperforms TPAB-T, TPAB-C, and TPAB-B, indicating that the use of temporal popularity, popularity coarsening, and popularity bootstrapping *each* contribute to enhancing the generalization ability of TPAB. Here is a possible explanation for each result: Coarsening helps reduce TPAB's sensitivity to minor fluctuations in popularity. The bootstrapping loss promotes the invariance of property embeddings to popularity embeddings, as demonstrated by the Theorem 3.1 on bootstrapping loss. Moreover, using temporal popularity instead of global/static popularity allows the coarsening and bootstrapping to consider temporal dynamics in popularity.

*4.3.2 Variance.* Note that [15] demonstrated that equalizing training risks across different environments enhances generalization. Guided by these insights, we assess the variance in recommendation scores across different time stages for TPAB and its two variants,

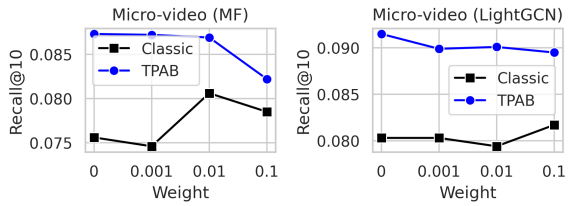



**Figure 5: Comparison of disentanglement methods with varying weights of the geometric separation. While Classic occasionally benefits from this technique, it harms the expressiveness of embeddings in TPAB.**

**Table 3: Efficiency comparison of the average running time (seconds) per epoch for TPAB and seven competitors.**

| Models | Vanilla | PDA | PDRO | DICE | Margin | DCCL | InvCF | TPAB |
|---|---|---|---|---|---|---|---|---|
| Runtime (s) | 10.14 | 10.85 | 20.70 | 15.62 | 15.73 | 15.84 | 15.84 | 13.06 |

TPAB-C and TPAB-B, in Figure 4 (refer to the legend for specific numbers). Lower variance indicates greater equality. The results consistently show that, after convergence, TPAB has the lowest variance on both datasets. This aligns with the observed trend in the recommendation accuracy, further supporting the effectiveness of popularity coarsening and bootstrapping in enhancing generalization ability. We attribute the smaller variances to the fact that both techniques encourage items across different time stages to share the same popularity embeddings.

## 4.4 Analysis on Geometric Separation

As noted in Section 1, in addition to the ERM loss and the invariance learning loss, many disentanglement methods utilize geometric separation for orthogonality disentanglement between two types of embeddings for users ($I_{\text{prop}}$ and $I_{\text{pop}}$); and items ($U_{\text{int}}$ and $U_{\text{conf}}$) [35, 39]. Specifically, it maximizes the distance metric (e.g., L1, L2, and distance correlation [24, 25]) between these two types of embeddings. To answer RQ3, we investigate the impact of such geometric separation to the expressiveness of embeddings of TPAB and Classic, the latter representing the conventional disentanglement approach using only $\mathcal{L}_{\text{bpr}}$ with global/static popularity $p_i$. We test varying regularization weights [0, 0.001, 0.01, 0.1] and use distance correlation as the metric [35, 39]. The results are in Figure 5.

The results show that while Classic occasionally benefits from the separation effect, TPAB *degrades* with its use. Note that geometric separation acts as "regularization," which, although potentially beneficial for disentanglement when applied judiciously, can limit the expressiveness of embeddings in any disentanglement method, including Classic. However, for TPAB, it constrains the expressiveness of embeddings in most scenarios, suggesting that geometric separation is not advisable for TPAB. Moreover, the performance of TPAB without geometric separation already exceeds that of Classic even when enhanced with separation. This suggests that TPAB effectively disentangles invariant property embeddings and temporal popularity embeddings through temporal-aware popularity coarsening and bootstrapping.

