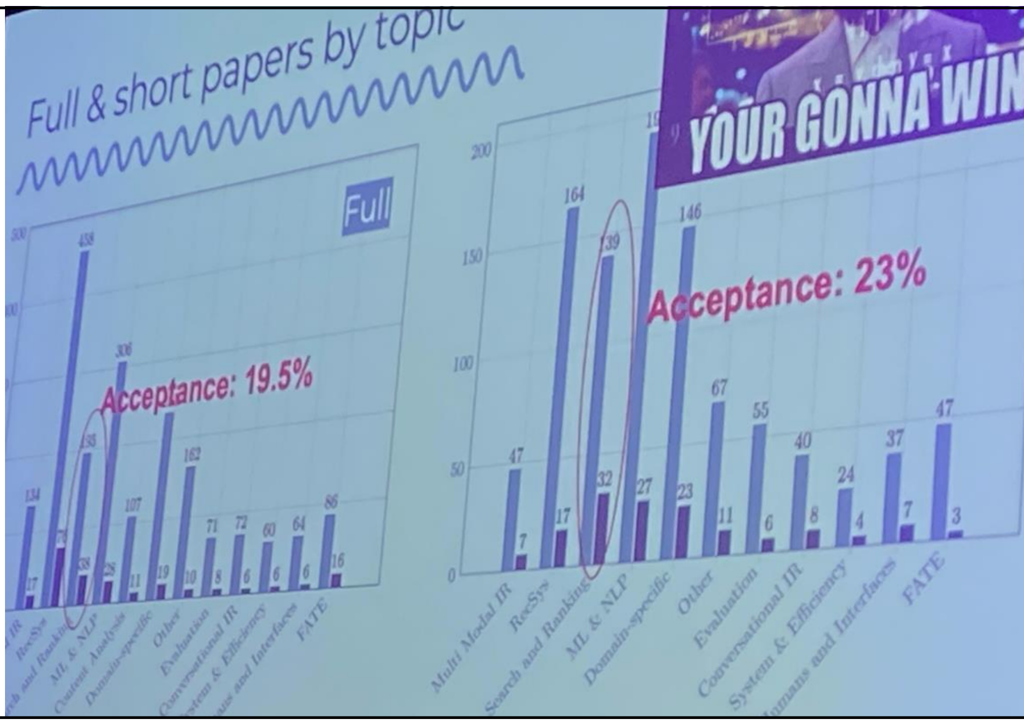
# Graph 13. Graph-based Learning for Recommender Systems

