WSDM 2022 Tutorial

wsdm Phoenix, AZ, USA 2022

The 15th ACM International Conference on Web Search and Data Mining

February 21–25, 2022

Graph Neural Networks for Recommender System

Chen Gao[1], Xiang Wang[2], Xiangnan He[3] and Yong Li[1]

[1]Tsinghua University

[2]National University of Singapore

[3]University of Science and Technology of China

About us

Chen Gao

PostDoc Researcher

Tsinghua University

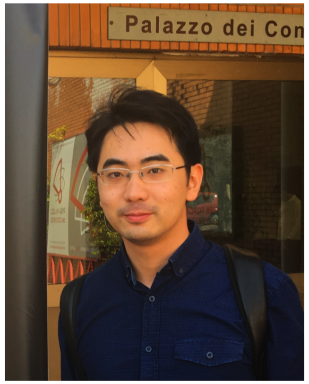
chgao96@gmail.com

Xiang Wang

PostDoc Researcher

National University of Singapore

xiangwang@u.nus.edu



Xiangnan He

Professor

University of Science and Technology of China

xiangnanhe@gmail.com

Yong Li

Associate Professor

Tsinghua University

liyong07@tsinghua.edu.cn

Outline

- Background
- Motivations and Challenges of GNN-based RecSys
- Recent Advances of GNN-based RecSys
    - I) Collaborative Filtering, Knowledge Graph-based RecSys
    - II) Feature-based Sequential/Bundle/Multi-behavior/Diversified RecSys
- Open Problems and Future Directions

Outline

- Background
  - Recommender System
  - Graph Neural Network

- Motivations and Challenges of GNN-based RecSys

- Recent Advances of GNN-based RecSys

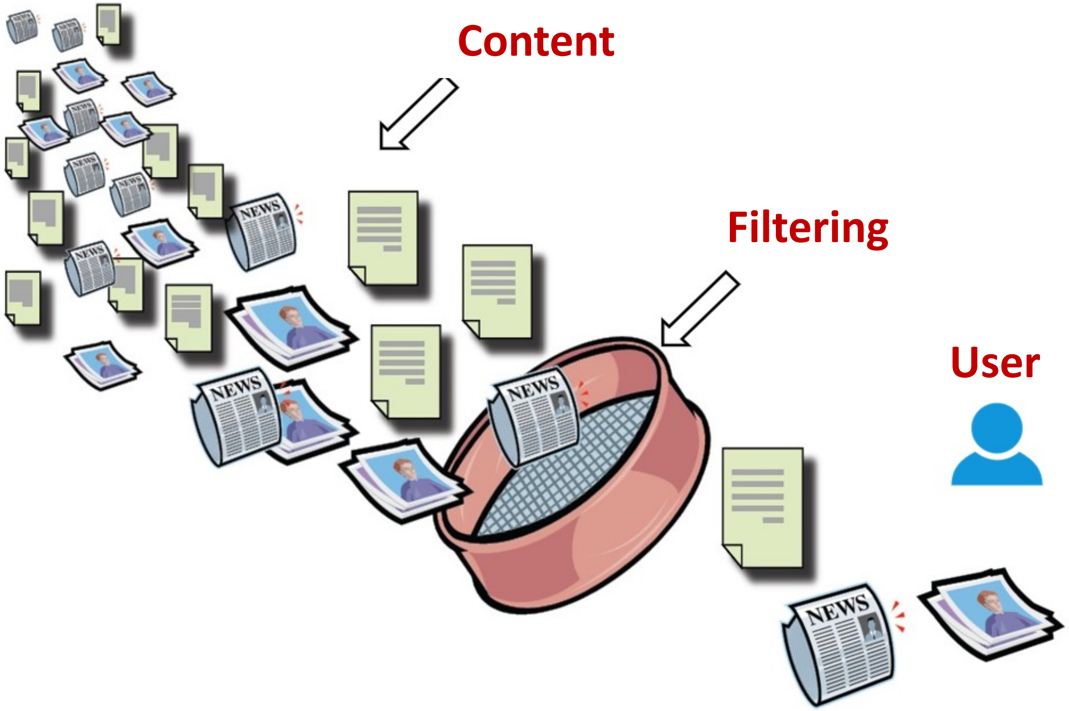
- Open Problems and Future Directions

Outline

- Background
  - Recommender System
  - Graph Neural Network
- Motivations and Challenges of GNN-based RecSys
- Recent Advances of GNN-based RecSys
- Open Problems and Future Directions

Information-Overload Era

- Information-overload in Internet
  - Weibo： >0.5B posts/day
  - Flickr: >0.3B images/day
  - Taobao： >1B products



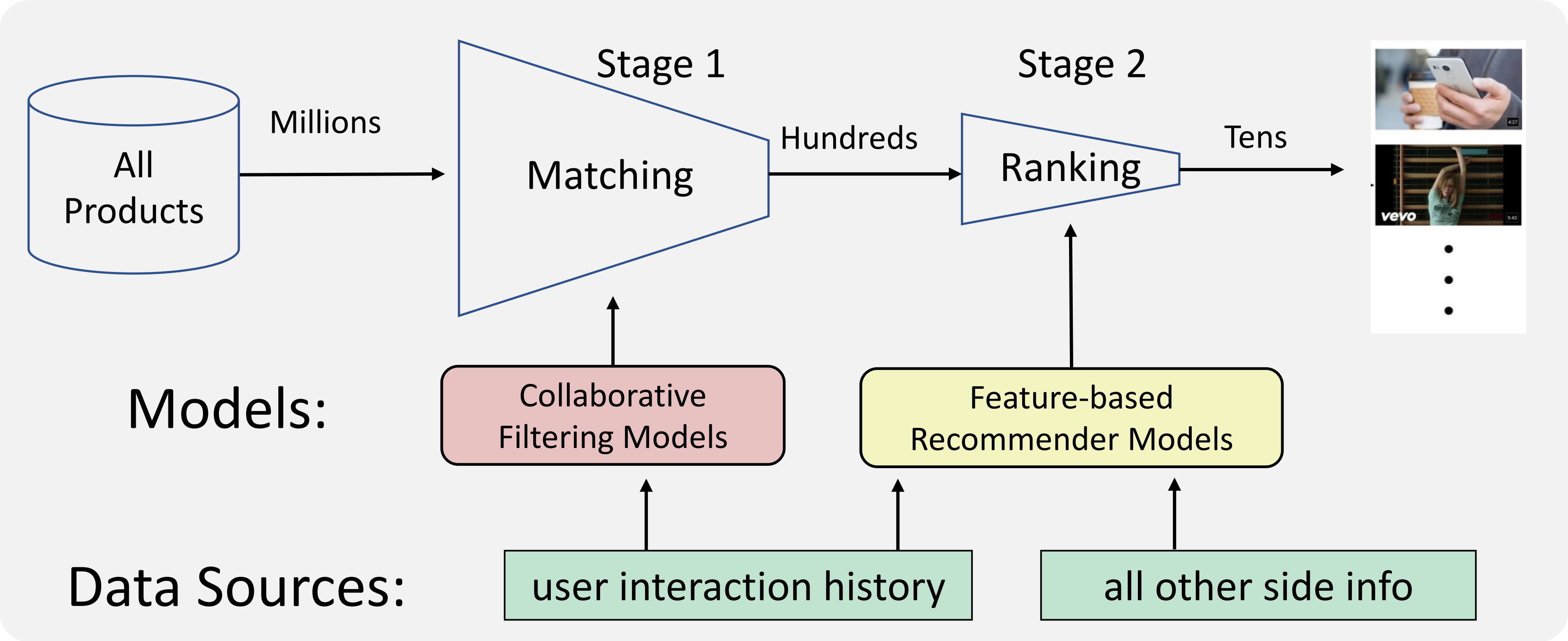
Effective/Efficient information filtering→Recommender System

Recommender System

- Overview of Recommender System
- **Stages**
  - Matching (collaborative filtering), ranking
- **Scenarios**
  - Social, Sequential, Session, Bundle, KG-Based, etc.
- **Objectives**
  - Accuracy, multi-behavior, diversity, explainability, fairness, etc.

Recommender System

- Stages
  - Matching: recall items from all-item pool
  - Collaborative-filtering models



Stage 1

Stage 2

All Products

Millions

Matching

Hundreds

Ranking

Tens

Models:

Collaborative Filtering Models

Feature-based Recommender Models

Data Sources:

user interaction history

all other side info

Recommender System

➢ Collaborative filtering

	items			
users	1	0	0	1
	0	1	0	0
	1	1	0	0
	1	0	0	1

0/1 Interaction matrix

OR

	items			
users	5	-	-	3
	-	2	-	-
	4	1	-	-
	3	-	-	3

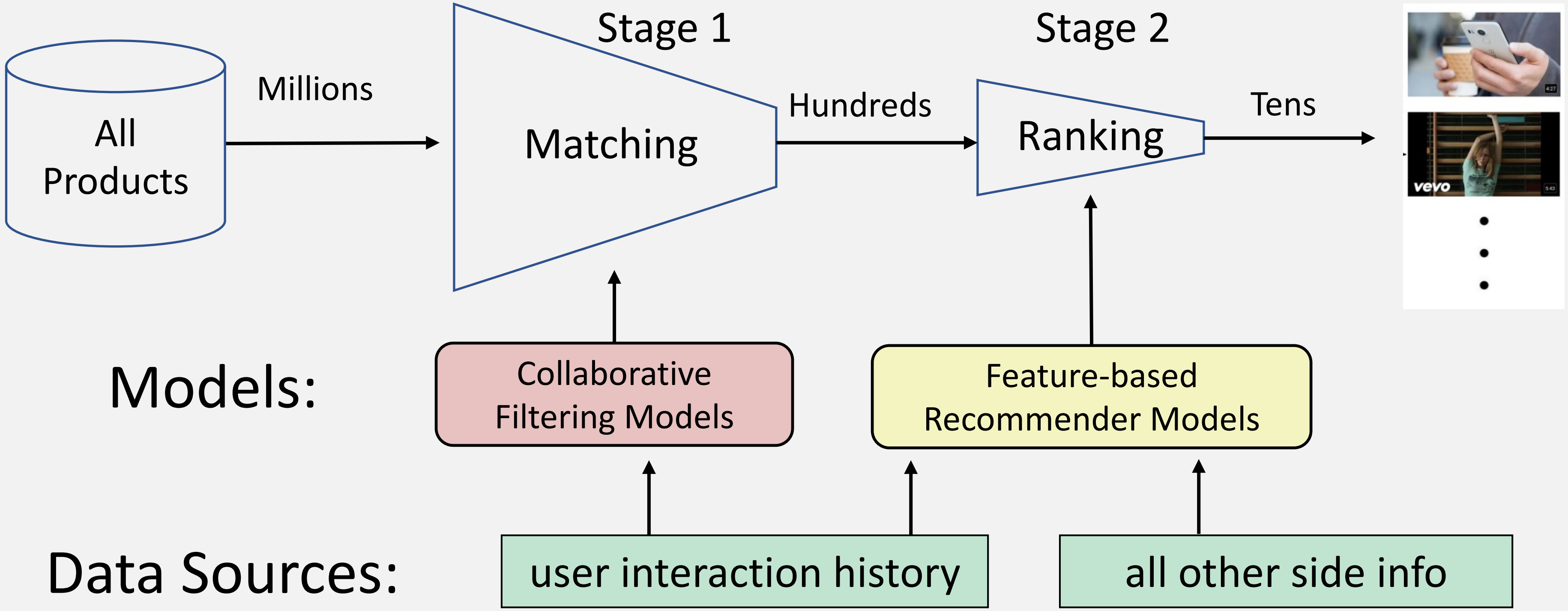
Rating matrix

- Implicit CF
- Application: e-commerce, ads, etc.
- Data: an interaction matrix
- Task: estimate positive position
- Measurement: Ranking metrics

- Explicit CF
- Application: movie, POI, etc.
- Data: a rating matrix (e.g. 1-5)
- Task: estimate ratings on unknown positions
- Measurement: Regression metrics

Recommender System

- Stages
  - Ranking: rank items from matching stage's output
  - Feature-based Recommender Models / CTR



Stage 1

Stage 2

All Products

Millions

Matching

Hundreds

Ranking

Tens

Models:

Collaborative Filtering Models

Feature-based Recommender Models

Data Sources:

user interaction history

all other side info

Recommender System

- ➢ Feature-based Recommender Models
  - ➢ Also known as Click-Through Rate Prediction
- ➢ Input: user/item attributes (ID can regarded as a kind of attribute)

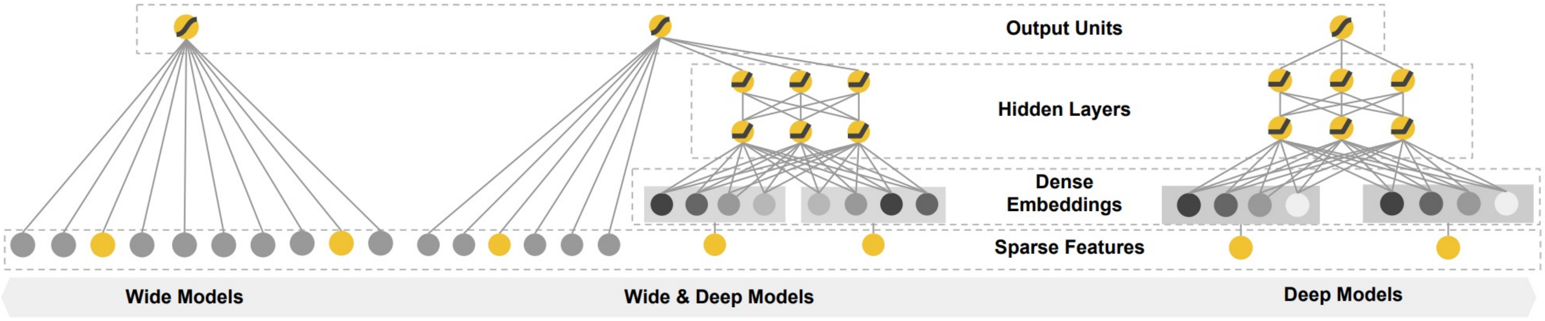
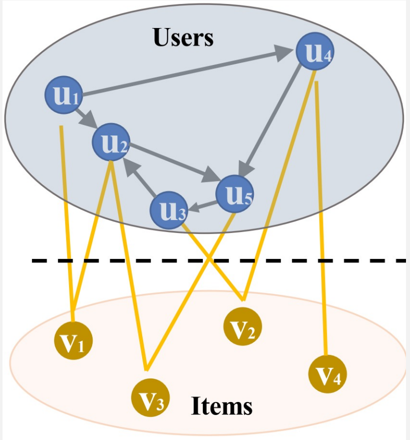