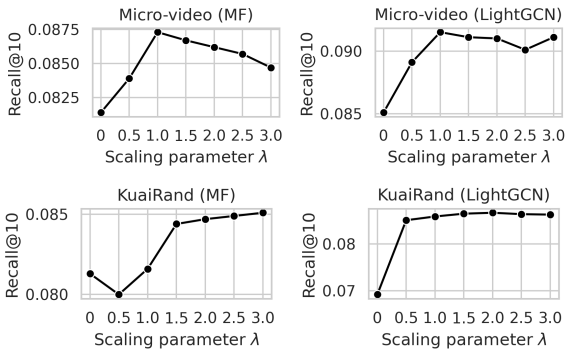



**Figure 6: Performance of TPAB with varying $\lambda$. Using the bootstrapping loss consistently enhances generalization. Initially, performance sharply increases with increasing $\lambda$ values, then stabilizes or slightly declines thereafter.**

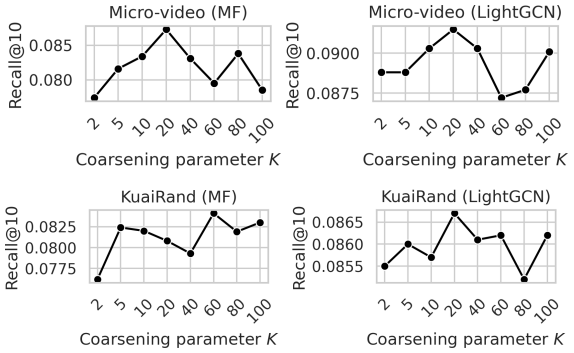



**Figure 7: Performance of TPAB with varying $K$. Setting $K$ either too high or too low leads to suboptimal performance due to overly weak or high coarsening effects, respectively.**

## 4.5 Hyperparameter Analysis

To answer RQ4, we examine the impact of hyperparameters $\lambda$ (the scaling parameter for the bootstrapping loss $\mathcal{L}_{\text{boot}}$) and $K$ (the number of buckets for popularity coarsening) on the recommendation performance of TPAB.

*4.5.1 Bootstrapping scaling parameter $\lambda$.* We show the recommendation accuracy of TPAB with varying $\lambda$ values [0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0] in Figure 6. Note that when $\lambda = 0$, TPAB does not use $\mathcal{L}_{\text{boot}}$. The results indicate that incorporating $\mathcal{L}_{\text{boot}}$ alongside $\mathcal{L}_{\text{bpr}}$ consistently leads to better generalization in all cases. Specifically, on Micro-video, optimal performance occurs at $\lambda = 1.0$, with a slight decline thereafter, possibly due to excessive bootstrapping (randomness). Similarly, on the KuaiRand dataset, accuracy increases sharply at $\lambda = 1.5$ and $\lambda = 0.5$ for MF and LightGCN-based models, respectively, and then stabilizes.

*4.5.2 Coarsening parameter $K$.* We show the performance of TPAB with varying $K$ values [2, 5, 10, 20, 40, 60, 80, 100] in Figure 7. The results show that values around $K = 20$ consistently performs best in most cases, especially compared to when k is too small or too large. This suggests that setting $K$ too high results in a weak coarsening effect, while setting it too low, such as $K = 2$, results in overly strong coarsening with only two popularity levels.

Thus, selecting the appropriate level of coarsening requires careful consideration.

We acknowledge some erratic behavior in the intermediate results. However, it is notable that $K = 20$ significantly outperforms cases without coarsening (i.e., TPAB-C). Preliminary experiments also show that $K > 100$ results in a performance drop, highlighting a trend of diminishing performance as $K$ becomes excessively large.

An exception is observed in KuaiRand, particularly in MF, likely due to its lower long-tailness (head-tail ratio of 1.57 compared to <1.0 in other datasets) and higher maximum temporal popularity (e.g., $max(p_i^t) = 2678$), indicating extreme hubs. The current exponential partitioning may be less effective here, so $K$ should be carefully selected based on dataset characteristics.