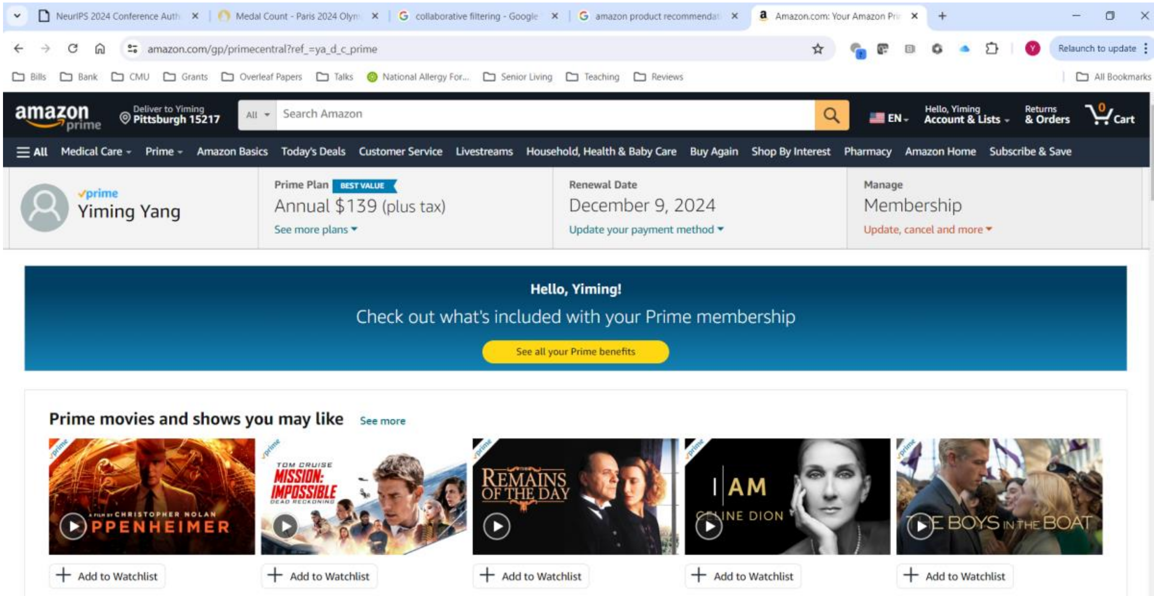
1




2

2

# Amazon Product Recommendation




3

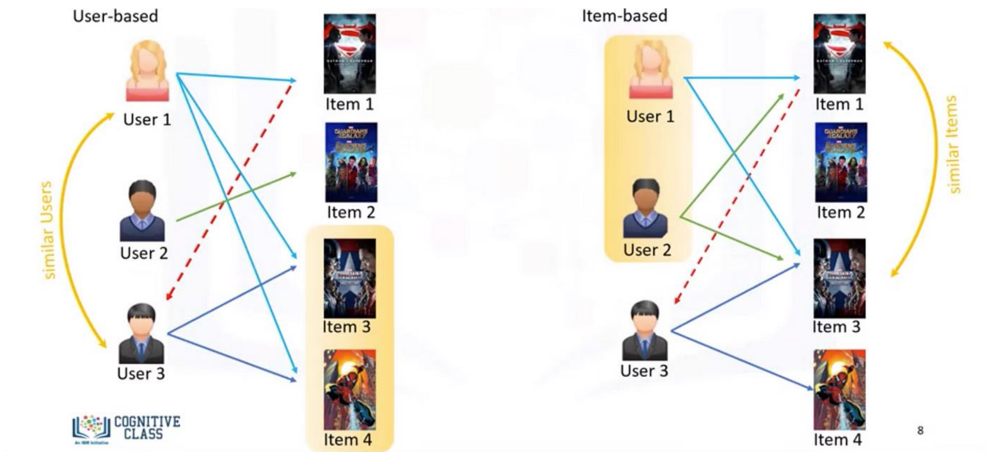
## Recommender Systems

- **Task**
  - Recommending items to each user based on the system-estimated relevance
- **Question**
  - How can the system predict the interests of each user over items?
- **Answers**
  - Based on the user's query, e.g., "recent Oscar winning movies" — Content-based Retrieval
  - Based on her past records of users, e.g., Tom Hank seems to be her favorite actor — Collaborative filtering (CF)



4

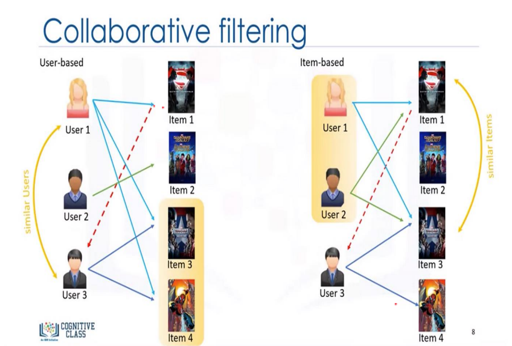
## Bipartite Graph with Users (Left) and Items (Right)






5

## CF with User-based or Item-based Reasoning




- **Left (user-based):** “User1 and User3 have similar tastes” (as both like items 3 & 4) & “User1 likes Item1” → User3 also likes Item1 (red).
- **Right (item-based):** “Item3 and Item1 are similar” (as both are liked by user1)” & “User3 likes Item3” → User3 also like Item1 (red).
- **Insight**: We need to measure the user-user or item-item similarity based on the bipartite graph.
- **Approach 1**: Using the adjacency matrix of the graph to compute the similarities (as dot-product or cosine).
- **Approach 2**: Graph-based embedding for both users and items, and then calculating their similarities (today’s lecture)



6

## Representing CF Training Data

- Consider a matrix with $n$ users as the rows and $m$ items as the columns
  - "+" shows an observed user-item interaction;
  - "?" shows a cell with the missing value (to predict).
- **Question:** How to deal with the missing values for training?
  - Naively, we can replace "+" by 1 and "?" by 0.
  - Then, the trained model tend to predict zero for each "?", which is not what desirable!

[the BPR paper by Rendle et al. in UAI 2009]

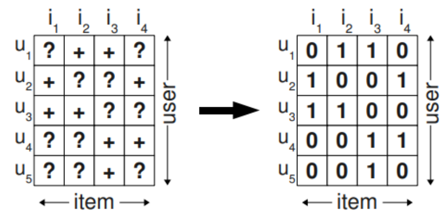



Figure 1: On the left side, the observed data $S$ is shown. Learning directly from $S$ is not feasible as only positive feedback is observed. Usually negative data is generated by filling the matrix with 0 values.