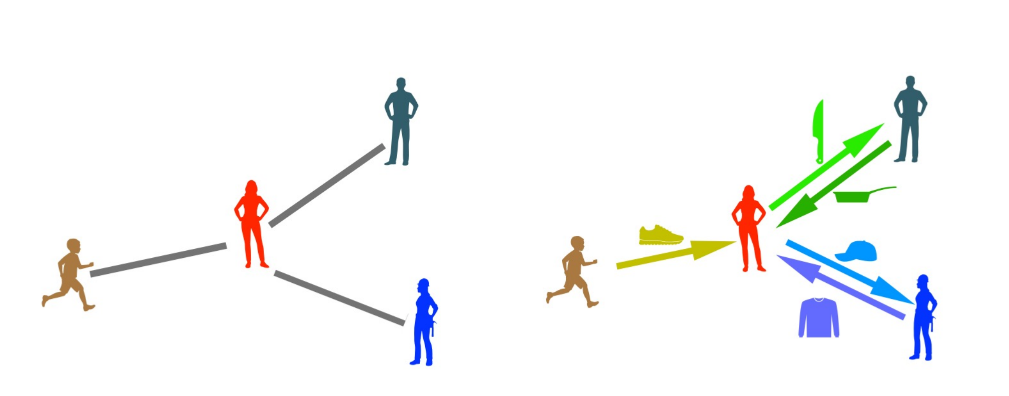
Figure from:
Cheng, H. T et al. Wide & deep learning for recommender systems. In Proceedings of the 1st workshop on deep learning for recommender systems

Recommender System

- Scenario: social recommendation
  - Definition: Improve recommendation with social network
  - *Social-trust* assumption: friends tend to have similar interests
  - **Input**: user interaction data + social relation data
  - **Output**: user-item interaction probability



Social Recommendation

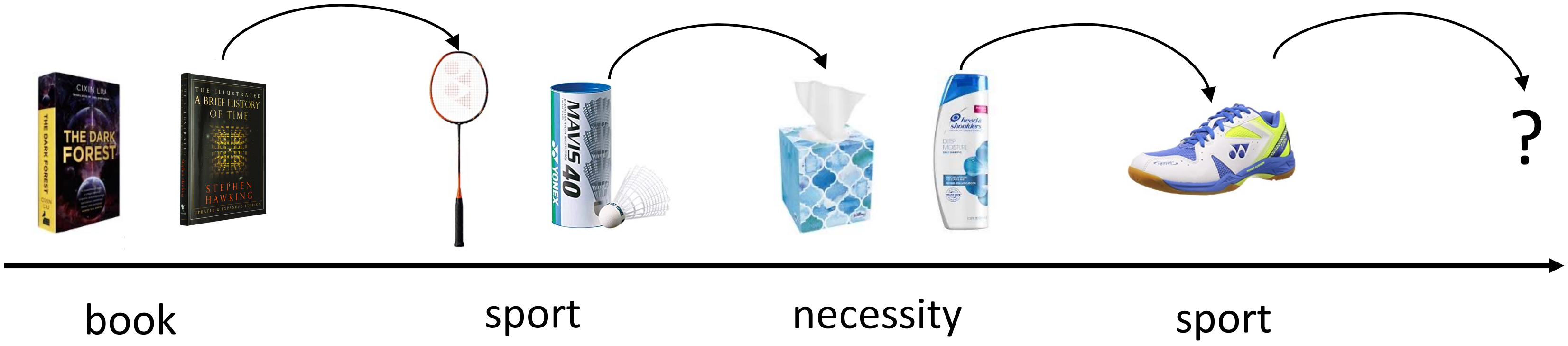


Traditional Social RecSys v.s Social E-Commerce RecSys, such as Pinduoduo

Figures are from:
Wu et al. DiffNet++: A Neural Influence and Interest Diffusion Network for Social Recommendation. TKDE 2020
Lin et al. Recommender Systems with Characterized Social Regularization. CIKM 2018

Recommender System

- Scenario: sequential recommendation
  - Definition: predict user's next interaction based on historical sequential interactions
  - **Input**: user-item interactions at timestamps $t_1, t_2, \ldots, t_n$
  - **Output**: user-item interaction at timestamp t_{n+1}



?

book

sport

necessity

sport

Recommender System

- Scenario: session-based recommendation
  - Definition: predict next interaction based on anonymous short sequences
  - **Input**: *anonymous* behavior sessions
  - **Output**: next interaction of a given session
  - *Difference with Sequential Recommendation*
    - **Anonymous** (No user ID)
    - **Repetitive** items in one session
    - **Shorter** (as is collected in a short period)

Recommender System

- Scenario: cross-domain recommendation
  - Definition: recommendation with multi-domain datasets
  - Improve the target domain's performance with the auxiliary domain
  - **Input**: user-item historical interactions in multiple domains
  - **Output**: user-item interaction probability at target domain(s)
  - Challenges
    - Different user behaviors
    - Different data distribution
    - No overlapped user/item

Recommender System

- Scenario: bundle/list recommendation
  - Definition: Recommend a bundle that is made with several items to user
  - **Input**: user-item/bundle historical interactions
  - **Output**: user-bundle interaction probability

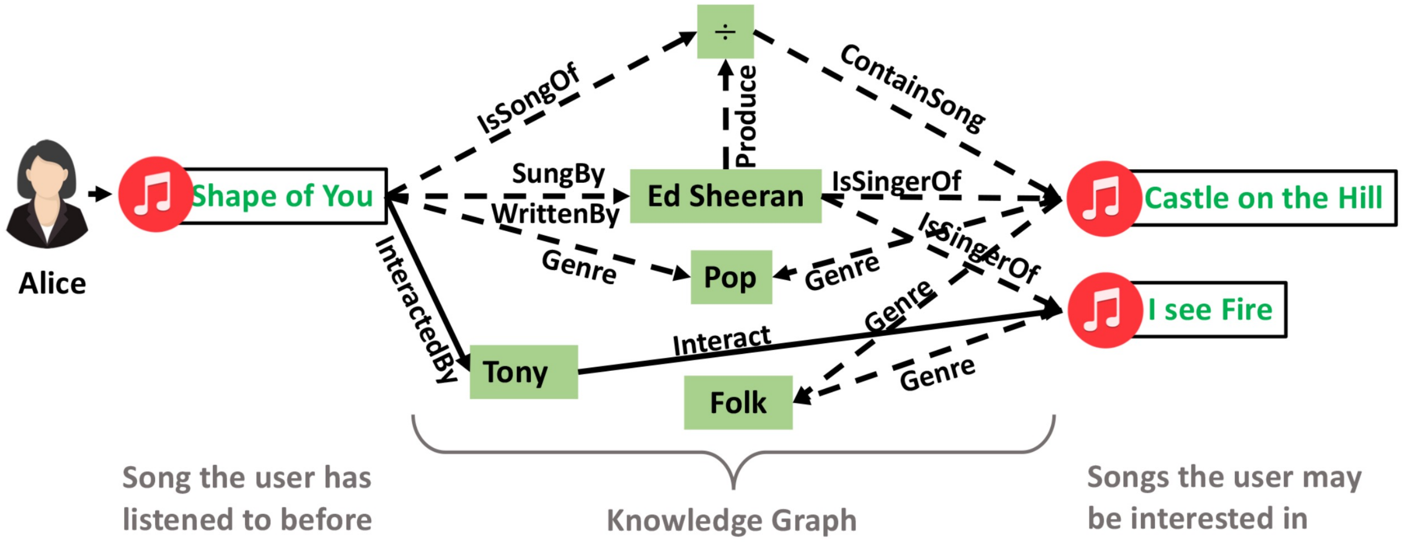
App Bundle

Suit Bundle

Game Bundle

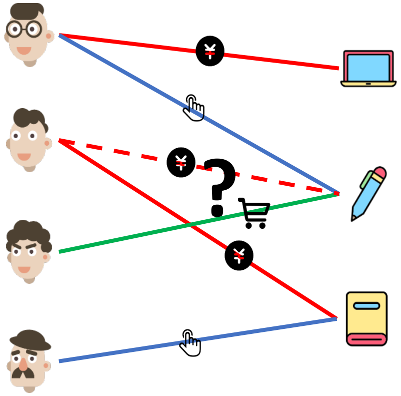
Recommender System

- Scenario: KG-based Recommendation
  - Definition: Improve recommendation with KG
  - **Input**: user-item interaction; knowledge graph
  - **Output**: user-item interaction probability



Recommender System

- Scenario: multi-behavior recommendation
  - In today's information systems, user can interact in multiple kinds of forms
    - Click, purchase, adding to cart, like, sharing, etc.
  - **Input:** user-item interaction on multiple behaviors
  - **Output**: user-item interaction probability on target behavior(s)



Recommender System

- Objective: accuracy (the most important)
  - Generally, it can be understood the ***whether the recommended items match with ground truth***
  - Top-K metrics
    - Hit Ratio (HR), Recall, NDCG, MRR, etc.
  - More metrics
    - AUC, GAUC, LogLoss, etc.
  - Most existing recommender systems are designed towards achieving high recommendation accuracy
    - *High accuracy → high CTR/CVR*

      → better user experience and higher business profit

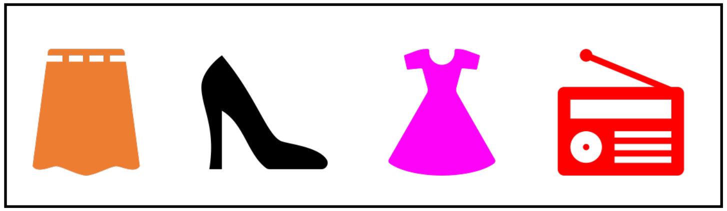
Recommender System

- Objective: diversity
  - Recommend diverse items to user while keeping high recommendation accuracy
  - Motivation: only pursuing high accuracy

    → the recommendation list become redundant

    → user can only be recommended certain categories of items
  - Metrics (always defined on **item category**)
    - Gini, entropy, coverage, etc.
    - Accuracy should be also considered of course

☹ accurate but redundant



😆 accurate and diverse

Recommender System

- Objective: explainability
  - What to explain
    - Two folds: explain 1) the model or 2) recommendation results
  - How to explain the model
    - Design explainable model
    - Such as attention modules, explicit feature-interaction, etc.
  - How to explain the results
    - User/Item-based explanation (CF effect / Social-trust)
    - Textual explanation (such as key words in reviews)
    - Knowledge-graph based explanation (via meta-path in KG)

Recommender System

- ➢ Objective: fairness
  - ➢ Motivation: users’ demand on to be fairly treated by RecSys

Can I be treated fairly on my gender?
Can I be treated fairly on my age?

Figure from Li et al. Personalized Counterfactual Fairness in Recommendation, SIGIR 2021

Recommender System

- Objective: privacy
  - When and where the privacy is highly concerned
    - Data collection: recommender may be the attacker
    - Data/model sharing: target company may be the attacker
    - Model/Results public-release: any third-party may be the attacker
  - Representative solutions
    - Transferring/sharing nosensitive model parameters
      - Distributed machine learning model
      - Sharing item-side information
    - Data protection mechanism
      - Data perturbations such as differential privacy-based ones
    - Federated learning

Outline

- Background
  - Recommender System
  - Graph Neural Network
- Motivations and Challenges of GNN-based RecSys
- Recent Advances of GNN-based RecSys
- Open Problems and Future Directions

Graph Neural Networks

- A message-passing-framework perspective
  - **Node embedding updated by neighbors**
  - **K-layer GNN access K-hop neighbors**
  - Named “Neighborhood propagation/aggregation”

- Representative variants of GNN
  - Spectral : GCN

$$\mathbf{H}^{l+1} = \delta(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{l}\mathbf{W}^{l})$$

  - Spatial: GraphSage (GAT, etc.)

$$\begin{aligned}\mathbf{h}^{l}_{\mathcal{N}_i} &= \mathrm{AGGREGATE}_l\left(\{\mathbf{h}^{l}_{j}, \forall j \in \mathcal{N}_i\}\right),\\ \mathbf{h}^{l+1}_{i} &= \delta\left(\mathbf{W}^{l}\left[\mathbf{h}^{l}_{i}\|\mathbf{h}^{l}_{\mathcal{N}_i}\right]\right),\end{aligned}$$

Graph Neural Networks

- Pro: Node feature + structural information
  - Embeddings contain 1) own features 2) neighbors’ features

- Keys
  - Where to deploy GNN layers
  - Design of propagation/aggregation layer
  - Depth of GNN layers

- Possible Cons
  - Over-smoothing, computational cost, etc.