## 4.6 Time-efficiency Results

To answer RQ5, we show the average running time per epoch (seconds) for TPAB and seven competitors using MF as backbones in Table 3. Disentanglement-based methods (i.e., DICE, MARGIN, INVCF, DCCL, TPAB) are slightly slower than the Vanilla method. PDRO is the slowest due to its group-based distributionally robust optimization. As shown in the complexity analysis in Section 3.5, TPAB's $\mathcal{L}_{\text{boot}}$ shares the same structure as $\mathcal{L}_{\text{bpr}}$, differing only in the randomly sampled popularity $p'$. Without more complex invariance learning or additional geometric separation, TPAB does not add significant computation to Vanilla.

## 5 Related Works

**Temporal popularity distribution shifts.** In real-world recommendation scenarios, changing popularity distributions over time can lead to variations in training and test set distributions, impacting model performance. While classic debiasing techniques, such as reweighting [11, 28], and regularization [5, 40], have been proposed, recent works specifically focus on addressing out-of-distribution generalization ability for future online service through the concept of Distributionally Robust Optimization (DRO) [3, 17, 21]. For instance, [30] uses DRO for better distribution adaptation by estimating the nominal distribution from input data and optimize the model within a robust radius. Another study [29] applies group-DRO, emphasizing the optimization of the worst user group performance, addressing both fairness and the recommendation quality.

While these methods address out-of-distribution generalization, they often neglect the temporal dynamics of popularity during method design. To address this, a recent work [37] proposes a new DRO objective that considers temporal popularity. It also utilizes recalibration of recommendation scores based on predicted next-time popularity following [36]. However, challenges persist as these methods may indiscriminately prioritize potential popular items due to entangled user/item embeddings, lacking a clear separation between intrinsic property factors and popularity factors.

**Disentangled representation learning.** Disentanglement methods aim to separate intrinsic property and popularity factors, offering effective solutions for generalization during popularity distribution shifts. Typically, they pursue two types of disentanglement objectives: (O1) invariance learning and (O2) geometric separation. For (O1), [39] uses popularity-aware negative item sampling, [4] employs co-training of unbiased and biased models, [34] quantifies popularity bias and uses these biases as margins for their loss function, and [35] uses techniques like representation augmentation to learn invariant embeddings. As for (O2), different geometric separation techniques are employed to maximize the discrepancy (e.g., L1 [39], L2 [39], Pearson correlation coefficient [4], and distance correlation [27, 35, 39]) between the two types of embeddings. However, all these methods lack consideration for temporal dynamics in popularity. Our approach, TPAB, effectively integrates temporal popularity with disentanglement using novel techniques like temporal embeddings, popularity coarsening, and bootstrapping.

**Sequential recommender.** We also discuss the distinctions between TPAB and sequential recommender systems in Appendix E.

## 6 Conclusion

In this paper, we observe that while existing disentanglement methods have advanced in separating intrinsic property and popularity to enhance generalization, they often overlook the temporal nature of popularity shifts. To address this gap, we propose TPAB, a novel disentanglement method that integrates temporal popularity dynamics, based on temporal-aware embeddings, popularity coarsening, and bootstrapping. We also offer a theoretical analysis showing that the bootstrapping loss promotes the invariance of property embeddings to temporal popularity changes. During inference, TPAB utilizes both derived next-time popularity embeddings and property embeddings. Extensive experiments on real-world temporal datasets validate the strong generalizability of TPAB during temporal popularity distribution shifts.

## Acknowledgments

This work is supported by NSF (2416070 ), NIFA (2020-67021-32799), and IBM-Illinois Discovery Accelerator Institute. The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