## Issue & Remedies

- Issue
  - Replacing each "?" by "0" sends the wrong signal for training, i.e., it assumes each user does not want to watch any movie s/he has not watched yet.
- **Remedies**
  - Imputation methods (replacing "?" by a neutral value or global average) (omitted today)
  - Bayesian Personalized Ranking (BPR) by Rendle et al. in UAI 2009
  - Neural Graph Collaborative Filtering (NGCF) by X Wang et al., SIGIR 2019

## BPR for Implicit CF (Rendle et al. in UAI 2009)

- **Construct a training set** with personalized preferences

$$D_S := \{(u,i,j) | i \in I_u^+ \wedge j \in I \setminus I_u^+\}$$

  - Here $I$ is the full set of items; $I_u^+ \subseteq I$ is the subset of items which user $u$ has observed links;
  - $I \setminus I_u^+$ is the complement set of items which user $u$ are not linked to (i.e., the ones with "?").

- **Bayesian Objective for Training**

$$\max_{\Theta} \sum_{(u,i,j) \in D_S} \{ln\ P(f_\Theta(u,i) > f_\Theta(u,j)) + lnP(\Theta)\}$$

  - Here $\Theta$ is the set of model parameters; $f_\Theta(.,.)$ is the system-learnt similarity function.
  - The 1st term in the objective is used to reinforce pairwise preference over items for each user; the 2nd term is to impose a Bayesian-prior of the model parameters.



9

## How to model user preference over item pairs?

- **BPR-MF** as an example (and BPR-KNN as another example in the paper)
  - Define $\Theta \in \mathbb{R}^{(n+m)\times d}$ as the embedding matrix (each row is the embedding of a user or item).
  - Denote the embeddings of $u, , i\ and\ j$ by $e_u, e_i, e_j \in \mathbb{R}^d$, respectively.
  - Define $f_\Theta(u,i) = \langle e_u, e_i \rangle$ as the choice of similarity function (other choices could be cosine, MLP or any kernel function).
  - Estimate probabilities as

$$P(f_\Theta(u,i) > f_\Theta(u,j)) \coloneqq \sigma(x_{uij}) := \frac{1}{1+e^{-x_{uij}}}, \quad x_{uij} \triangleq \langle e_u, e_i \rangle - \langle e_u, e_j \rangle$$

In $P(\Theta) = \lambda ||\Theta||^2$ under the assumption $\Theta \sim N(0, \lambda I)$



10

# BPR-MF vs. SVD (as a baseline)

- **BPR**
  - Using the ?-marked cells (right) to formulate personalized preferences
- **SVD**
  - Treating all the ?'s as the same value of 0
  - Overfitting the matrix after unreasonable imputation

[from the BPR paper by Rendle et al. in UAI 2009]

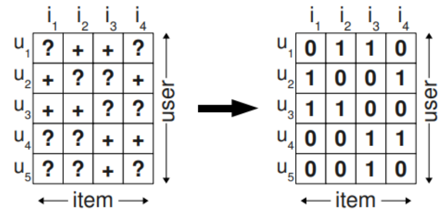



Figure 1: On the left side, the observed data $S$ is shown. Learning directly from $S$ is not feasible as only positive feedback is observed. Usually negative data is generated by filling the matrix with 0 values.