Outline

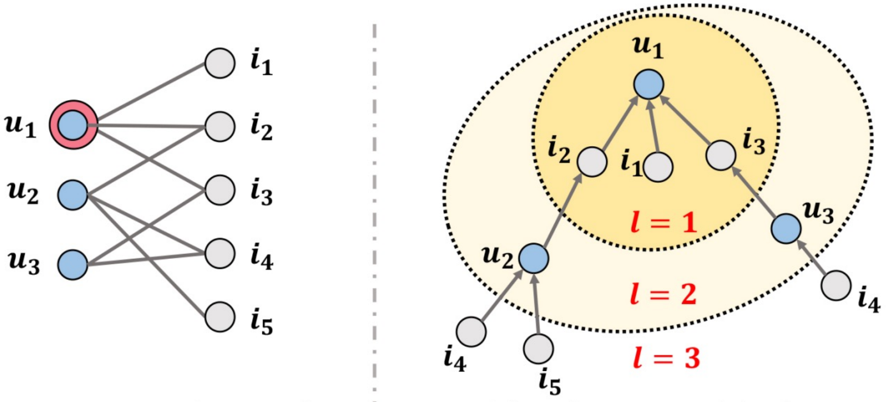
- Background
  - Recommender System
  - Graph Neural Network
- Motivations and Challenges of GNN-based RecSys
- Recent Advances of GNN-based RecSys
- Open Problems and Future Directions

Motivation: Why GNN are needed for RecSys

- High-order connectivity
- Supervision signal
- Structured data

Motivation: Why GNN are needed for RecSys

- High-order connectivity
  - Recommender systems rely on capturing similarity
    - User-user (User-CF), item-item (Item-CF), user-item (Model-CF)
  - GNN extends similarity to high-orders
    - Connectivity among high-order neighbors
  - Besides, data sparsity issue is well addressed



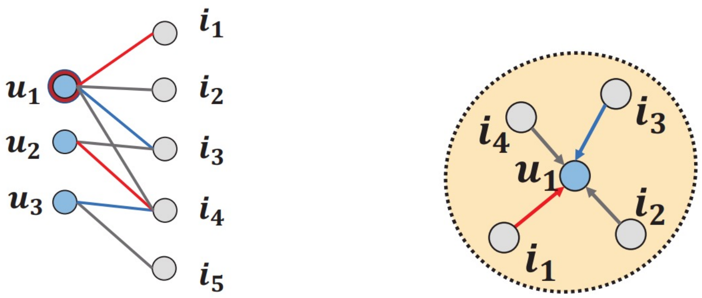
User-Item Interaction Graph

High-order Connectivity for u_1

Figures are from:
Wang et al. Neural Graph Collaborative Filtering, SIGIR 2019

Motivation: Why GNN are needed for RecSys

- Supervision signal
  - Users' feedback can be sparse
    - Semi-supervised signal in GNN learning
  - Users' feedback can be various
    - Well handled by various-form graph (nodes and edges)



(a) U-I Interaction Graph (b) Local Graph of u_1

Figures are from:
Jin et al. Multi-behavior Recommendation with Graph Convolutional Networks, SIGIR 2020

Motivation: Why GNN are needed for RecSys

- Structured data
  - The input of today’s recommender system is always structured
    - Can be utilized to construct graph
  - Learning from not only features but also structural information
    - Structural reveals implicit signals that cannot be learned by traditional works
  - GNN’s strong power to learn from graph-structured data

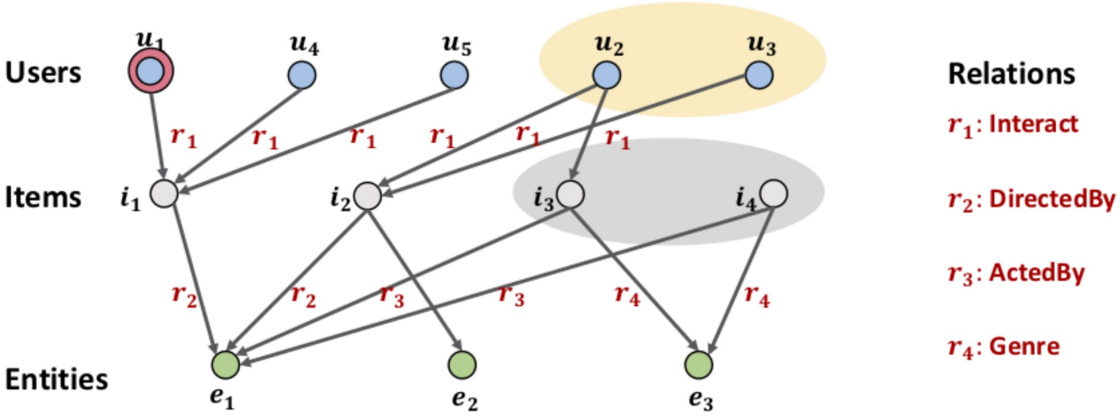


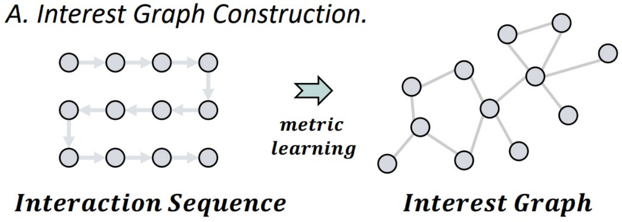
Figure from:
Wang et al. KGAT: Knowledge Graph Attention Network for Recommendation, KDD 2019

Challenges of GNN-based RecSys

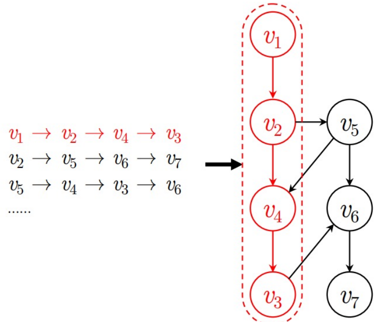
- Graph construction
- Message propagation and aggregation
- Model optimization

Challenges of GNN-based RecSys

- Graph construction
  - Node / edge definition
    - Heterogeneous/Homogenous
  - Distinguish more/less important, and even noisy data
  - Handle graph scale to balance efficiency and utility
    - Sampling, filtering, pruning, etc.
  - **Most importantly, the graph must match the key to the problem**



Sequential Recommendation


$v_1 \to v_2 \to v_4 \to v_3$
$v_2 \to v_5 \to v_6 \to v_7$
$v_5 \to v_4 \to v_3 \to v_6$
......
v_1
v_2
v_4
v_3
v_5
v_6
v_7

Session-based Recommendation

Figure from:
Chang et al. Sequential Recommendation with Graph Neural Networks, SIGIR 2021
Wu et al. Session-based Recommendation with Graph Neural Networks, AAAI 2019

Challenges of GNN-based RecSys

- Message propagation and aggregation
  - **How to propagate**
    - Neighbor set (uniform/attentional)
    - Path/Width
    - Propagation operations
  - **How to aggregate**
    - Utility & Efficiency
    - Aggregation operations
  - **Propagate-aggregate Depth**
- Model optimization
  - Optimization goal / loss function / data sampling / others

Outline

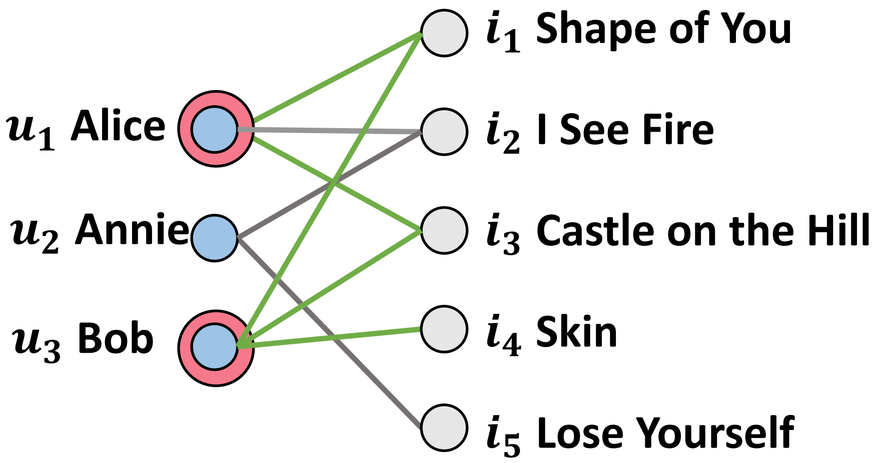
- **GNN for Collaborative Filtering**
  - **Q1: Are GNNs suitable for CF?**
    - **NGCF (SIGIR’2019)**
  - **Q2: How to tailor GNNs for CF?**
    - **LightGCN (SIGIR’2020)**
  - **Q3: How to inject self-supervised learning into GNN-based CF?**
    - **SGL (SIGIR’2021)**

- GNN for Knowledge Graph-based Recommendation
  - Q1: Are GNNs suitable for KG-based Rec?
    - KGAT (KDD’2019)
  - Q2: How to tailor GNNs for KG-based Rec?
    - KGIN (WWW’2021)

Q1: Are GNNs suitable for CF?
Recap Collaborative Filtering (CF)

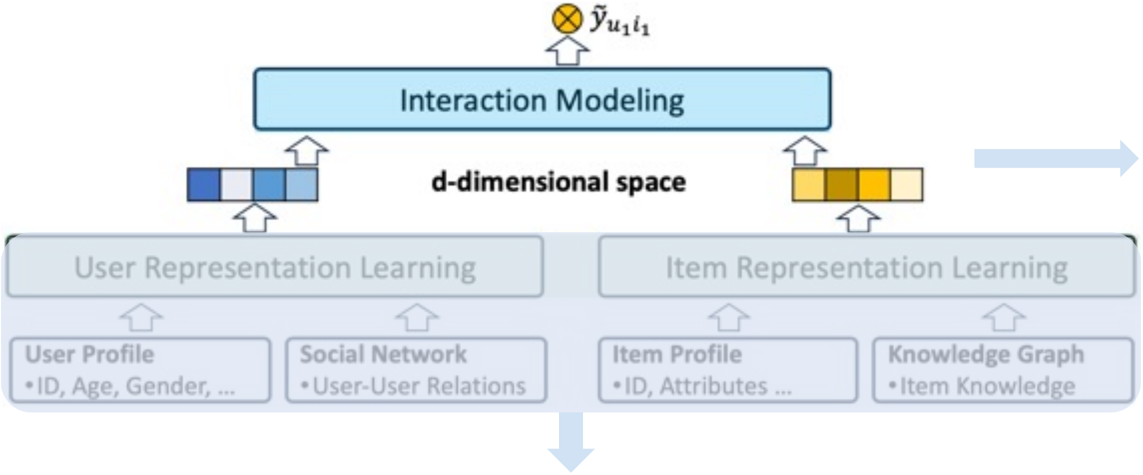
- **Collaborative Filtering**
  - Basic assumption: (behaviorally) similar users would have similar preferences on items

- **Collaborative Signal → Behavioral Patterns of Users**
  - if u_1 and u_3 have interacted with the same items $\{i_1, i_3\}$, u_1 is likely to have similar preferences on other items $\{i_4\}$.



Q1: Are GNNs suitable for CF?
Limitations of Current CF Models

Existing works are not sufficient to yield satisfactory embeddings for CF, due to the **implicit modeling of CF signals in Embedding function.**



e.g., matrix factorization (MF)

- Representation Learning: present ID of users and items as embedding vectors
- Interaction Modeling: inner product.

Representation Learning

- Mainly consider **descriptive features** (e.g., ID & attributes)
- **Without encoding CF signal explicitly**

Interaction Modeling

- Reconstruct user-item interactions, defining the objective function for model training
- Have to be well-designed to make up for the **deficiency of suboptimal embeddings**

Q1: Are GNNs suitable for CF?

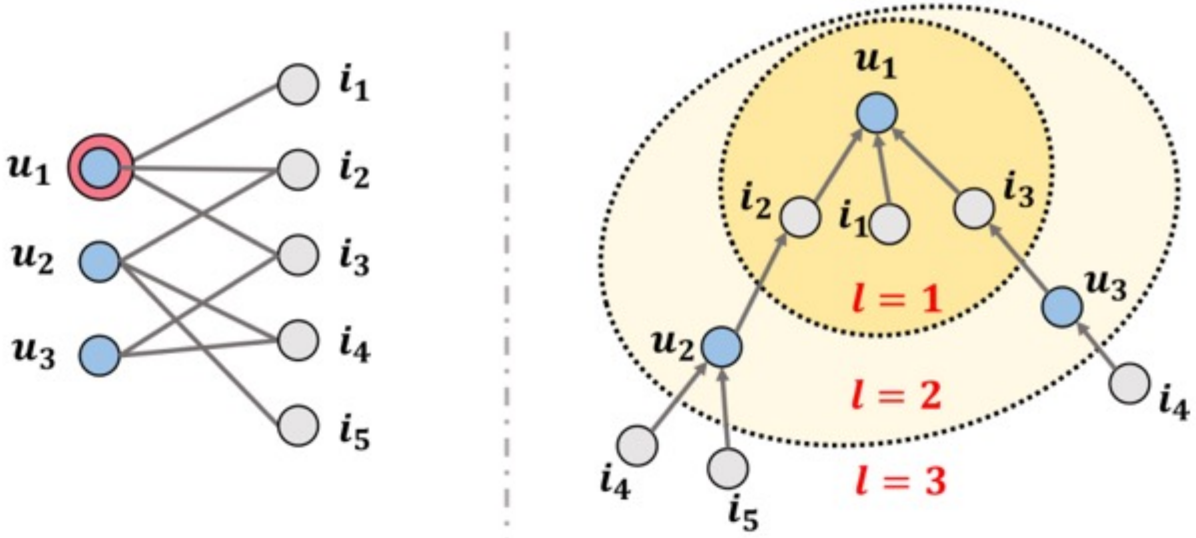
Revisiting CF via High-Order Connectivity

High-order Connectivity from User-item Interactions

- Definition: the paths that reach u_1 from any node with the path length l larger than 1.
- A natural way to encode collaborative signal in the interaction graph structure.

Why u_1 may like i_4?

- $u_1 \leftarrow i_2 \leftarrow u_2 \leftarrow i_4$
- $u_1 \leftarrow i_3 \leftarrow u_3 \leftarrow i_4$



User-Item Interaction Graph

High-order Connectivity for u_1

Q1: Are GNNs suitable for CF?

