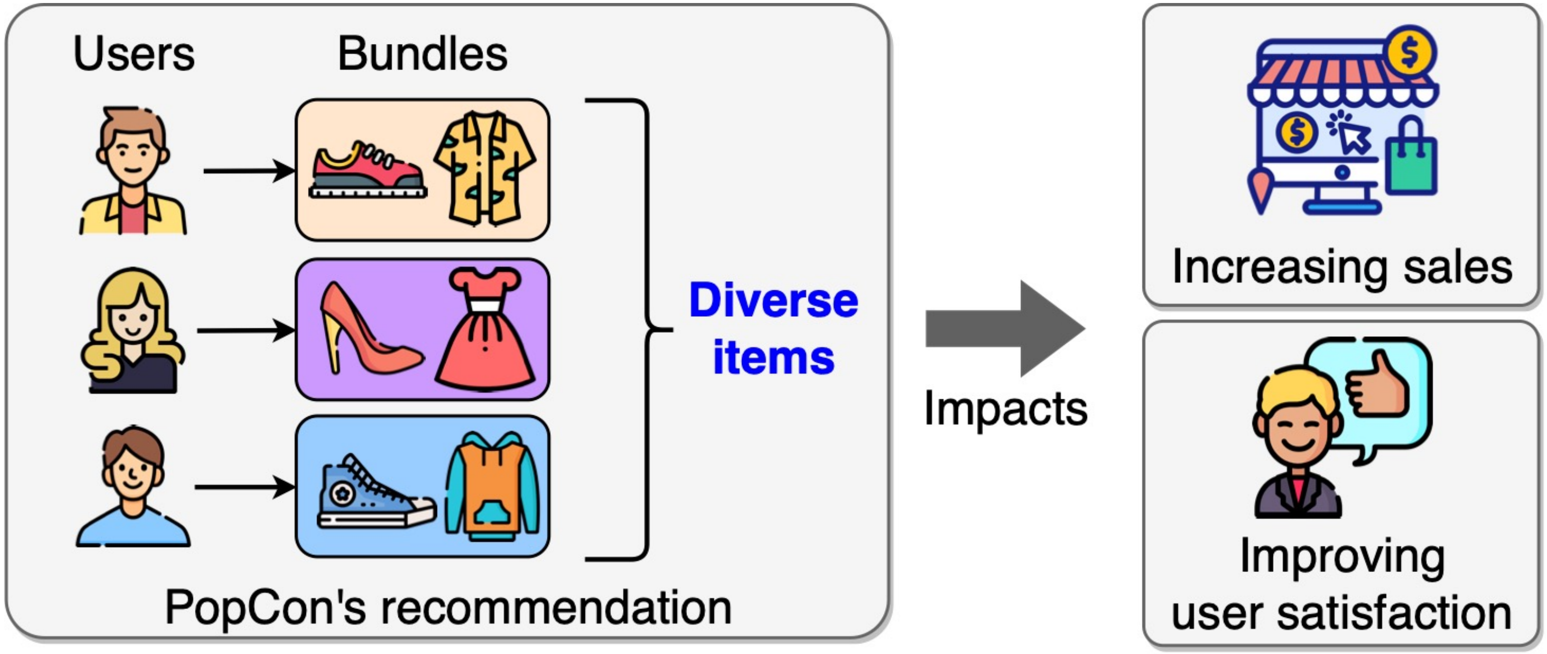
Aggregately Diversified Bundle Recommendation via Popularity Debiasing and Configuration-aware Reranking

Hyunsik Jeon, ***Jongjin Kim***, ***Jaeri Lee***,
Jong-eun Lee, *and* ***U Kang***
Seoul National University

PAKDD 2023

Research Overview

- How can we expose **diverse items** across all users in **bundle recommendations**?
- **PopCon** (our method) diversifies the item exposure when recommending bundles!



Outline

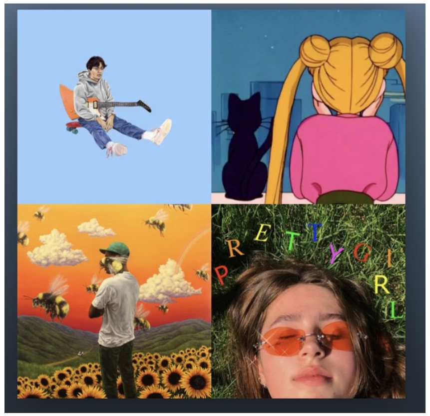
- **<u>Introduction</u>**
- Proposed Method
- Experiments
- Conclusion

Recommender System

- It provides **<u>personalized items</u>** for each user
  - It enhances users' experience and increase sales revenue
- Applications
  - Essential for various online services