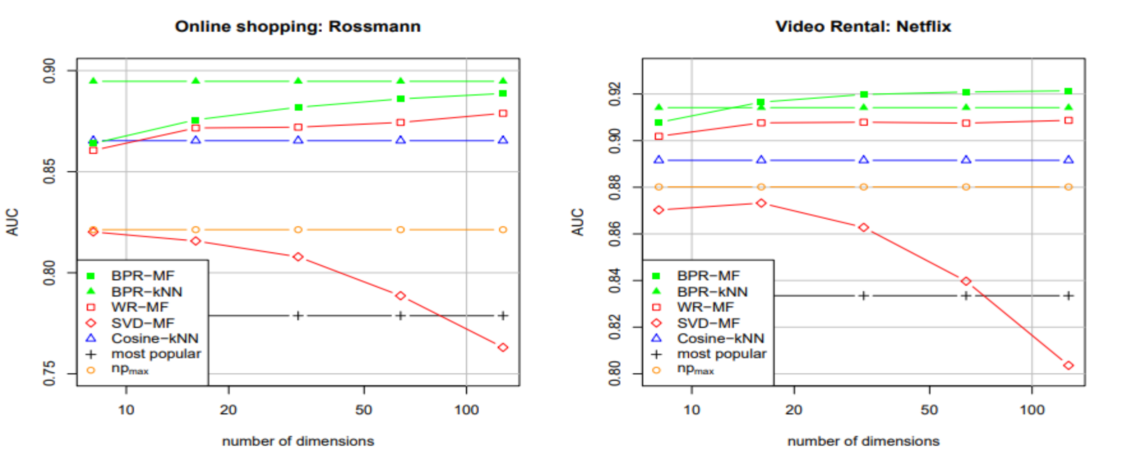



Figure 6: Area under the ROC curve (AUC) prediction quality for the Rossmann dataset and a Netflix subsample. Our BPR optimizations for matrix factorization BPR-MF and k-nearest neighbor BPR-kNN are compared against weighted regularized matrix factorization (WR-MF) [5, 10], singular value decomposition (SVD-MF), k-nearest neighbor (Cosine-kNN) [2] and the most-popular model. For the factorization methods BPR-MF, WR-MF and SVD-MF, the model dimensions are increased from 8 to 128 dimensions. Finally, $np_{max}$ is the theoretical upper bound for any non-personalized ranking method.

## Issue & Remedies

- Issue.
  - Replacing each "?" by "0" sends the wrong signal for training, i.e., it assumes each user does not want to watch any movie s/he has not watched yet.
- **Remedies**
  - Imputation methods (replacing "?" by a neutral value or global average) (omitted)
  - Bayesian Personalized Ranking (BPR) by Rendle et al. in UAI 2009
  - Neural Graph Collaborative Filtering (NGCF) by X Wang et al., SIGIR 2019





## Major Difference between BPR and NGCF

- BPP (and many early CF methods) leverages first-order connectivity only.
- NGCF (and many later CF methods) leverages higher-order connectivity by performing multi-layer embedding of nodes with a graph neural network.

## Higher-order Connectivity over a Bipartite
[X Wang et al., SIGIR 2019]

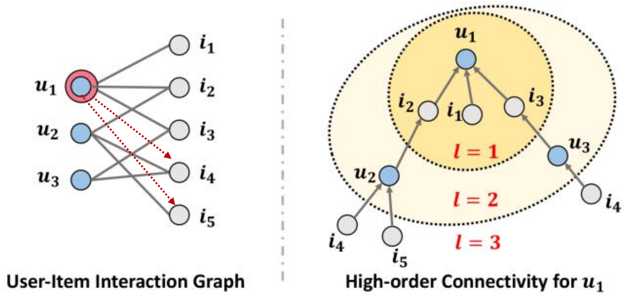



**Figure 1: An illustration of the user-item interaction graph and the high-order connectivity. The node $u_1$ is the target user to provide recommendations for.**





## NGCF Node/Item Embedding in Vector Formulation
[X. Wang et al., SIGIR 2019]

- **Multi-layer Node Embedding** (item embedding is similar)

$$h_u^{(l+1)} := \sigma\left(v_{u\leftarrow u}^{(l)} + \sum_{i\in\mathcal{N}_u} v_{u\leftarrow i}^{(l)}\right)$$

$$v_{u\leftarrow u}^{(l)} := W_1^{(l)}\, h_u^{(l)}, \qquad \boxed{\text{Self-looping}}$$

$$v_{u\leftarrow i}^{(l)} := \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}}\left(W_1^{(l)}\, h_i^{(l)} + W_2^{(l)}\left(h_i^{(l)} \odot h_u^{(l)}\right)\right) \quad \boxed{\text{Neighborhood Aggregation}}$$

- Node embedding and item embedding are mutually depending on each other.
- Through multi-layer embedding we capture the user-item connectivity in the higher-order graph.

# NGCF Node/Item Embedding in Matrix Formulation
[X. Wang et al., SIGIR 2019]

- **At the input layer**
  - We have $H^{(0)}$ as an $(n+m) \times d$ matrix whose rows the initial embeddings of users and items..
- **At a higher layer**

$$H^{(l+1)} := \sigma\left(\underbrace{\tilde{A}H^{(l)}W_1^{(l)}}_{\text{Self-looping}} + \underbrace{\left(A_{sys}(H^{(l)} \odot H^{(l)})\right)W_2^{(l)}}_{\text{Neighborhood Aggregation}}\right)$$