## References

[1] Yikun Ban, Yuchen Yan, Arindam Banerjee, and Jingrui He. 2021. Ee-net: Exploitation-exploration neural networks in contextual bandits. *arXiv preprint arXiv:2110.03177* (2021).
[2] Albert-László Barabási. 2003. Linked: The new science of networks.
[3] Aharon Ben-Tal, Dick Den Hertog, Anja De Waegenaere, Bertrand Melenberg, and Gijs Rennen. 2013. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science* 59, 2 (2013), 341–357.
[4] Zhihong Chen, Jiawei Wu, Chenliang Li, Jingxu Chen, Rong Xiao, and Binqiang Zhao. 2022. Co-training disentangled domain adaptation network for leveraging popularity bias in recommenders. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 60–69.
[5] Zhihong Chen, Rong Xiao, Chenliang Li, Gangfeng Ye, Haochuan Sun, and Hongbo Deng. 2020. ESAM: Discriminative Domain Adaptation with Non-Displayed Items to Improve Long-Tail Performance. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2020). https://api.semanticscholar.org/CorpusID:218763654
[6] Sung Min Cho, Eunhyeok Park, and Sungjoo Yoo. 2020. MEANTIME: Mixture of attention mechanisms with multi-temporal embeddings for sequential recommendation. In *Proceedings of the 14th ACM Conference on recommender systems*. 515–520.
[7] Evert Gummesson. 2007. Case study research and network theory: birds of a feather. *Qualitative Research in Organizations and Management: An International Journal* 2, 3 (2007), 226–248.
[8] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for

recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 639–648.
[9] Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. 2018. Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230* (2018).
[10] Hawoong Jeong, Zoltan Néda, and Albert-László Barabási. 2003. Measuring preferential attachment in evolving networks. *Europhysics letters* 61, 4 (2003), 567.
[11] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2016. Unbiased Learning-to-Rank with Biased Feedback. *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (2016). https://api.semanticscholar.org/CorpusID:300418
[12] Diederik P Kingma and Jimmy Ba. 2011. Adam: A method for stochastic optimization. *Computing in science & engineering* 13 (2011), 22–30.
[13] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[14] Walid Krichene and Steffen Rendle. 2020. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 1748–1757.
[15] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. 2021. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*. PMLR, 5815–5826.
[16] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time interval aware self-attention for sequential recommendation. In *Proceedings of the 13th international conference on web search and data mining*. 322–330.
[17] Fengming Lin, Xiaolei Fang, and Zheming Gao. 2022. Distributionally robust optimization: A review on theory and applications. *Numerical Algebra, Control and Optimization* 12, 1 (2022), 159–212.
[18] Yunzhe Qi, Yikun Ban, and Jingrui He. 2023. Graph neural bandits. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1920–1931.
[19] Ahmed Rashed, Shereen Elsayed, and Lars Schmidt-Thieme. 2022. Context and attribute-aware sequential recommendation via cross-attention. In *Proceedings of the 16th ACM Conference on Recommender Systems*. 71–80.
[20] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).
[21] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2019. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731* (2019).
[22] Paras Sheth, Ruocheng Guo, Kaize Ding, Lu Cheng, K Selçuk Candan, and Huan Liu. 2022. Causal disentanglement with network information for debiased recommendations. In *International Conference on Similarity Search and Applications*. Springer, 265–273.