Product Bundling

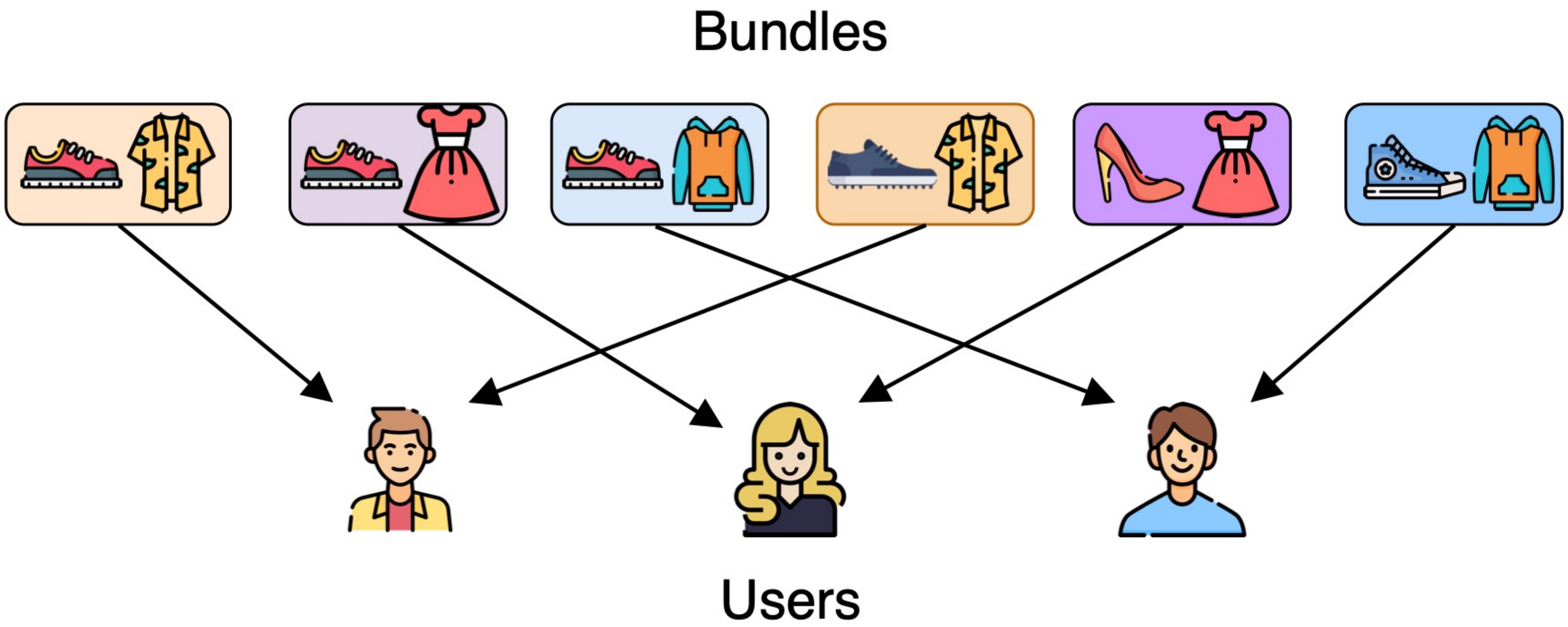
- It is a **<u>prevalent strategy</u>** in industries
  - One-stop convenience for customers
  - Increased exposure to lesser-known products
  - Cost-efficient offerings to customers