- **At the final layer**

$$e_u^* = e_u^{(0)} \parallel e_u^{(1)} \parallel \cdots e_u^{(L)}, \quad e_i^* = e_i^{(0)} \parallel e_i^{(1)} \parallel \cdots e_i^{(L)}$$

$$\hat{y}_{u,i} = \langle e_u^*, e_i^* \rangle, \qquad Loss = \sum_{(u,,i,j)\in D_s}\{ln\, \sigma\left(\hat{y}_{u,i} - \hat{y}_{u,j}\right) + \lambda\|\Theta\|^2\}$$





# Comparison of GCN vs. NGCF

❑ **Conventional GCN** [Thomas N. Kipf and Max Welling, ICLR 2017]

$$H^{(l+1)} := \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right), \quad H^{(l)} \in \mathbb{R}^{(n+m)\times d^{(l)}}$$

where $H^{(l+1)} \in R^{n\times d}$ is a matrix of graph-based node embeddings.

❑ **Neural Graph Collaborative Filtering (NGCF)** [X. Wang et al., SIGIR 2019]

$$H^{(l+1)} := \sigma\left(\tilde{A}H^{(l)}W_1^{(l)} + \left(A_{sys}(H^{(l)} \odot H^{(l)})\right)W_2^{(l)}\right)$$

where $H^{(l+1)} \in R^{(n+m)\times d}$ is a matrix of higher-order graph node/edge embeddings.

## Adjacency Matrix for a Bipartite Graph

- **Initial Adjacency Matrix**

$$A \triangleq \begin{bmatrix} 0 & R \\ R^T & 0 \end{bmatrix} \in \{0,1\}^{(m+n)\times(m+n)} \text{ with } \mathrm{A[u,i]} = 1 \text{ iff } edge(u,i) \in E$$

- **Symmetrical Normalization**

$$A_{sym} \triangleq D^{-\frac{1}{2}} \mathrm{A}\, D^{-\frac{1}{2}} \text{ with } A_{sym}[\mathrm{i,j}] = \frac{A_{ij}}{\sqrt{N_i}\sqrt{N_j}} \qquad (N_i \text{ and } N_j \text{ are the node degrees})$$

$$D \triangleq \mathrm{diag}\{D_{ii}\}_{i=1}^{m+n}, \quad D_{ii} = \textstyle\sum_{j=1}^{n} A_{ij} \qquad \text{(sum of the elements in each row)}$$

- **Adding Self-loop**

$$\tilde{A} \triangleq \mathrm{I} + A^{sys} = \mathrm{I} + D^{-\frac{1}{2}} \mathrm{A}\, D^{-\frac{1}{2}}$$





## Closer Look at $H^{(l+1)} := \sigma\left(\boxed{\tilde{A}H^{(l)}} W_1^{(l)} + \left(A_{sys}(H^{(l)} \odot H^{(l)})\right) W_2^{(l)}\right)$

$$\tilde{A}H^{(l)} = (I + A_{sym})H^{(l)} = H^{(l)} + \boxed{A_{sym}\ H^{(l)}}$$

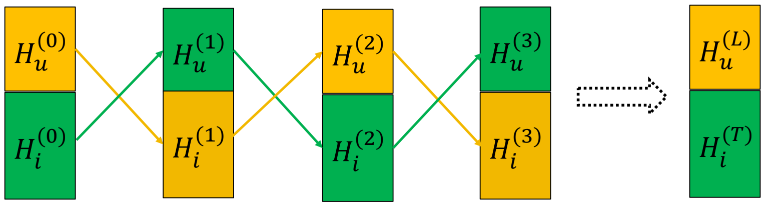
Neighborhood Aggregation

$$\overset{A_{sym}}{\begin{bmatrix} 0 & R_{sym} \\ R_{sym}^T & 0 \end{bmatrix}} \overset{H^{(l)}}{\begin{bmatrix} H_u^{(l)} \\ H_i^{(l)} \end{bmatrix}} = \begin{bmatrix} H_u^{(l+1)} = R_{sym} H_i^{(l)} \\ H_i^{(l+1)} = R_{sym}^T H_u^{(l)} \end{bmatrix}$$

$$R_{sys}[\mathrm{i,j}] = \frac{1}{\sqrt{|\mathcal{N}_i||\mathcal{N}_j|}}$$

$A_{sys}\ H^{(l)}$ is doing the cross-bipartite update.