[23] Harald Steck. 2011. Item popularity and recommendation accuracy. In *Proceedings of the fifth ACM conference on Recommender systems*. 125–132.
[24] Gábor J Székely and Maria L Rizzo. 2009. Brownian distance covariance. *The annals of applied statistics* (2009), 1236–1265.
[25] Gábor J Székely, Maria L Rizzo, and Nail K Bakirov. 2007. Measuring and testing dependence by correlation of distances. (2007).
[26] Viet Anh Tran, Guillaume Salha-Galvan, Bruno Sguerra, and Romain Hennequin. 2023. Attention mixtures for time-aware sequential recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1821–1826.
[27] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. 2020. Disentangled Graph Collaborative Filtering. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2020). https://api.semanticscholar.org/CorpusID:220347145
[28] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. 2019. Doubly Robust Joint Learning for Recommendation on Data Missing Not at Random. In *International Conference on Machine Learning*. https://api.semanticscholar.org/CorpusID:174800959
[29] Hongyi Wen, Xinyang Yi, Tiansheng Yao, Jiaxi Tang, Lichan Hong, and Ed H Chi. 2022. Distributionally-robust Recommendations for Improving Worst-case User Experience. In *Proceedings of the ACM Web Conference 2022*. 3606–3610.
[30] Zhengyi Yang, Xiangnan He, Jizhi Zhang, Jiancan Wu, Xin Xin, Jiawei Chen, and Xiang Wang. 2023. A generic learning framework for sequential recommendation with distribution shifts. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 331–340.
[31] Hyunsik Yoo, Yeon-Chang Lee, Kijung Shin, and Sang-Wook Kim. 2023. Disentangling Degree-related Biases and Interest for Out-of-Distribution Generalized Directed Network Embedding. In *Proceedings of the ACM Web Conference 2023*. 231–239.
[32] Hyunsik Yoo, Zhichen Zeng, Jian Kang, Ruizhong Qiu, David Zhou, Zhining Liu, Fei Wang, Charlie Xu, Eunice Chan, and Hanghang Tong. 2024. Ensuring user-side fairness in dynamic recommender systems. In *Proceedings of the ACM on Web Conference 2024*. 3667–3678.
[33] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*. 1294–1303.
[34] An Zhang, Wenchang Ma, Xiang Wang, and Tat-Seng Chua. 2022. Incorporating bias-aware margins into contrastive loss for collaborative filtering. *Advances in Neural Information Processing Systems* 35 (2022), 7866–7878.
[35] An Zhang, Jingnan Zheng, Xiang Wang, Yancheng Yuan, and Tat-Seng Chua. 2023. Invariant Collaborative Filtering to Popularity Distribution Shift. In *Proceedings of the ACM Web Conference 2023*. 1240–1251.
[36] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guohui Ling, and Yongdong Zhang. 2021. Causal intervention for leveraging popularity bias in recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 11–20.
[37] Jujia Zhao, Wenjie Wang, Xinyu Lin, Leigang Qu, Jizhi Zhang, and Tat-Seng Chua. 2023. Popularity-aware Distributionally Robust Optimization for Recommendation System. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. 4967–4973.
[38] Weiqi Zhao, Dian Tang, Xin Chen, Dawei Lv, Daoli Ou, Biao Li, Peng Jiang, and Kun Gai. 2023. Disentangled causal embedding with contrastive learning for recommender system. In *Companion Proceedings of the ACM Web Conference 2023*. 406–410.
[39] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin. 2021. Disentangling user interest and conformity for recommendation with causal embedding. In *Proceedings of the Web Conference 2021*. 2980–2991.
[40] Ziwei Zhu, Yun He, Xing Zhao, Yin Zhang, Jianling Wang, and James Caverlee. 2021. Popularity-Opportunity Bias in Collaborative Filtering. *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (2021). https://api.semanticscholar.org/CorpusID:232073630