Modeling First-Order Connectivity

Inspired by GNNs

1. Propagate embeddings recursively on the user-item graph
2. Construct information flows in the embedding space

➢ Information Aggregation

message passed from i to u

$$\mathbf{m}_{u\leftarrow i} = \frac{1}{\sqrt{|\mathcal{N}_u||\mathcal{N}_i|}}\left(\mathbf{W}_1\mathbf{e}_i + \mathbf{W}_2(\mathbf{e}_i \odot \mathbf{e}_u)\right)$$

discount factor

message dependent on the neighbor

➢ Representation Update

$$\mathbf{e}_u^{(1)} = \mathrm{LeakyReLU}\left(\mathbf{m}_{u\leftarrow u} + \sum_{i\in\mathcal{N}_u}\mathbf{m}_{u\leftarrow i}\right)$$

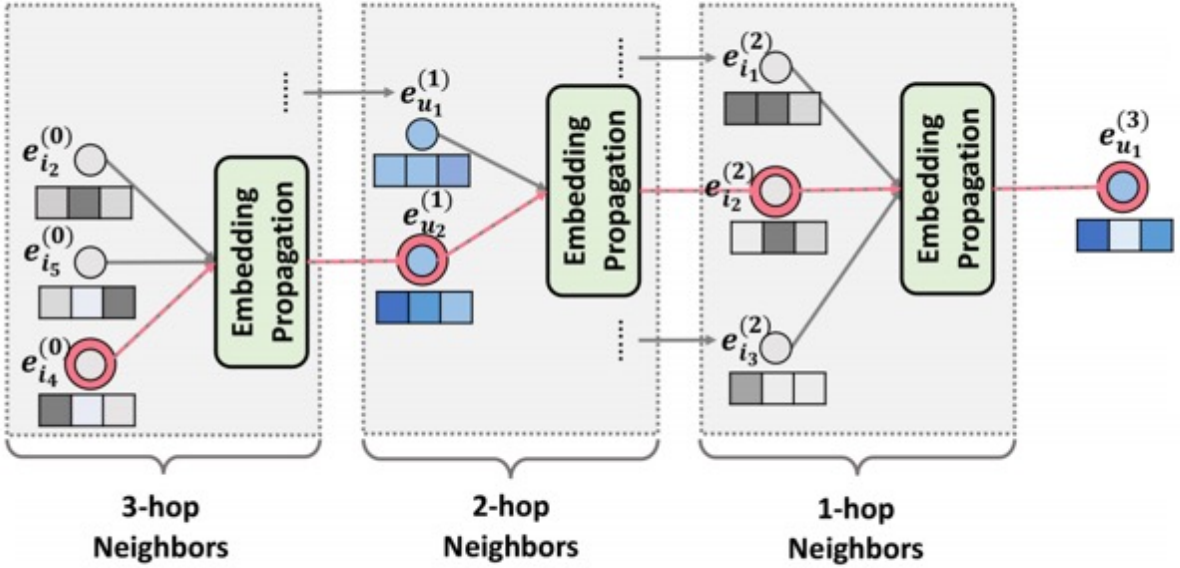
self-connections

all neighbors of u

Wang et al. Neural Graph Collaborative Filtering. SIGIR’2019

Q1: Are GNNs suitable for CF?
Modeling High-Order Connectivity

- Stack more embedding propagation layers to explore the high-order connectivity



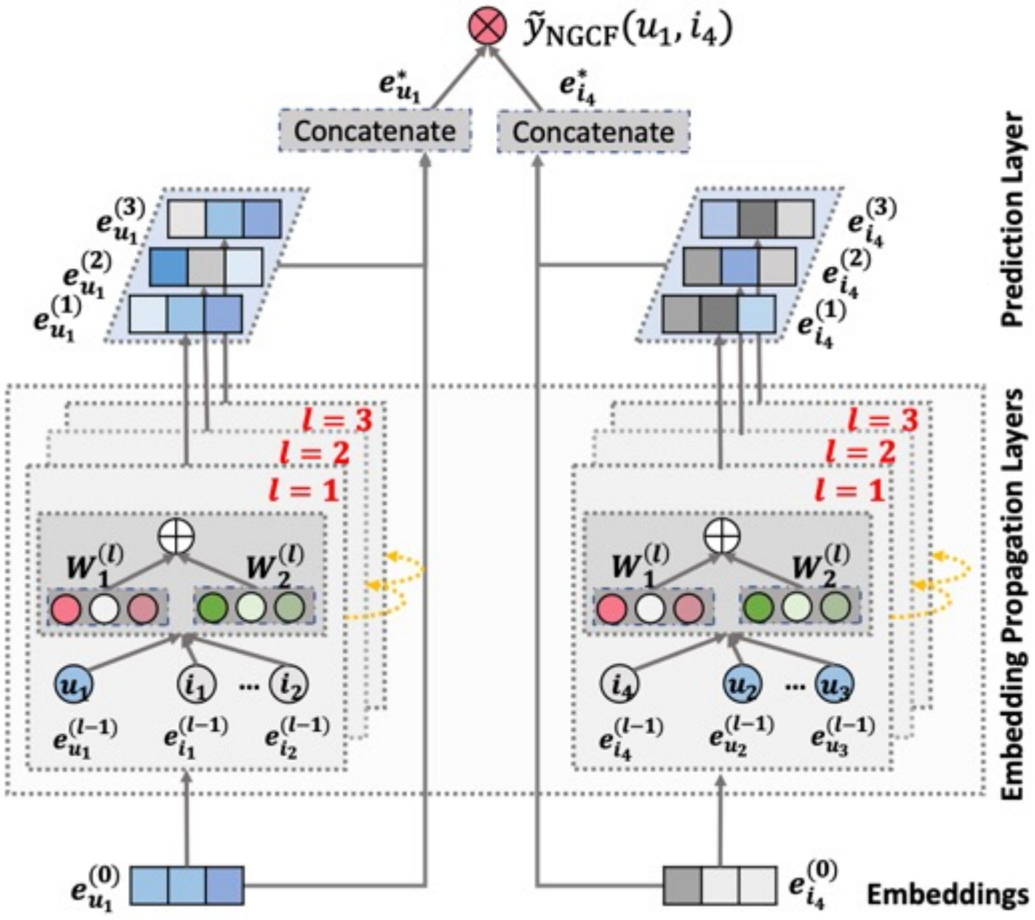
- The collaborative signal like u1 ← i2 ← u2 ← i4 can be captured in the embedding propagation process.

- **Collaborative signal can be injected into the representation learning process.**

Wang et al. Neural Graph Collaborative Filtering. SIGIR'2019

Q1: Are GNNs suitable for CF?

Aggregating Multi-Grained Connectivity



$$\mathbf{e}^*_u = \mathbf{e}^{(0)}_u \| \cdots \| \mathbf{e}^{(L)}_u$$

$$\mathbf{e}^*_i = \mathbf{e}^{(0)}_i \| \cdots \| \mathbf{e}^{(L)}_i$$

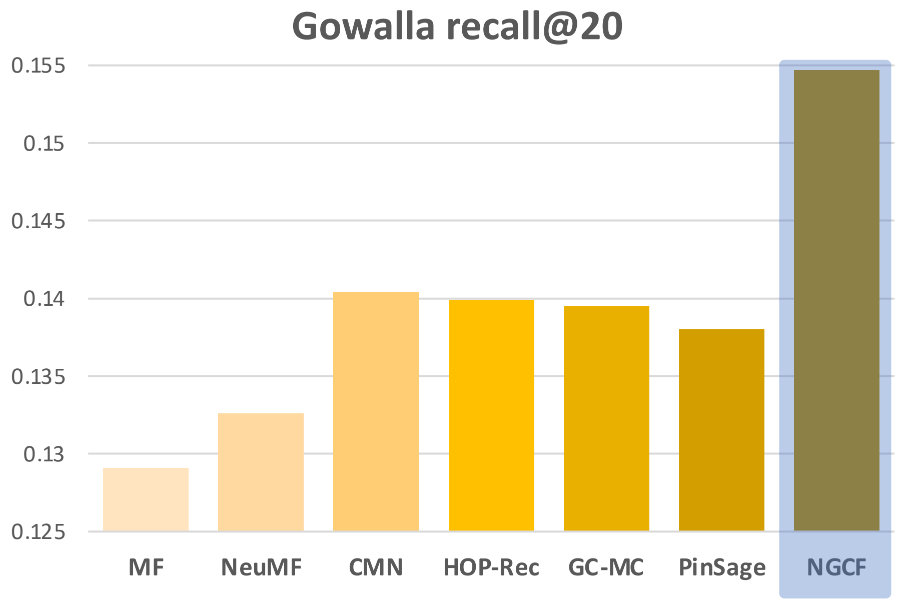
$$\hat{y}_{\mathrm{NGCF}}(u, i) = {\mathbf{e}^*_u}^\top \mathbf{e}^*_i$$

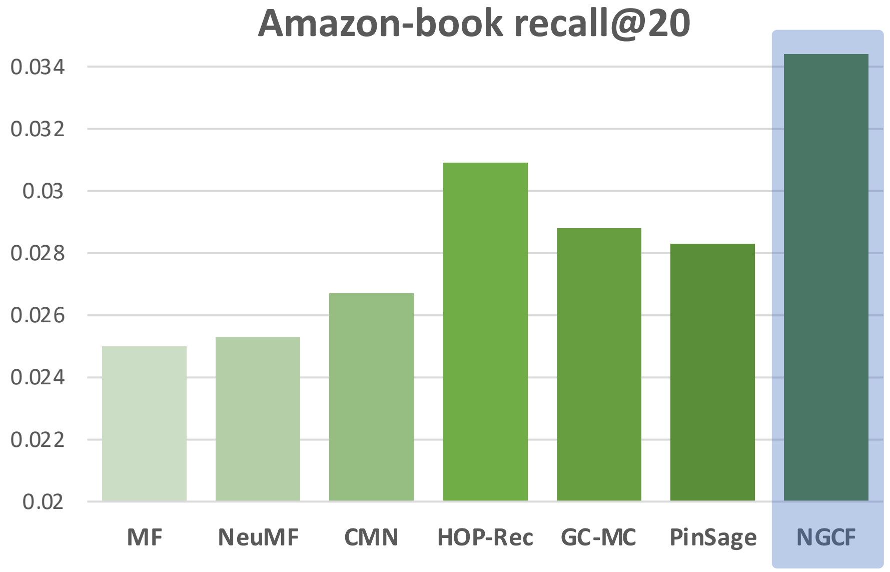
The representations at different layers

- emphasize the messages passed over different connections
- have different contributions in reflecting user preference

Wang et al. Neural Graph Collaborative Filtering. SIGIR’2019

Q1: Are GNNs suitable for CF?
Experimental Results





- NGCF consistently yields the best performance on all the datasets.

- **This verifies the importance of capturing collaborative signal in embedding function.**

Q2: How to tailor GNNs for CF?

Limitations of Directly Applying GNNs on CF

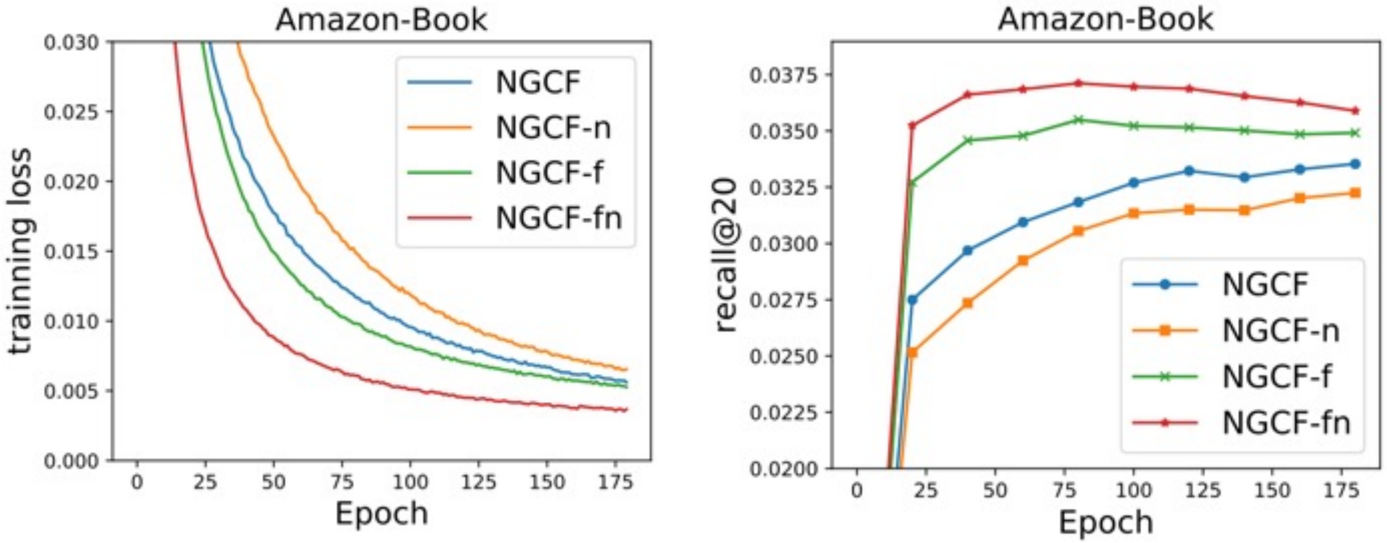
	GNNs	NGCF
Original task	Node classification	Collaborative filtering
Input data	Rich node features • Attributes, text, image data	Only node ID • One-hot encoding
Feature transformation	Distill useful information	Generate ID embeddings
Neighborhood aggregation	Pass messages from neighbors to the egos	Pass messages from neighbors to the egos
Nonlinear activation	Enhance representation ability	Negatively increases the difficulty for model training

He et al. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. SIGIR'2020

Q2: How to tailor GNNs for CF?
Redundant Operations for CF

- Removing feature transformation (NGCF-f)→consistent improvement
- Removing nonlinear activation (NGCF-n) → hurt

- Removing nonlinear activation & feature transformation (NGCF-fn) → significant improvements over NGCF!