Examples of product bundling

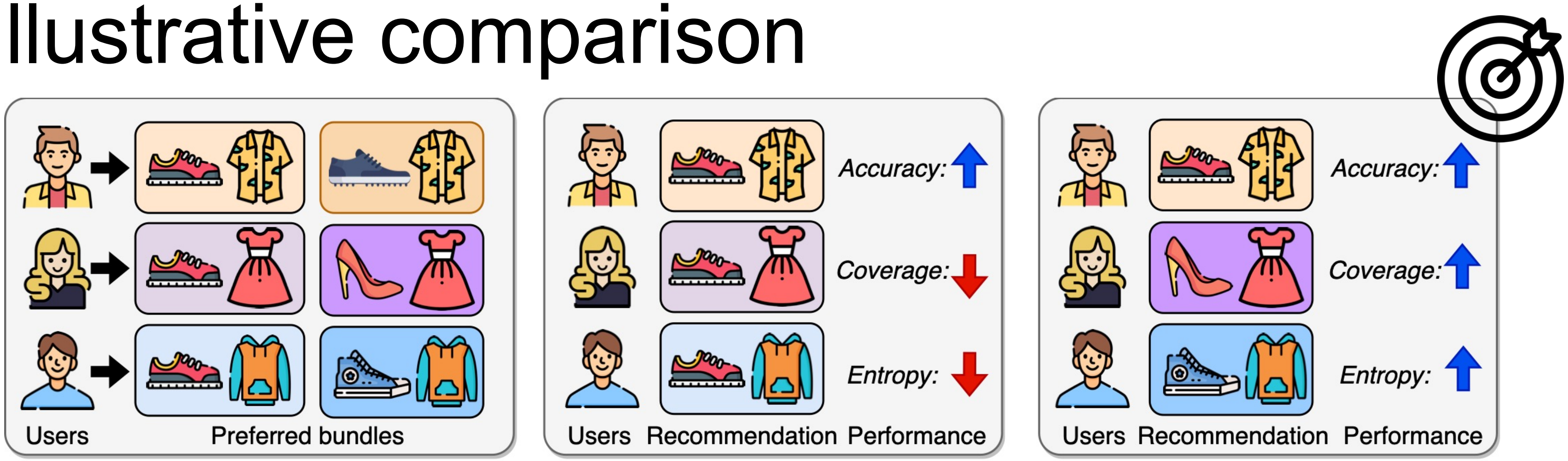
Bundle Recommendation

- It aims to recommend **bundles** instead of individual items
- It has become an **important technique** in industries



Aggregate Diversity

- The degree of **<u>fair exposure of items</u>**
  - It is measured by coverage and entropy
- Previous works on bundle recommendation
  - They have focused only on accuracy
- Illustrative comparison



(a) Ground-truth preferences of users

(b) Traditional bundle recommendation

(c) Aggregately diversified bundle recommendation

Problem Definition

Aggregately diversified bundle recommendation

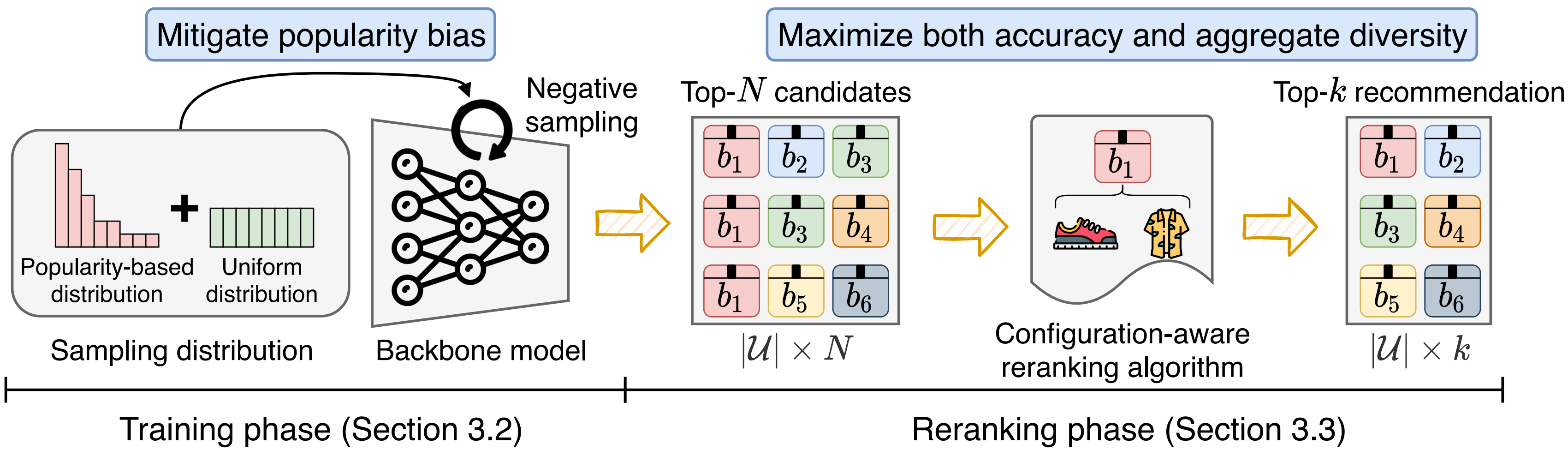
- **Given**
  - A user-bundle interaction matrix $\mathbf{X} \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{B}|}$
  - A user-item interaction matrix $\mathbf{Y} \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{I}|}$
  - A bundle-item affiliation matrix $\mathbf{Z} \in \mathbb{R}^{|\mathcal{B}| \times |\mathcal{I}|}$
    - $\mathcal{U}$, $\mathcal{B}$, $\mathcal{I}$: sets of users, bundles, and items, resp.
- **Recommend**
  - k bundles to each user $u \in \mathcal{U}$
- **Such that**
  - Users are satisfied with the recommended bundles
  - The recommended items are aggregately diverse

Outline

- Introduction
- **<u>Proposed Method</u>**
- Experiments
- Conclusion

Overview

- We propose **PopCon** to address the problem
  - Two phases: training and reranking
    - **Training phase.** Mitigates popularity bias when training a model
    - **Reranking phase.** Maximizes both accuracy and aggregate diversity simultaneously



Training Phase (1/3)

- **Goal.** Train a model $f(u,b)$
  - f estimates how much user u likes bundle b
- **Objective function.** BPR (Bayesian Personalized Ranking) loss:

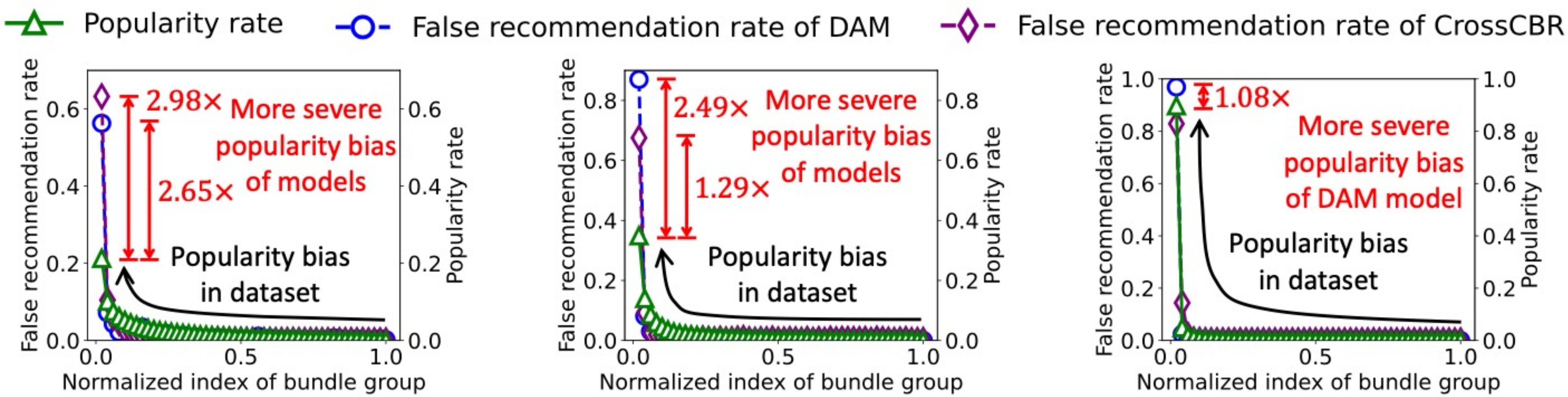
$$\sum_{(u,b,b')\in D} -\ln\sigma\left(f(u,b)-f(u,b')\right)$$

u: target user
b: positive bundle
b': negative bundle
D: set of triplets
$\sigma(\cdot)$: sigmoid func.

  - It maximizes the difference between positive (interacted) target b and negative (non-interacted) target b' for each user u
  - Previous works sample b' from the uniform distribution
    - $p(b')=\frac{1}{|\mathcal{B}|}$ where $\mathcal{B}$ is the set of bundles

Training Phase (2/3)

- **Observation.** Models easily overfit to some popular bundles
  - It makes it difficult to achieve high aggregate diversity when using the model $f(\cdot)$ in the reranking phase



(a) NetEase (b) Youshu (c) Steam

Training Phase (3/3)

- **Challenge 1.** How can we mitigate the popularity bias of the model?
- **Idea 1.** Popularity-based negative sampling
  - Objective function: BPR loss (same as previous works)
  - Sampling distribution of negative bundle b'

$$p(b') = \underbrace{\alpha \frac{freq(b')}{\sum_{j \in \mathcal{B}} freq(j)}}_{\text{popularity-based distribution}} + \underbrace{(1-\alpha)\frac{1}{|\mathcal{B}|}}_{\text{uniform distribution}}$$

$\alpha \in [0,1]$: balance coefficient
$freq(j)$: number of bundle j's interactions
$\mathcal{B}$: set of bundles

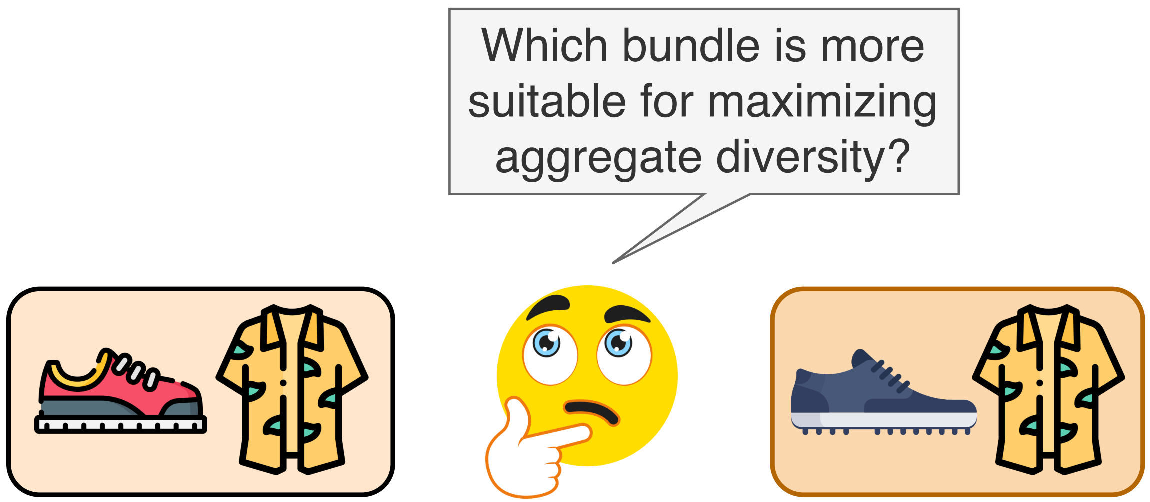
    - Sampling probability is large if a bundle b' is popular
    - It mitigates the popularity bias