# A Proof of Theorem on Bootstrapping

Proof of Theorem 3.1. Recall that $p$ represents coarsened popularity, indicating $K$ popularity buckets. The user chooses either $i_1$ and $i_2$ with probability $\sigma(s_1(p_1) - s_2(p_2))$ and $\sigma(s_2(p_2) - s_1(p_1))$, respectively. Lastly, let $U$ be the uniform sampling distribution of possible $p$. Under this setting,

$$\mathcal{L}_{\text{boot}} = \mathop{\mathbb{E}}_{p_1,p_2\sim P,\, p_1',p_2'\sim U} \Big[ (-\sigma(s_1(p_1) - s_2(p_2)) \log \sigma(\hat{s}_1(p_1') - \hat{s}_2(p_2'))) - (\sigma(s_2(p_2) - s_1(p_1)) \log \sigma(\hat{s}_2(p_2') - \hat{s}_1(p_1'))) \Big], \tag{14}$$

where $p_1$ and $p_2$ are true popularity and $p_1'$ and $p_2'$ are **uniformly and randomly** sampled popularity due to the bootstrapping strategy; and $\sigma(\cdot)$ is a sigmoid function. Since $(p_1', p_2')$ has $K^2$ combinations, we further have the following:

$$\mathcal{L}_{\text{boot}} = -\frac{1}{K^2} \sum_{p_1',p_2'} \Bigg( \mathop{\mathbb{E}}_{p_1,p_2\sim P} [\sigma(s_1(p_1) - s_2(p_2))] \log \sigma(\hat{s}_1^{\text{prop}} - \hat{s}_2^{\text{prop}} + \hat{s}_{p_1'}^{\text{pop}} - \hat{s}_{p_2'}^{\text{pop}}) + \mathop{\mathbb{E}}_{p_1,p_2\sim P} [\sigma(s_2(p_2) - s_1(p_1))] \log \sigma(\hat{s}_2^{\text{prop}} - \hat{s}_1^{\text{prop}} + \hat{s}_{p_2'}^{\text{pop}} - \hat{s}_{p_1'}^{\text{pop}}) \Bigg). \tag{15}$$

The optimal solution of the system in the form of $-a \log x - (1 - a) \log(1 - x)$ is $x = a$. Therefore, the optimal solution of the above optimization is:

$$\sigma(\hat{s}_1^{\text{prop}} - \hat{s}_2^{\text{prop}} + \hat{s}_{p_1'}^{\text{pop}} - \hat{s}_{p_2'}^{\text{pop}}) = \mathop{\mathbb{E}}_{p_1,p_2\sim P} [\sigma(s_1(p_1) - s_2(p_2))], \quad \forall p_1', p_2'. \tag{16}$$

Solving the equations above gives the following optimal solution:

$$\hat{s}_1^{\text{prop}} - \hat{s}_2^{\text{prop}} = \log \frac{\mathbb{E}_{p_1,p_2}\, \sigma(s_1(p_1) - s_2(p_2))}{\mathbb{E}_{p_1,p_2}\, \sigma(s_2(p_2) - s_1(p_1))}; \quad \hat{s}_{p_1'}^{\text{pop}} = \hat{s}_{p_2'}^{\text{pop}}, \quad \forall p_1', p_2'. \tag{17}$$

**Table 5: Performance comparison of TPAB and seven competitors using SimGCL backbone on Micro-video dataset.**

| | | R@10 | N@10 | R@20 | N@20 |
|---|---|---|---|---|---|
| SimGCL | **Vanilla** | 0.0863 | 0.0598 | 0.1261 | 0.0727 |
| | **PDA** | 0.0875 | 0.0604 | 0.126 | 0.0729 |
| | **PDRO** | 0.0863 | 0.0598 | 0.1253 | 0.0723 |
| | **DICE** | 0.0846 | 0.0585 | 0.1265 | 0.0719 |
| | **Margin** | 0.0788 | 0.054 | 0.1142 | 0.0653 |
| | **InvCF** | 0.0869 | 0.0597 | 0.1288 | 0.0731 |
| | **DCCL** | 0.0888 | 0.0604 | 0.1303 | 0.0740 |
| | **TPAB** | **0.0902** | **0.0616** | **0.1315** | **0.0751** |

**Table 4: Performance comparison of TPAB and seven competitors on Yelp2022 and Amazon-book datasets.**

| | | R@10 | N@10 | R@20 | N@20 |
|---|---|---|---|---|---|
| Yelp | **Vanilla** | 0.0193 | 0.0151 | 0.0337 | 0.0197 |
| | **PDA** | 0.0198 | 0.0153 | 0.0342 | 0.0199 |
| | **PDRO** | 0.016 | 0.0123 | 0.0281 | 0.0161 |
| | **DICE** | 0.0208 | 0.0167 | 0.0351 | 0.0211 |
| | **Margin** | 0.0123 | 0.01 | 0.0228 | 0.0132 |
| | **InvCF** | 0.0194 | 0.0149 | 0.0335 | 0.0195 |
| | **DCCL** | 0.0187 | 0.0148 | 0.0326 | 0.0192 |
| | **TPAB** | **0.0232** | **0.0179** | **0.04** | **0.0232** |
| Amazon | **Vanilla** | 0.0239 | 0.0156 | 0.04 | 0.0207 |
| | **PDA** | 0.0232 | 0.0151 | 0.0389 | 0.0201 |
| | **PDRO** | 0.0202 | 0.0129 | 0.0341 | 0.0173 |
| | **DICE** | 0.025 | 0.0164 | 0.0414 | 0.0216 |
| | **Margin** | 0.0148 | 0.0091 | 0.0235 | 0.0118 |
| | **InvCF** | 0.0232 | 0.0152 | 0.0382 | 0.0199 |
| | **DCCL** | 0.0239 | 0.0156 | 0.0399 | 0.0207 |
| | **TPAB** | **0.0317** | **0.0213** | **0.0519** | **0.0275** |