He et al. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. SIGIR'2020

Q2: How to tailor GNNs for CF?
Light Graph Convolution

NGCF

- Graph Convolution Layer

$$\mathbf{e}_u^{(l)} = \cancel{\text{LeakyReLU}}\Big(\cancel{\mathbf{m}_{u\leftarrow u}^{(l)}} + \sum_{i\in\mathcal{N}_u} \mathbf{m}_{u\leftarrow i}^{(l)}\Big)$$

- Layer Combination

$$\mathbf{e}_u^{*} = \mathbf{e}_u^{(0)} \| \cdots \| \mathbf{e}_u^{(L)}$$

- Matrix Form

$$\mathbf{E}^{(l)} = \cancel{\text{LeakyReLU}}\Big((\mathcal{L}+\mathbf{I})\mathbf{E}^{(l-1)}\cancel{\mathbf{W}_1^{(l)}} + \mathcal{L}\cancel{\mathbf{E}^{(l-1)}} \odot \mathbf{E}^{(l-1)}\cancel{\mathbf{W}_2^{(l)}}\Big)$$

LightGCN

- **Light** Graph Convolution Layer

$$\mathbf{e}_u^{(k+1)} = \sum_{i\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}} \mathbf{e}_i^{(k)},$$

- Layer Combination

$$\mathbf{e}_u = \sum_{k=0}^{K} \alpha_k \mathbf{e}_u^{(k)}$$

- Matrix Form

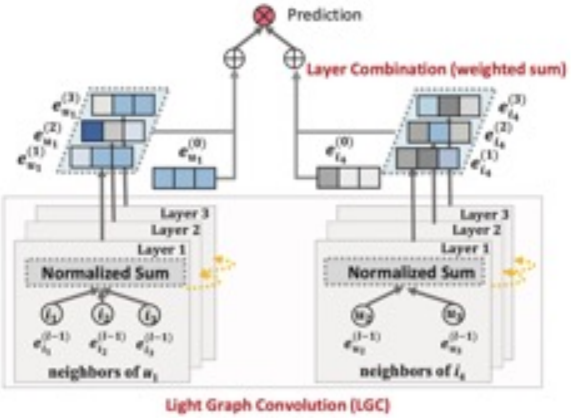
$$\mathbf{E}^{(k+1)} = (\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})\mathbf{E}^{(k)}$$

Only simple weighted sum aggregator is remained

- No feature transformation
- No nonlinear activation
- No self connection

He et al. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. SIGIR’2020

Q2: How to tailor GNNs for CF?
LightGCN



$$\begin{aligned}\mathbf{E} &= \alpha_0\mathbf{E}^{(0)} + \alpha_1\mathbf{E}^{(1)} + \alpha_2\mathbf{E}^{(2)} + \ldots + \alpha_K\mathbf{E}^{(K)} \\ &= \alpha_0\mathbf{E}^{(0)} + \alpha_1\tilde{\mathbf{A}}\mathbf{E}^{(0)} + \alpha_2\tilde{\mathbf{A}}^2\mathbf{E}^{(0)} + \ldots + \alpha_K\tilde{\mathbf{A}}^K\mathbf{E}^{(0)}\end{aligned}$$

importance of the k-th layer embedding in constituting the final embedding

- Relation with **SGC** [2019]:
  - By doing layer combination, LightGCN subsumes the effect of self-connection → **no need to add self-connection in adjacency matrix.**

$$\mathbf{E}^{(K)} = \binom{K}{0}\mathbf{E}^{(0)} + \binom{K}{1}\mathbf{A}\mathbf{E}^{(0)} + \binom{K}{2}\mathbf{A}^2\mathbf{E}^{(0)} + \ldots + \binom{K}{K}\mathbf{A}^K\mathbf{E}^{(0)}.$$

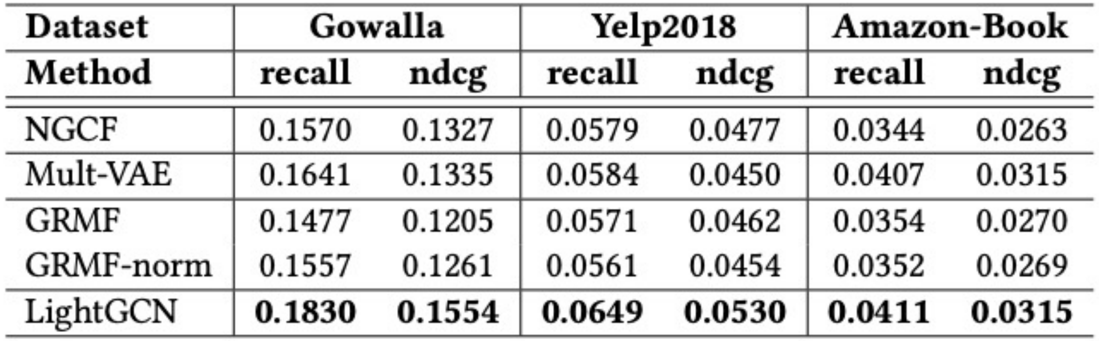
- Relation with **APPNP** [2019]:
  - By setting α_k properly, LightGCN can recover APPNP → **use a large K for long-range modeling with controllable oversmoothing.**

$$\mathbf{E}^{(K)} = \beta\mathbf{E}^{(0)} + \beta(1-\beta)\tilde{\mathbf{A}}\mathbf{E}^{(0)} + \beta(1-\beta)^2\tilde{\mathbf{A}}^2\mathbf{E}^{(0)} + \ldots + (1-\beta)^K\tilde{\mathbf{A}}^K\mathbf{E}^{(0)}.$$

He et al. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. SIGIR'2020

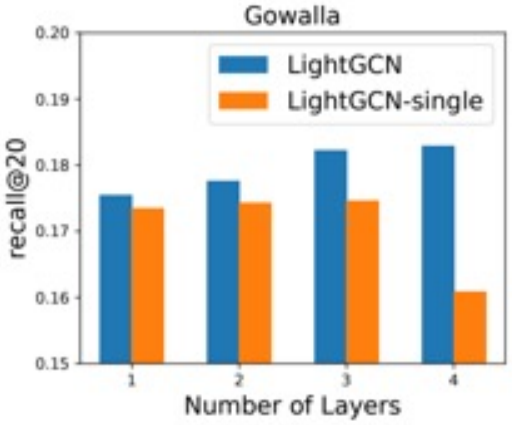
Q2: How to tailor GNNs for CF?
Experimental Results

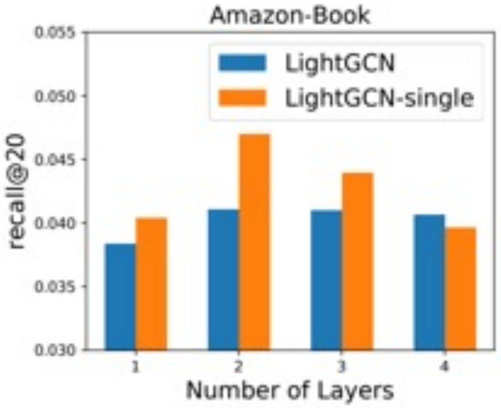
- LightGCN achieves significant improvements over the state-of-the-art baselines → **outstanding performance**



Dataset	Gowalla		Yelp2018		Amazon-Book	
Method	recall	ndcg	recall	ndcg	recall	ndcg
NGCF	0.1570	0.1327	0.0579	0.0477	0.0344	0.0263
Mult-VAE	0.1641	0.1335	0.0584	0.0450	0.0407	0.0315
GRMF	0.1477	0.1205	0.0571	0.0462	0.0354	0.0270
GRMF-norm	0.1557	0.1261	0.0561	0.0454	0.0352	0.0269
LightGCN	**0.1830**	**0.1554**	**0.0649**	**0.0530**	**0.0411**	**0.0315**

- LightGCN-single (only uses the final layer's output) performs better than LightGCN on sparser datasets → **can be further simplified**.





Q3: How to do self-supervised learning?
Common Issues in Recommendation

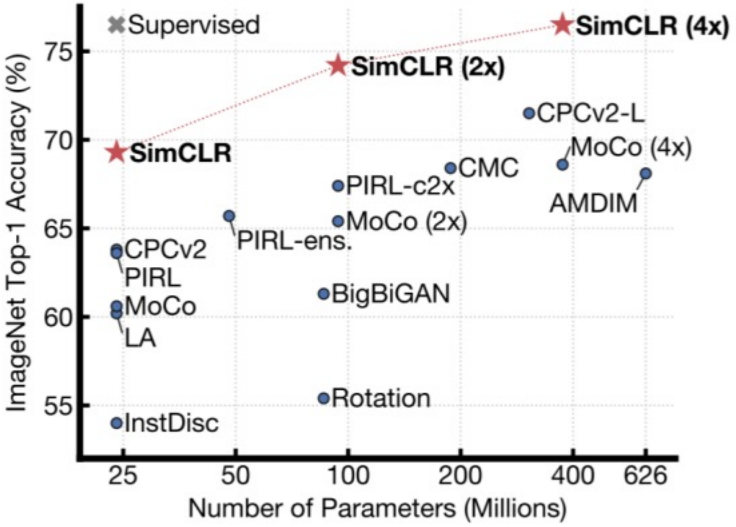
- **Sparse Supervision Signal**
  - The observed interactions → extremely sparse (e.g., sparsity ≈ 99%)

- **Skewed Data Distribution**
  - Power-law distribution
  - High-degree items exert larger impact on the representation learning

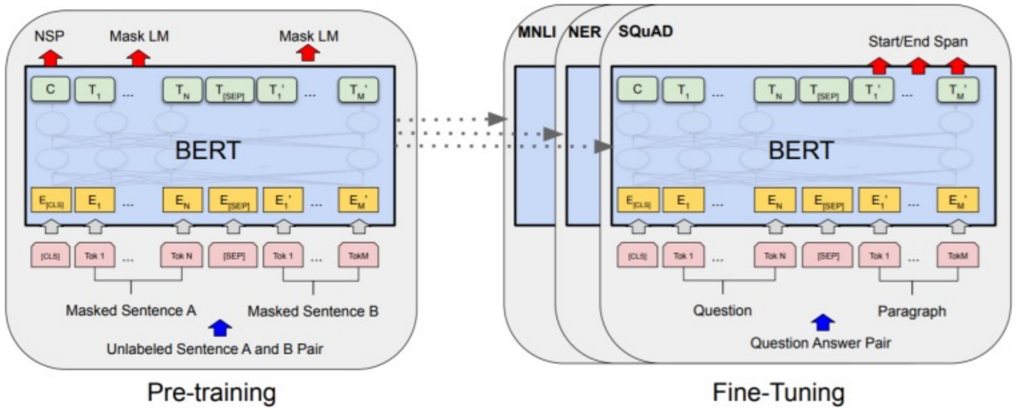
- **Noises in Interactions**
  - Implicit feedback makes the learning more vulnerable to interaction noises

Q3: How to do self-supervised learning?

Self-supervised Contrastive Learning



CV: MoCo, SimCLR



Pre-training

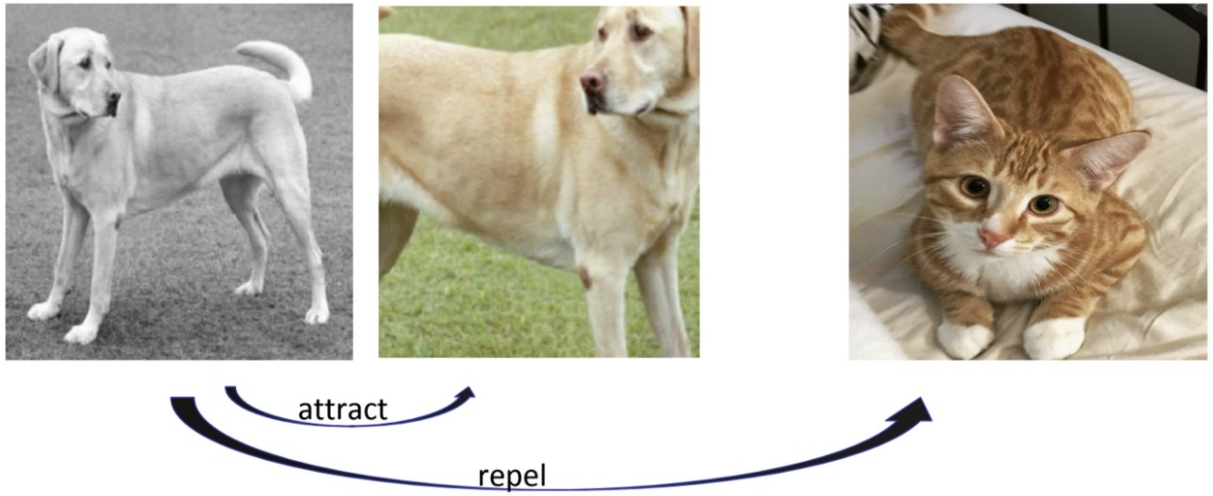
Fine-Tuning

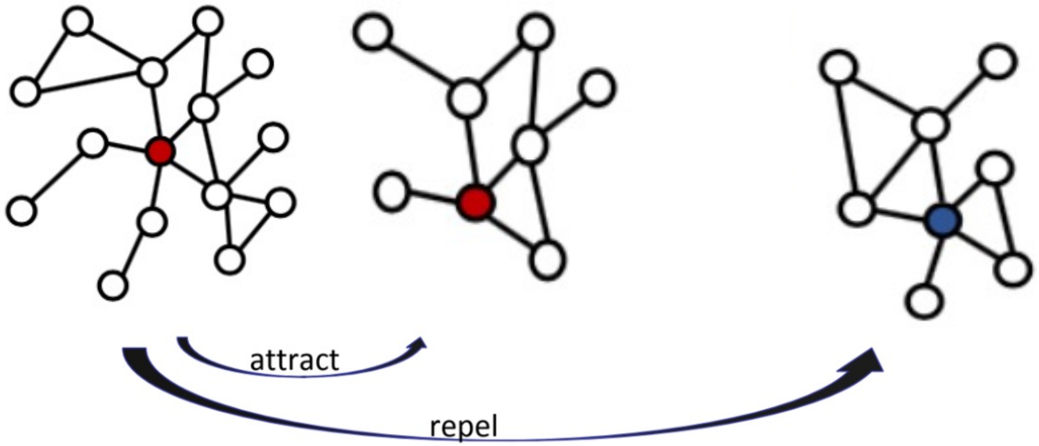
NLP: BERT

Basic Idea:

1. Create **auxiliary pre-text task** for the model **from the input data itself**
2. Learn the “**extra supervision signal**” from the data
3. **Pre-train** the model on the pre-text task
4. **Fine-tune** the model on the main task of interest

Q3: How to do self-supervised learning?
Graph Contrastive Learning





Pre-text task: Image self-discrimination

1. **Positive instances**
   - Two augmented versions of the **same** image
2. **Negative instances**
   - Two augmented versions of **different** images
3. **Contrastive Learning**
   - **Maximize the agreement** of positives, as compared to that of negatives

Pre-text task: Graph Self-discrimination

1. **Positive instances**
   - Two augmented versions of the **same** graph
2. **Negative instances**
   - Two augmented versions of **different** graphs
3. **Contrastive Learning**
   - **Maximize the agreement** of positives, as compared to that of negatives

Q3: How to do self-supervised learning?