Reranking Phase (1/6)

- **Goal.** Maximize both accuracy and aggregate diversity of result of $f(\cdot)$
- **Algorithm.** Recommend k bundles to each user
  - 1. For each user $u \in \mathcal{U}$, we select top-N $(k < N < |\mathcal{B}|)$ candidate bundles that maximize $f(u, b)$
  - 2. (Repeat k times.) For each user $u \in \mathcal{U}$, we select a bundle that maximizes $g(u, b)$
    - $g(u, b) \in \mathbb{R}$: a scoring function that measures accuracy and aggregate diversity simultaneously

Reranking Phase (2/6)

- We have two challenges to design $g(\cdot)$
- **Challenge 2-1.** How to measure the aggregate diversity considering bundles' configuration?



Candidate bundle b_1

Candidate bundle b_2

Reranking Phase (3/6)

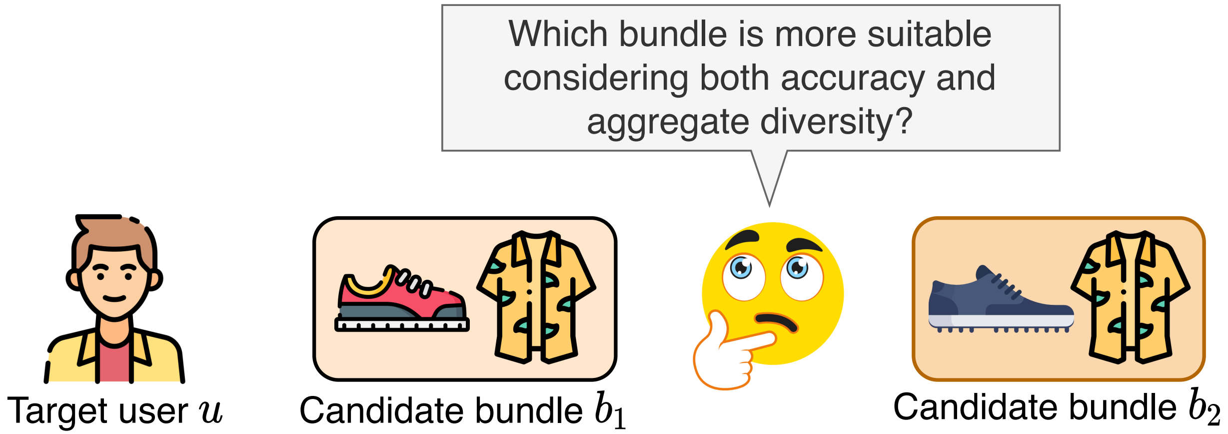
- **Idea 2-1.** Configuration-aware diversity gain

$$DivGain(b, \hat{\mathbf{R}}(k)) = \frac{1}{2} CovGain(b, \hat{\mathbf{R}}(k)) + \frac{1}{2} EntGain(b, \hat{\mathbf{R}}(k))$$

  - It measures the gains of both coverage and entropy when adding bundle b to the current recommendation result $\widehat{\mathbf{R}}(k)$
  - $CovGain\left(b, \widehat{\mathbf{R}}(k)\right) \in [0,1]$: the gain of item coverage
    - It considers the appearance of new items
  - $EntGain\left(b, \widehat{\mathbf{R}}(k)\right) \in [-1,1]$: the gain of item entropy
    - It considers the fair appearance of items

Reranking Phase (4/6)

- We have two challenges to design $g(\cdot)$
- **Challenge 2-2.** How to handle the opposite two criteria: accuracy and aggregate diversity?
  - Accuracy and aggregate diversity are opposite in most cases



$f(u, b_1) = 0.9$ $\quad$ $f(u, b_2) = 0.4$

$DivGain(b_1, \hat{\mathbf{R}}(k)) = 0.1$ $\quad$ $DivGain(b_2, \hat{\mathbf{R}}(k)) = 0.5$

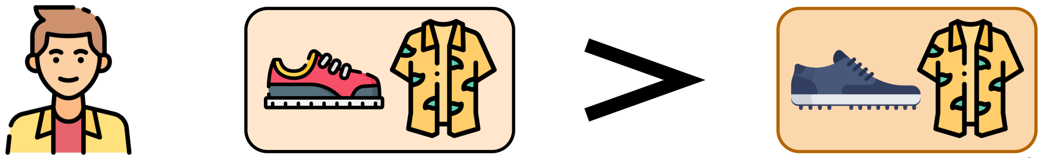
Reranking Phase (5/6)