□

## B Longer/larger Datasets

We have conducted comparative experiments on two larger and longer datasets that are widely used: Yelp2022 and Amazon-book.

- Yelp2022[6]: It contains approximately 2M ratings from 80K users to 75K businesses (e.g., restaurants). Time span is 10 years from 2012-01-01 to 2022-01-19. Note that we sampled the most recent 10 year for this dataset.
- Amazon-book[7]: It contains around 2M ratings from 65K to 88K book products. Time span is 17 years from 1997-01-31 to 2014-07-23)

These datasets validate the model's generalizability over longer periods, with test data spanning around 1.7 and 1 year, respectively. The results, presented in Table 4 consistently show that TPAB outperforms all seven competitors on both datasets, showing that TPAB is also generalizable over longer time periods.

## C Additional Backbone

We compare TPAB with seven competitors using simGCL [33], a recent and advanced recommendation backbone, on the Micro-video dataset. The results show that TPAB consistently outperforms all competitors.

## D Implementation Details

For PDA and PDRO, we select the popularity weighting parameter $\gamma$ from [0, 2, 4, 6, 8]. We also adhere to the suggested hyperparameters for PDRO. For DICE, Margin, InvCF, and DCCL, each method utilizes a scaling parameter for its respective invariance learning loss. We vary this parameter across the values [0, 0.01, 0.05, 0.1, 0.5, 1]. Additionally, we adjust the scaling parameter for geometric separation from the range [0, 0.001, 0.01, 0.1, 1.0].

[6]https://www.yelp.com/dataset.
[7]https://cseweb.ucsd.edu/ jmcauley/datasets/amazon/links.html.

## E Discussion on Sequential Models

We clarify the distinctions between TPAB and sequential recommender systems.

First, TPAB differs fundamentally in problem setting. Sequential recommendation systems typically use a user's item sequence as input, focusing on modeling sequential patterns and dependencies. In contrast, TPAB is a general two-tower recommender that explicitly learns separate user and item embeddings, where modeling item sequences for each user is unnecessary.

Second, while item sequences exist for each user, applying TPAB to a sequential setting is not straightforward and requires additional considerations. Sequential models generally learn entangled embeddings for items and user-item sequences, where item properties and popularity are intertwined. Such models cannot disentangle static/invariant item properties from dynamic/variant popularity, which is central to TPAB's design rationale.

A potential approach to learning disentangled item embeddings in a sequential setting might involve learning separate property and popularity embeddings, either through a separate attention network or within a unified attention framework. However, this has several limitations: (1) Most importantly, item property embeddings would implicitly capture both sequential patterns and popularity shifts, which contradicts TPAB's core idea of maintaining static/invariant property embeddings independent of popularity changes. (2) While temporal item popularity embeddings could be learned, no user conformity embedding is involved, preventing personalization for users. For example, if users $u_1$ and $u_2$ have the same popularity sequence, they would have the same predicted popularity score, whereas TPAB can assign different scores based on user-side conformity embeddings.

Third, some existing sequential methods [6, 16, 19, 26] incorporate the time context of user-item interactions or the time interval between items in the sequence. However, they do not disentangle property and popularity embeddings and thus do not explicitly capture temporal popularity. Even if they attempt this, they tend to preserve sequential patterns of popularity rather than capturing distinct popularity at different time stages.