Graph Augmentation for CF

$$Z_1^{(l)} = H\left(Z_1^{(l-1)}, s_1(\mathrm{G})\right), \quad Z_2^{(l)} = H\left(Z_2^{(l-1)}, s_2(\mathrm{G})\right), \quad s_1, s_2 \sim \mathrm{S}$$

- **Node Dropout (ND)**

$$s_1(\mathrm{G}) = (M^{'} \odot \mathrm{V}, \mathrm{E}), \quad s_2(\mathrm{G}) = (M^{"} \odot \mathrm{V}, \mathrm{E}) \quad M^{'}, M^{"} \in \{0,1\}^{|\mathrm{V}|}$$

  - **Identify the influential nodes** from differently augmented views

- **Edge Dropout (ED)**

$$s_1(\mathrm{G}) = (\mathrm{V}, M^{'} \odot \mathrm{E}), \quad s_2(\mathrm{G}) = (\mathrm{V}, M^{"} \odot \mathrm{E}) \quad M^{'}, M^{"} \in \{0,1\}^{|\mathrm{E}|}$$

  - Capture the useful patterns of the **local structures of a node**

- **Random Walk (RW)**

$$s_1(\mathrm{G}) = (\mathrm{V}, M_1^{(l)} \odot \mathrm{E}), \quad s_2(\mathrm{G}) = (\mathrm{V}, M_2^{(l)} \odot \mathrm{E}) \quad M_1^{(l)}, M_2^{(l)} \in \{0,1\}^{|\mathrm{E}|}$$

  - Layer-sensitive local structure

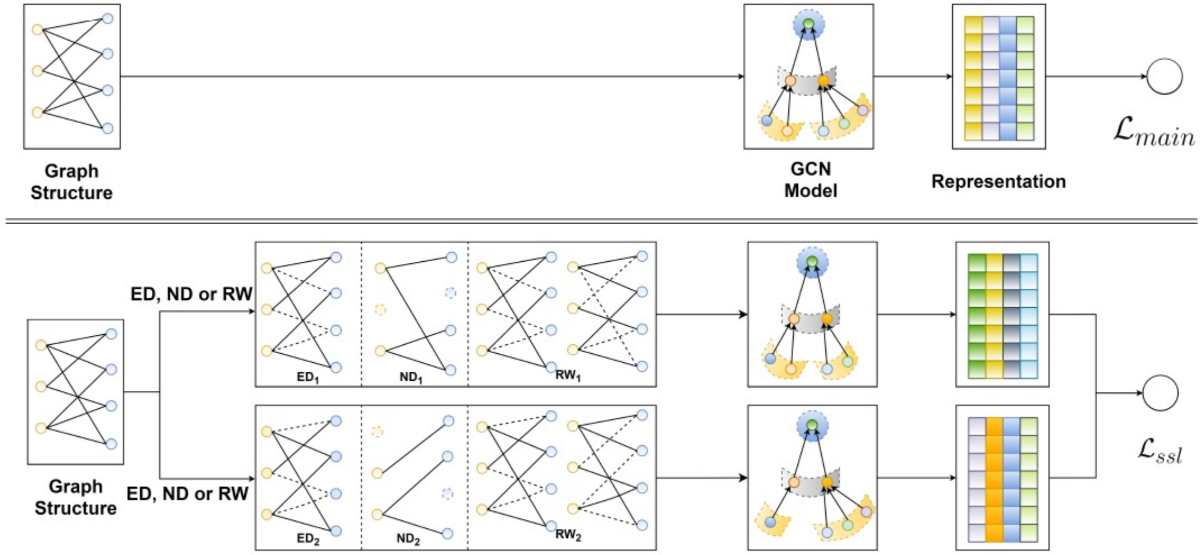
Wu et al. Self-supervised Graph Learning for Recommendation. SIGIR’2021

Q3: How to do self-supervised learning?
Self-supervised Graph Learning (SGL)

- **Contrastive Loss --- InfoNCE**
  - maximize the agreement of positive pairs
  - minimize that of negative pairs

$$\mathcal{L}^{user}_{ssl} = \sum_{u\in\mathcal{U}} -\log \frac{\exp(s(\mathbf{z}'_u,\mathbf{z}''_u)/\tau)}{\sum_{v\in\mathcal{U}}\exp(s(\mathbf{z}'_u,\mathbf{z}''_v)/\tau)}$$

$$\mathcal{L}_{ssl} = \mathcal{L}^{user}_{ssl} + \mathcal{L}^{item}_{ssl}$$



- **Supervised Loss --- BPR**

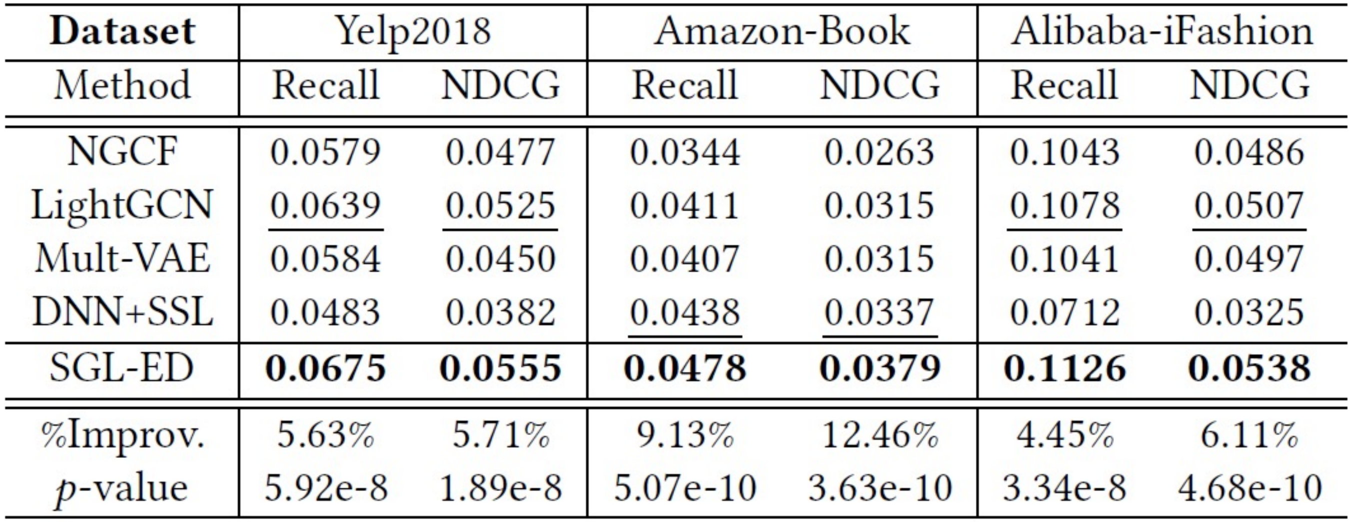
$$L_{main} = \sum_{(u,i,j)\in O} -\log\sigma\left(\hat{y}_{ui}-\hat{y}_{uj}\right)$$

- **Pre-training/Fine-Tuning & Multi-task Training**

$$L = L_{main} + \lambda_1 L_{ssl} + \lambda_2 \|\Theta\|_2^2$$

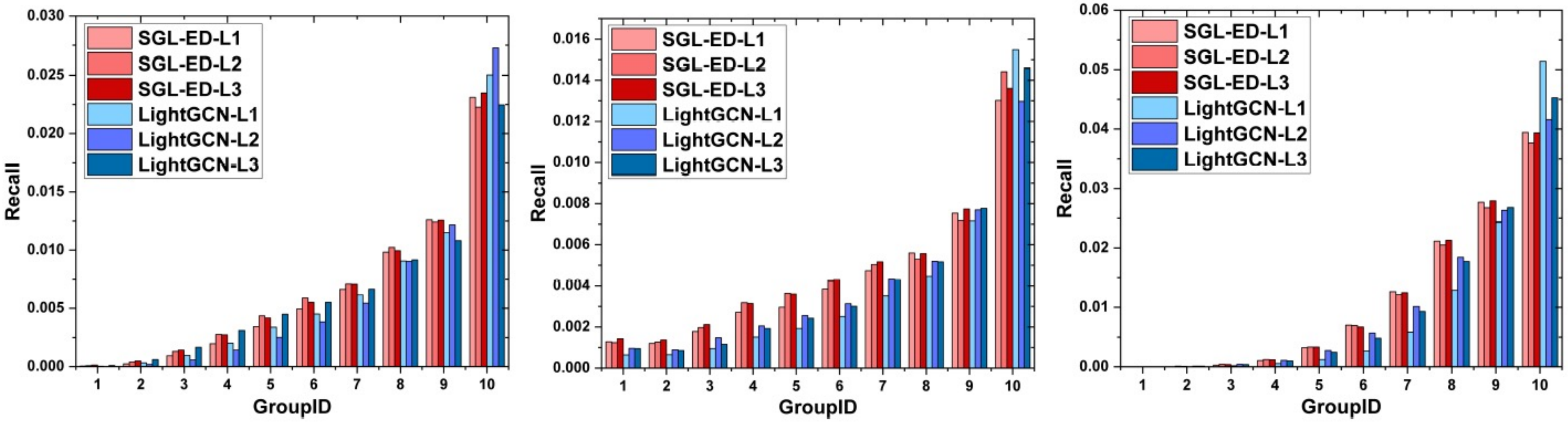
Wu et al. Self-supervised Graph Learning for Recommendation. SIGIR’2021

Q3: How to do self-supervised learning?
Experimental Results



Dataset	Yelp2018		Amazon-Book		Alibaba-iFashion	
Method	Recall	NDCG	Recall	NDCG	Recall	NDCG
NGCF	0.0579	0.0477	0.0344	0.0263	0.1043	0.0486
LightGCN	0.0639	0.0525	0.0411	0.0315	0.1078	0.0507
Mult-VAE	0.0584	0.0450	0.0407	0.0315	0.1041	0.0497
DNN+SSL	0.0483	0.0382	0.0438	0.0337	0.0712	0.0325
SGL-ED	**0.0675**	**0.0555**	**0.0478**	**0.0379**	**0.1126**	**0.0538**
%Improv.	5.63%	5.71%	9.13%	12.46%	4.45%	6.11%
p-value	5.92e-8	1.89e-8	5.07e-10	3.63e-10	3.34e-8	4.68e-10

✓ SGL achieves significant improvements over the state-of-the-art baselines → **outstanding performance**



(a) Yelp2018 (b) Amazon-Book (c) Alibaba-iFashion

Wu et al. Self-supervised Graph Learning for Recommendation. SIGIR’2021

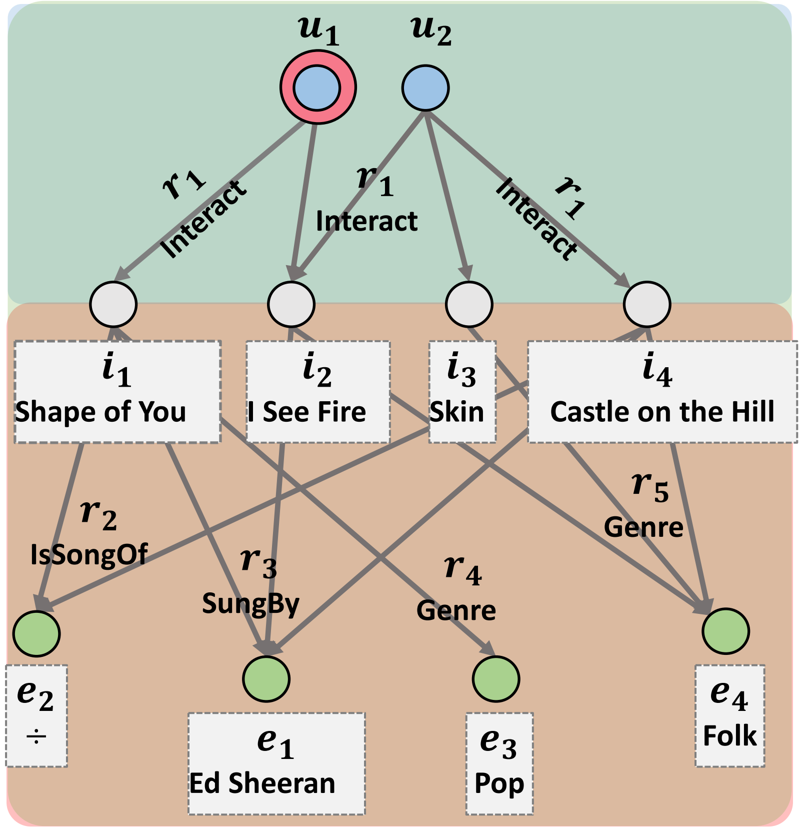
Outline

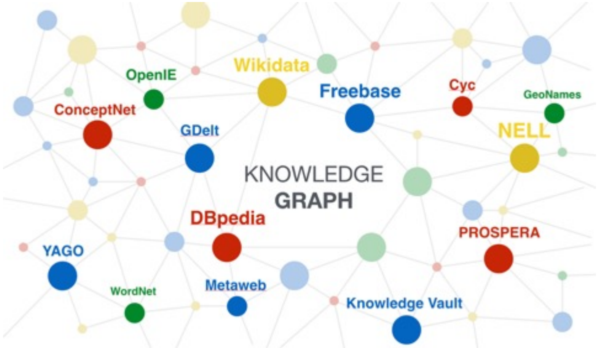
- GNN for Collaborative Filtering
  - Are GNNs suitable for CF?
    - NGCF (SIGIR’2019)
  - How to tailor GNNs for CF?
    - LightGCN (SIGIR’2020)
  - How to inject self-supervised learning into GNN-based CF?
    - SGL (SIGIR’2021)

- GNN for Knowledge Graph-based Recommendation
  - Q1: Are GNNs suitable for KG-based Rec?
    - KGAT (KDD’2019)
  - Q2: How to tailor GNNs for KG-based Rec?
    - KGIN (WWW’2021)

Q1: Are GNNs Suitable for KG-aware Rec?