- **Naïve solution.** Weighted sum of two scores

$$g(u, b, \hat{\mathbf{R}}(k)) = (1 - \beta)\sigma(\hat{x}_{ub}) + \beta DivGain(b, \hat{\mathbf{R}}(k))$$

$\hat{x}_{ub} \triangleq f(u, b)$

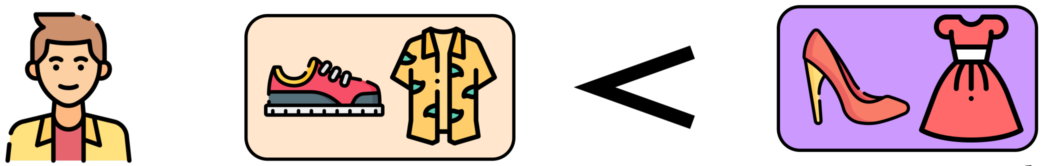
$\beta \in [0,1]$: balance coefficient



Target user u — Candidate bundle b_1 $>$ Candidate bundle b_2

$f(u, b_1) = 0.9$ — $f(u, b_2) = 0.4$

$DivGain(b_1, \hat{\mathbf{R}}(k)) = 0.1$ — $DivGain(b_2, \hat{\mathbf{R}}(k)) = 0.5$



Target user u — Candidate bundle b_1 $<$ Candidate bundle b_3

$f(u, b_1) = 0.9$ — $f(u, b_3) = 0.1$

$DivGain(b_1, \hat{\mathbf{R}}(k)) = 0.1$ — $DivGain(b_3, \hat{\mathbf{R}}(k)) = 1.0$

User u will not be satisfied with this recommendation

Consider examples when $\beta = 0.5$

Reranking Phase (6/6)

- **Idea 2-2.** Accuracy-prioritized coupling
  - We propose an **accuracy priority property** that reduces the influence of $DivGain(\cdot)$ as $\hat{x}_{ub}$ increases:

$$\text{If } \sigma(\hat{x}_{ub}) > \sigma(\hat{x}_{ub'}), \text{then } \frac{\partial g(u, b, \hat{\mathbf{R}}(k))}{\partial DivGain(b, \hat{\mathbf{R}}(k))} < \frac{\partial g(u, b', \hat{\mathbf{R}}(k))}{\partial DivGain(b', \hat{\mathbf{R}}(k))}.$$

$\hat{x}_{ub} \triangleq f(u, b)$

$\sigma(\hat{x}_{ub}) \in [0, 1]$

  - It ensures that bundles that users like a lot are recommended regardless of $DivGain(\cdot)$ to satisfy them
  - Thus, our scoring function:

$$g(u, b, \hat{\mathbf{R}}(k)) = \sigma(\hat{x}_{ub})^{\beta} + (1 - \sigma(\hat{x}_{ub})^{\beta}) DivGain(b, \hat{\mathbf{R}}(k))$$

$\beta \in [0,1]$: balance coefficient

Outline

- Introduction
- Proposed Method
- **<u>Experiments</u>**
- Conclusion

Experimental Questions

- **Q1. Performance comparison**
  - Does PopCon provide the best trade-off between accuracy and aggregate diversity?
- **Q2. Ablation study**
  - How do the main ideas help improve the performance?

Datasets

- We use **<u>three real-world datasets</u>**
  - U, B, I: users, bundles, and items, resp.

Dataset	#U	#B	#I	#U-B (dens.)	#U-I (dens.)	#B-I (dens.)	Avg. B size
Steam[1]	29,634	615	2,819	87,565 (0.48%)	902,967 (1.08%)	3,541 (0.20%)	5.76
Youshu[2]	8,039	4,771	32,770	51,377 (0.13%)	138,515 (0.05%)	176,667 (0.11%)	37.03
NetEase[3]	18,528	22,864	123,628	302,303 (0.07%)	1,128,065 (0.05%)	1,778,838 (0.06%)	77.80

  - Steam: game platform
  - Youshu: book review platform
  - NetEase: cloud music platform