## Multi-layer Neighborhood Aggregation




$$H^{(L)} = f\left(A_{sym} H^{(L-1)}\right) = \cdots = f\left(\cdots f\left(A_{sym}\, H^{(0)}\right) \cdots\right)$$

$$H_u^{(l+1)} = R_{sym} H_i^{(l)} \qquad H_i^{(l+1)} = R_{sym}^T H_u^{(l)}$$

$$y^{(l+1)} = A x^{(l)} \qquad x^{(l+1)} = A^T y^{(l)} \quad \Rightarrow \quad \textbf{HITS}$$





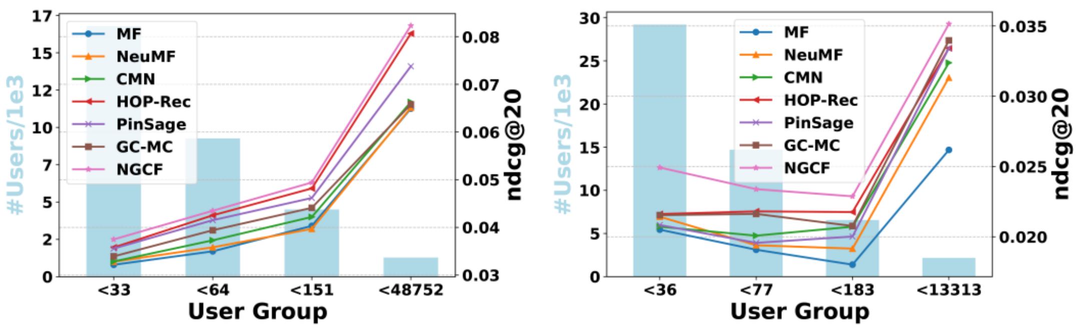
## Empirical Results [X Wang et al., SIGIR 2019]

**Table 2: Overall Performance Comparison.**

| | Gowalla | | Yelp2018* | | Amazon-Book | |
|---|---|---|---|---|---|---|
| | recall | ndcg | recall | ndcg | recall | ndcg |
| **BPR-MF** MF | 0.1291 | 0.1109 | 0.0433 | 0.0354 | 0.0250 | 0.0196 |
| NeuMF | 0.1399 | 0.1212 | 0.0451 | 0.0363 | 0.0258 | 0.0200 |
| CMN | 0.1405 | 0.1221 | 0.0457 | 0.0369 | 0.0267 | 0.0218 |
| HOP-Rec | 0.1399 | 0.1214 | 0.0517 | 0.0428 | 0.0309 | 0.0232 |
| GC-MC | 0.1395 | 0.1204 | 0.0462 | 0.0379 | 0.0288 | 0.0224 |
| PinSage | 0.1380 | 0.1196 | 0.0471 | 0.0393 | 0.0282 | 0.0219 |
| **3 layers** **NGCF-3** | **0.1569*** | **0.1327*** | **0.0579*** | **0.0477*** | **0.0337*** | **0.0261*** |
| %Improv. | 11.68% | 8.64% | 11.97% | 11.29% | 9.61% | 12.50% |
| *p*-value | 2.01e-7 | 3.03e-3 | 5.34e-3 | 4.62e-4 | 3.48e-5 | 1.26e-4 |

Compared to the best baseline (underlined)

# Empirical Results [X Wang et al., SIGIR 2019]




(b) ndcg on Yelp2018*

(c) ndcg on Amazon-book



23

# Concluding Remarks

- Recommender systems are a **prominent research topic** and hold significant practical importance across **various industries**.
- **Neural representation learning for users/items** plays a key role in recent methods.
- GNNs allow us to leverage **higher-order connectivity in bipartite graphs.**
- **Limitation1:** BPR and NGCF cannot handle multi-grade feedback (e.g., 1-5 start book/hotel/restaurant ratings)
- **Limitation 2:** BPR and NGCF do not leverage the rich **rich information beyond the bipartite graph** has been widely studied (e.g., user profiles, friendships, user groups, movie actors/actresses/producers/genre, etc.)
- **Cold-start** issue remains to be a big challenge in the field.



24

# References


- Steffen Rendle, Christoph Freudenthaler, Zeno Gantner and Lars Schmidt-Thieme. **BPR: Bayesian Personalized Ranking from Implicit Feedback**. UAI-2009, https://arxiv.org/abs/1205.2618
- Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, Tat-Seng Chua. **Neural Graph Collaborative Filtering**, SIGIR-2019, https://arxiv.org/pdf/1905.08108
- Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, Bin Cui. **Graph Neural Networks in Recommender Systems: A Survey**. ACM Comput. Surv. 2022. https://arxiv.org/pdf/2011.02260