CF + KG





User-Item Bipartite Graph

- User-Item Direct Interactions

$$u_1 \xrightarrow{r_1} i_1$$

+

Knowledge Graph (KG)

- Item-Item External Connections

$$i_1 \xrightarrow{r_2} e_1$$

- Background knowledge on items
- Rich semantics & Relations

=

Collaborative Knowledge Graph

- High-order connectivity between users and items

$$u_1 \xrightarrow{r_1} i_1 \xrightarrow{r_2} e_1 \xrightarrow{-r_2} i_2 \Rightarrow u_1 \xrightarrow{r_1} i_2$$

- Narrow down search space
- Explore user interests reasonably
- Offer explanations

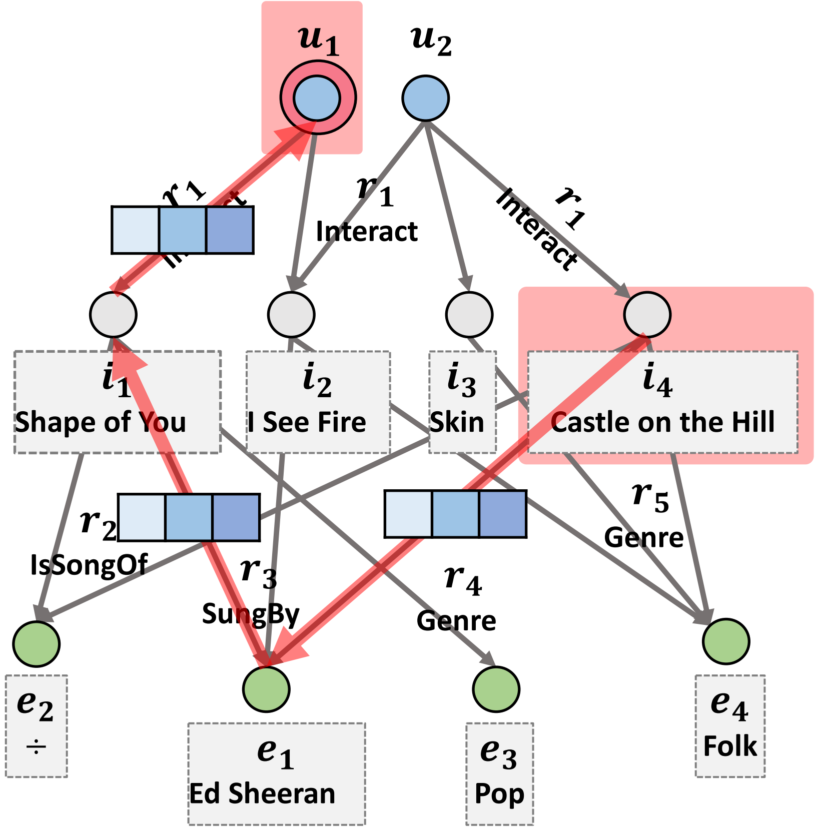
Q1: Are GNNs Suitable for KG-aware Rec?
Limitations of Prior Studies

Existing works suffer from the limited model capacity, due to the **suboptimal modeling of high-order & attributed CF signals.**

	Supervised Learning-based	Path-based	Regularization-based
Knowledge Usage	Item knowledge → a generic feature vector	Connectivity → paths connecting users & items	Graph structure → an additional item representations or loss
Relation Usage	-	To define meta-path Or select qualified paths	To regularize the learning of KG embeddings
Limitations	• Fail to capture CF signals • Ignore semantic & structure information	• Require labor-intensive feature engineering • Have rather high complexity	• Lack explicit modeling of high-order relations

Q1: Are GNNs Suitable for KG-aware Rec?

Attentive Embedding Propagation



Attentive Embedding Propagation, inspired by GNNs

- Propagate embeddings recursively on the graph
- Reveal the importance of a high-order connectivity via relation-aware attentions
- Construct information flows in the embedding space

Wang et al. Kgat: Knowledge graph attention network for recommendation. KDD’2019

Q1: Are GNNs Suitable for KG-aware Rec?

Knowledge-aware Attention

KGAT: Information Propagation

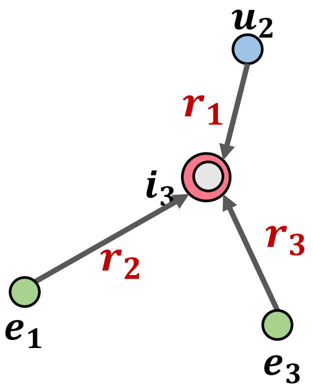
- **Information Aggregation**

The messages accounting for first-order connectivity

The set of triples, where the target node is the head entity

$$\mathbf{e}_{\mathcal{N}_h} = \sum_{(h,r,t)\in\mathcal{N}_h} \pi(h,r,t)\,\mathbf{e}_t$$

Tail representation



- **Knowledge-aware Attention**

decay factor on each propagation

$$\pi(h,r,t) = (\mathbf{W}_r\mathbf{e}_t)^\top \tanh\Big((\mathbf{W}_r\mathbf{e}_h + \mathbf{e}_r)\Big)$$

the attention score is dependent on the distance of e_t and e_h in r's space

- **Representation Update**

$$f_{\text{Bi-Interaction}} = \text{LeakyReLU}\Big(\mathbf{W}_1(\mathbf{e}_h + \mathbf{e}_{\mathcal{N}_h})\Big) + \text{LeakyReLU}\Big(\mathbf{W}_2(\mathbf{e}_h \odot \mathbf{e}_{\mathcal{N}_h})\Big),$$

Similar to NGCF

Wang et al. Kgat: Knowledge graph attention network for recommendation. KDD’2019

Q1: Are GNNs Suitable for KG-aware Rec?
KGAT Framework

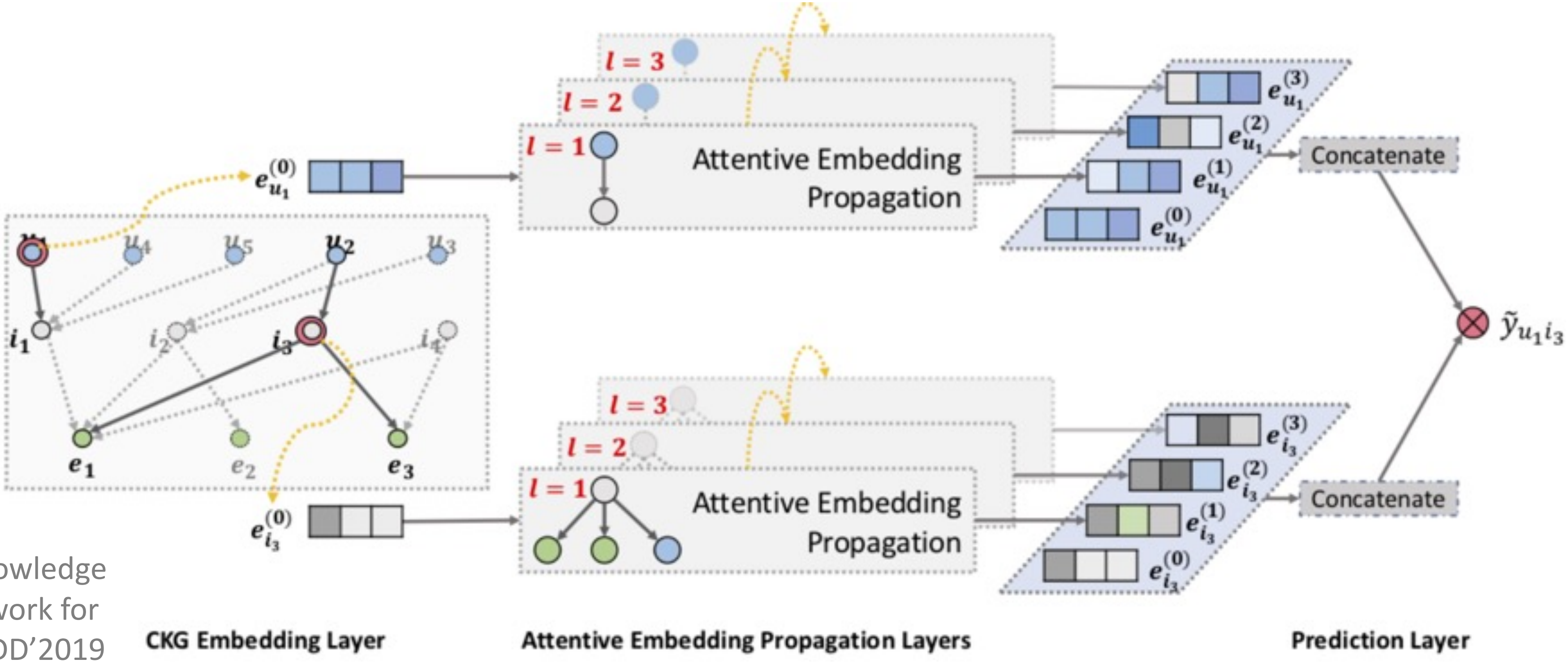
$$\mathbf{e}_u^* = \boxed{\mathbf{e}_u^{(0)} \| \cdots \| \mathbf{e}_u^{(L)}}$$

$$\mathbf{e}_i^* = \mathbf{e}_i^{(0)} \| \cdots \| \mathbf{e}_i^{(L)}$$

$$\hat{y}(u, i) = {\mathbf{e}_u^*}^\top \mathbf{e}_i^*$$

Similar to NGCF, the representations at different layers

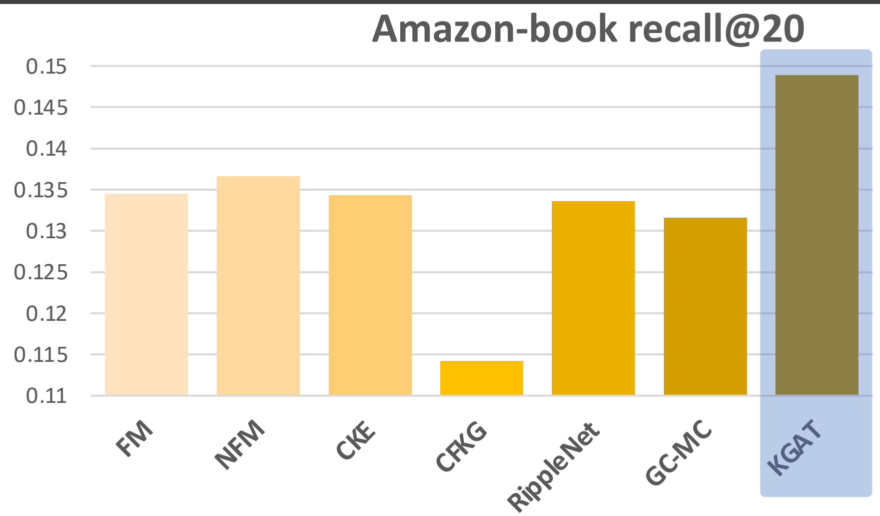
- emphasize the messages passed over different connections
- have different contributions in reflecting user preference



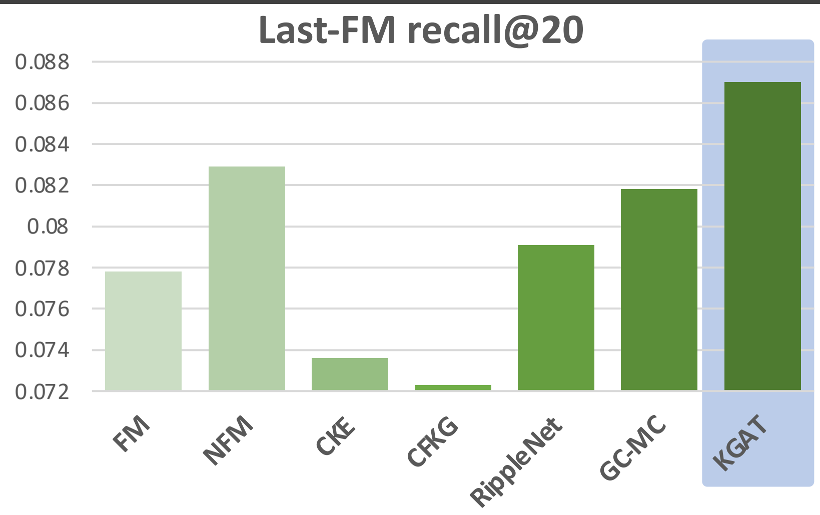
Wang et al. Kgat: Knowledge graph attention network for recommendation. KDD'2019

Q1: Are GNNs Suitable for KG-aware Rec?