Baselines and Backbones

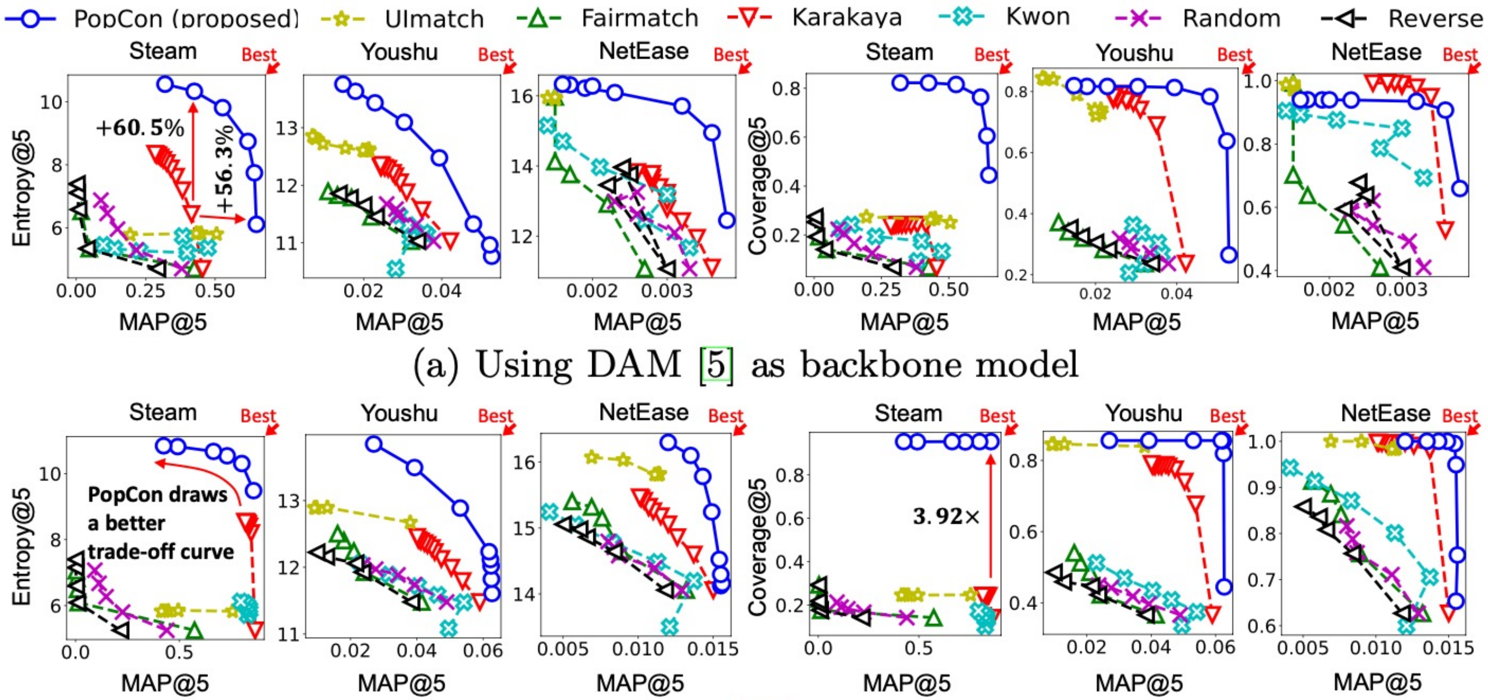
- We compare PopCon with **six baselines**
  - Reverse, Random, Kwon, Karakaya, Fairmatch, and UImatch
    - They rerank the backbone's results by treating bundles as atomic units
- We leverage **two backbone models**
  - DAM: SOTA matrix factorization-based model
  - CrossCBR: SOTA graph learning-based model

Evaluation

- **Leave-one-out protocol**
  - One of each user’s interactions is randomly selected for testing
- **Metrics**
  - MAP@5 (for accuracy)
  - Coverage@5 (for aggregate diversity)
  - Entropy@5 (for aggregate diversity)

Experimental Results (1/2)

- **Performance comparison**
  - **Q1.** Does PopCon provide the best trade-off between accuracy and aggregate diversity?
  - **A1.** PopCon outperforms baselines noticeably



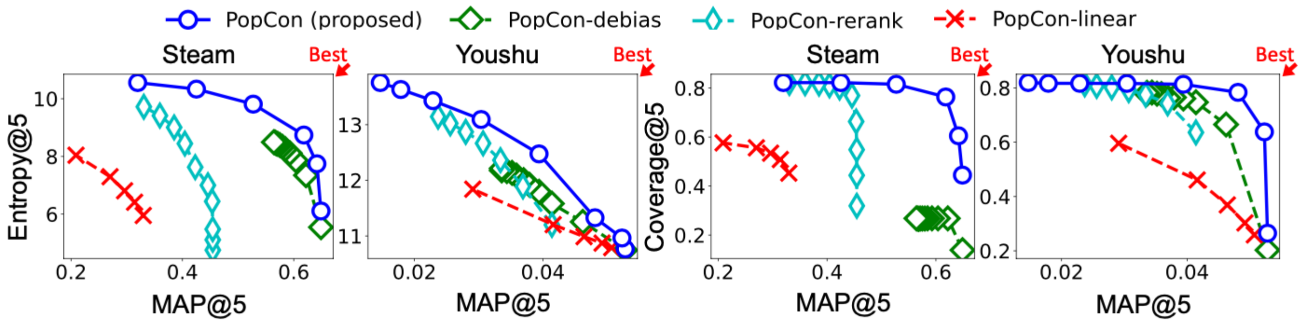
(a) Using DAM [5] as backbone model

(b) Using CrossCBR [14] as backbone model

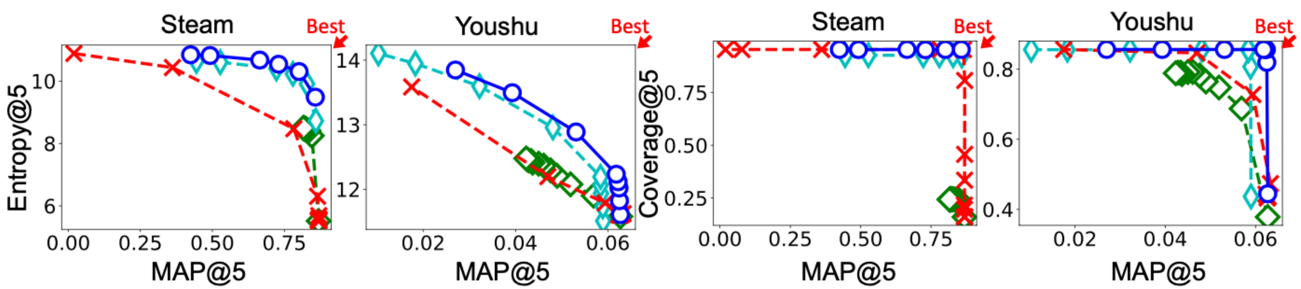
Experimental Results (2/2)

Variants	Training	Reranking
PopCon	Ours	Ours
PopCon-debias	Ours	Karakaya (baseline)
PopCon-rerank	No debiasing	Ours
PopCon-linear	Ours	Weighted sum

- **Ablation study**
  - **Q2.** How do the main ideas help improve the performance?
  - **A2.** All the main ideas help improve the performance



(a) Usage of DAM [5] as the backbone model



(b) Usage of CrossCBR [14] as the backbone model

Outline

- Introduction
- Proposed Method
- Experiments
- **<u>Conclusion</u>**

Conclusion

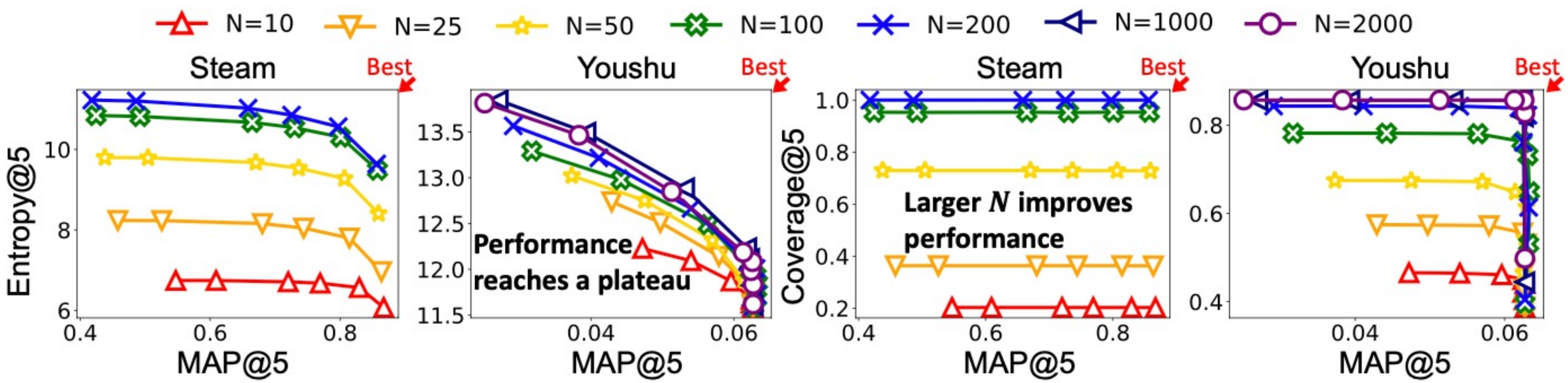
- We propose **PopCon** for aggregately diversified bundle recommendation
- Three main ideas of PopCon
  - 1) Popularity-based negative sampling
  - 2) Maximizing the gains of coverage and entropy
  - 3) Accuracy-prioritized coupling
- PopCon outperforms all baselines significantly
  - Experiments on three real-world datasets
  - It achieves up to **60.5%** higher Entropy@5 and **3.92×** higher Coverage@5 with comparable accuracies compared to the best competitor

Thank you!

Personal website: https://jeon185.github.io
Code: https://github.com/snudatalab/PopCon

Appendix
Experimental Results

- **Effects of number of candidates**
  - **Q3.** How does the number N of candidates affect the performance?
  - **A3.** The performance is improved as N increased and finally reaches a plateau



*Using **CrossCBR** as the backbone model*