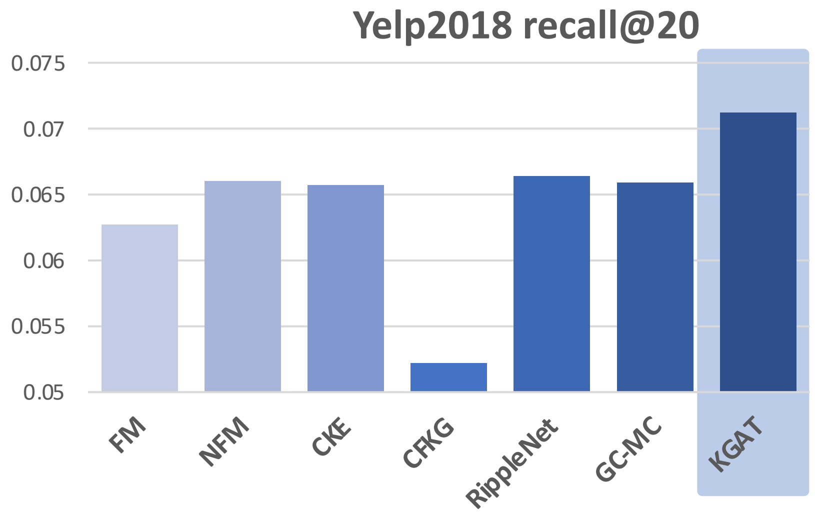
Experimental Results



Amazon-book recall@20



Last-FM recall@20



Yelp2018 recall@20

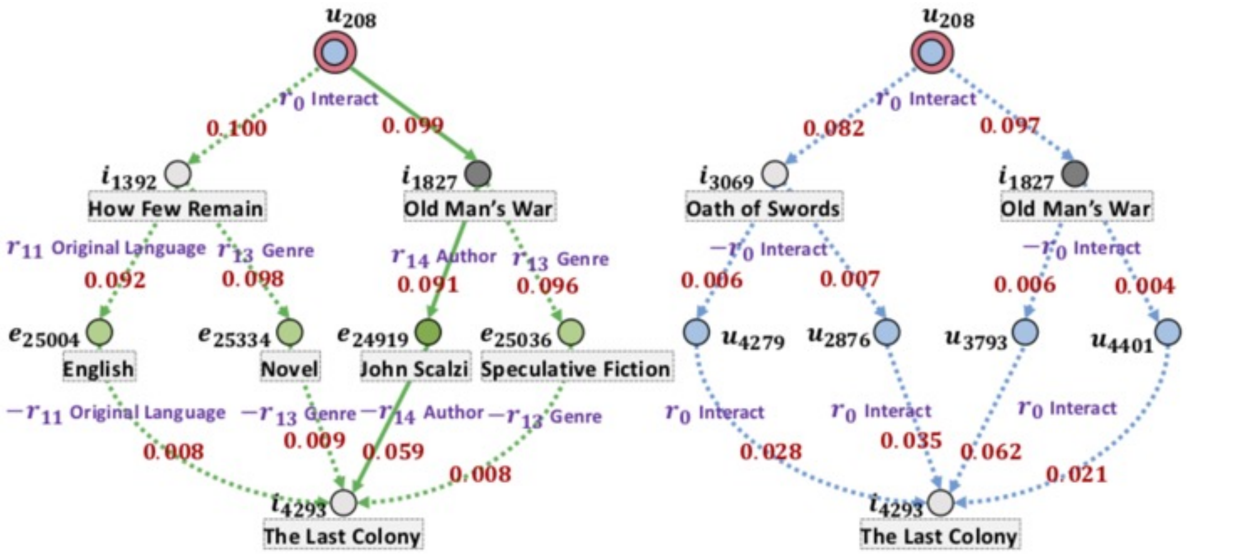


Figure 4: Real Example from Amazon-Book.

$$u_{208} \xrightarrow{r_0} \textit{Old Man's War} \xrightarrow{r_{14}} \textit{John Scalzi} \xrightarrow{-r_{14}} i_{4293}$$

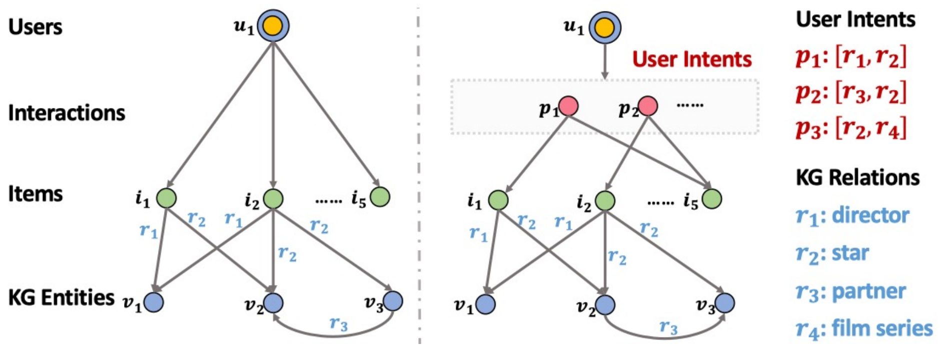
Wang et al. Kgat: Knowledge graph attention network for recommendation. KDD'2019

Q2: How to Tailor GNNs for KG-aware Rec?

Limitation on User Intents

None considers user-item relations at a finer-grained level of intents:

- They only model one single relation between users & items, however, a user generally has multiple intents to adopt items



- “director” & “star” → watch i_1 & i_5
- “star” & “partner” → watch i_2

Basic idea: Similar users have similar preferences on items.

However: Obscure intents would **confound** the modeling of users’ behavioral similarity

Our idea: Conditioning on similar intents, similar users have similar preferences on items.

Wang et al. Learning Intents behind Interactions with Knowledge Graph for Recommendation. WWW’2021

Q2: How to Tailor GNNs for KG-aware Rec?

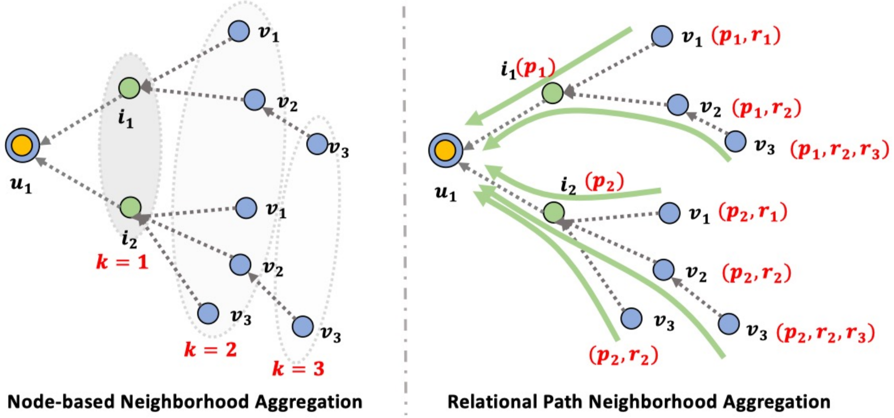
Limitation on Relational Paths

Information aggregation schemes are mostly node-based:

- They only collect information from neighboring nodes, without differentiating which paths it comes from.

Node-based

- 1-hop: $\{i_1, i_2\}$
- 2-hop: $\{v_1, v_2, v_3\}$
- 3-hop: $\{v_3\}$



Path-based

- Relation dependencies (p_1, r_2, r_3) between v_1 & v_3

Basic idea: Node-based aggregation mixes information of neighborhoods.

However: It fails to preserve the **relation dependencies & sequencies** carried by paths → **Relational paths**

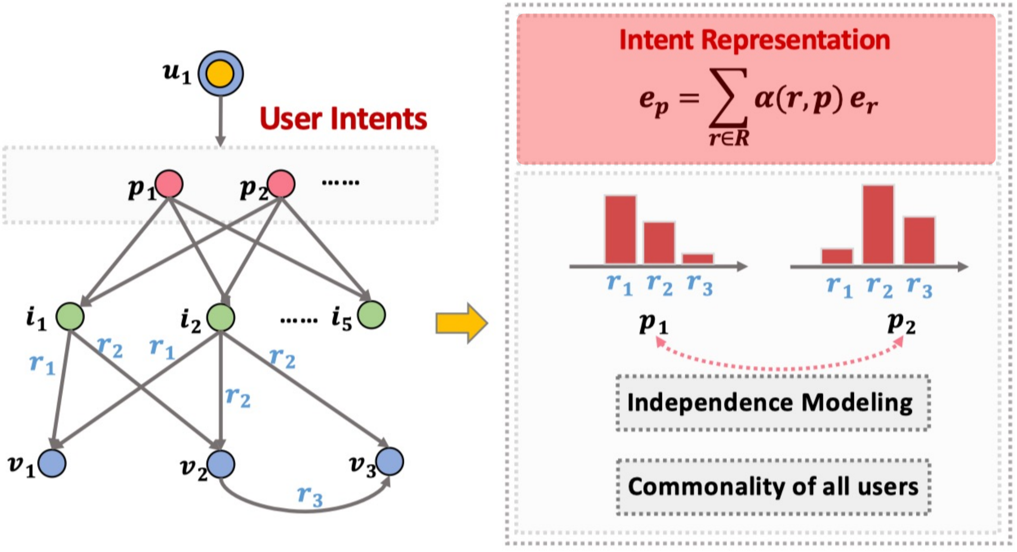
Our idea: Treating relational paths as **an information channel** to conduct information propagation.

Wang et al. Learning Intents behind Interactions with Knowledge Graph for Recommendation. WWW'2021

Q2: How to Tailor GNNs for KG-aware Rec?
Modeling of User Intents

Step 1. Representation Learning of Intents

- **Motivation**: Semantics of user intents can be expressed by KG relations.
- **Idea**: assign each intent with a distribution over KG relations → Use attention strategy to create intent embedding



User Intent Modeling

Intent embedding shared by all users

$$\mathbf{e}_p = \sum_{r \in \mathcal{R}} \alpha(r,p)\mathbf{e}_r,$$

Attentive combination over KG relation embeddings

$$\alpha(r,p) = \frac{\exp(w_{rp})}{\sum_{r' \in \mathcal{R}} \exp(w_{r'p})},$$

Quantify importance of relation v_3 to intent p

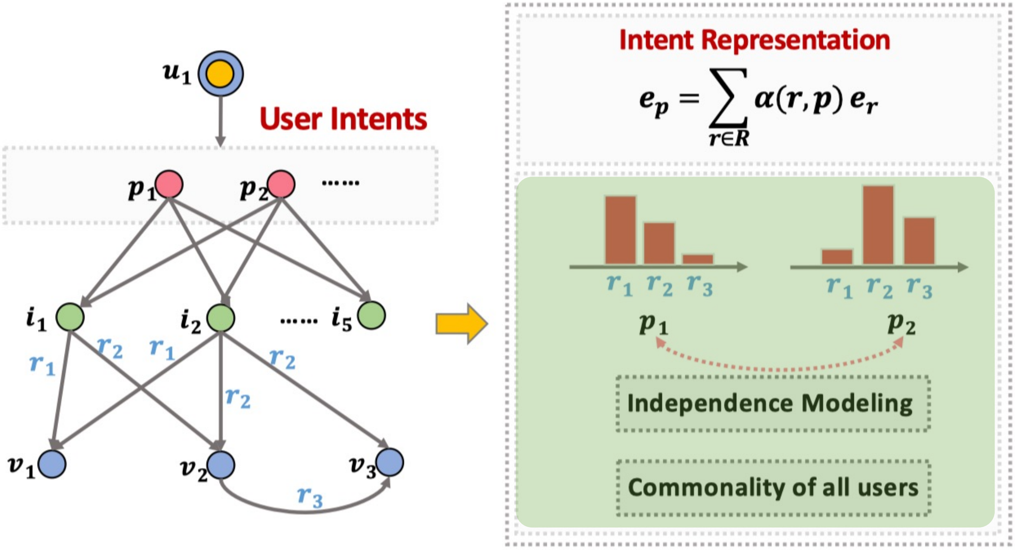
Wang et al. Learning Intents behind Interactions with Knowledge Graph for Recommendation. WWW'2021

Q2: How to Tailor GNNs for KG-aware Rec?

Modeling of User Intents

Step 2. Independence Modeling of Intents

- **Motivation**: Different intents should contain **different & unique** information.
- **Idea**: encourage the representations of intents to differ from each others → **Add independence regularization to intent embeddings**



User Intent Modeling

- **Mutual Information**

$$\mathcal{L}_{\text{IND}} = \sum_{p \in \mathcal{P}} -\log \frac{\exp\left(s(\mathbf{e}_p, \mathbf{e}_p)/\tau\right)}{\sum_{p' \in \mathcal{P}} \exp\left(s(\mathbf{e}_p, \mathbf{e}_{p'})/\tau\right)},$$

Minimize the information amount between any two different intents.

- **Distance Correlation**

$$\mathcal{L}_{\text{IND}} = \sum_{p,p' \in \mathcal{P},\ p \neq p'} dCor(\mathbf{e}_p, \mathbf{e}_{p'}),$$

Minimize the associations of any two different intents.

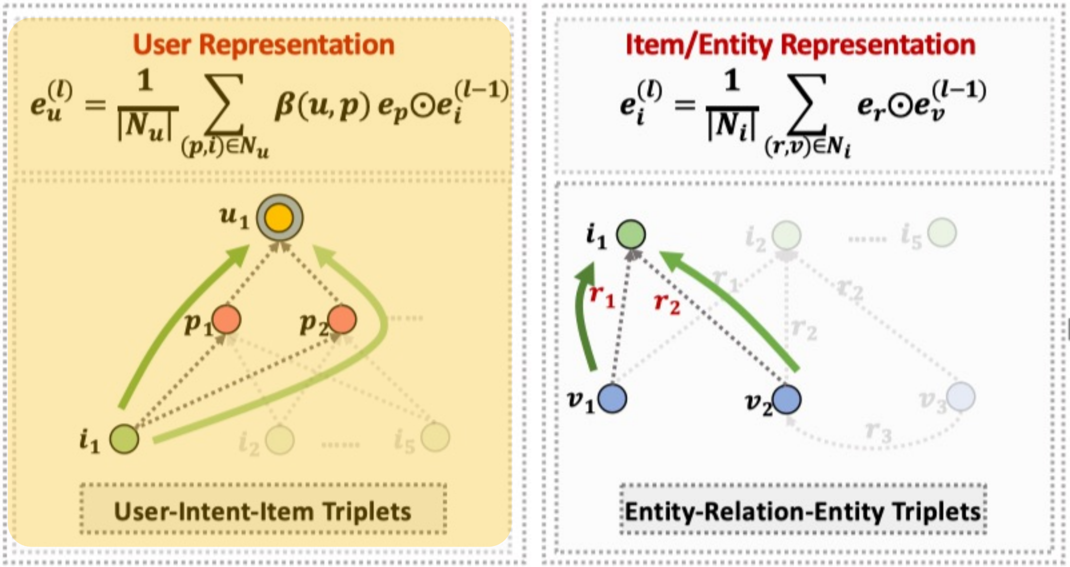
Wang et al. Learning Intents behind Interactions with Knowledge Graph for Recommendation. WWW'2021

Q2: How to Tailor GNNs for KG-aware Rec?

Modeling of Relational Paths

Step 1. Aggregation over Intent Graph (IG)

- **Motivation**: IG contains rich collaborative information of users.
- **Idea**: users with similar intents would exhibit similar preference towards items → **Intent-aware aggregation for user-intent-item triplet** (u, p, i)



User Representation
$e_u^{(l)} = \frac{1}{|N_u|} \sum_{(p,i)\in N_u} \beta(u,p)\, e_p \odot e_i^{(l-1)}$
u_1
p_1
p_2
i_1
i_2
i_5
User-Intent-Item Triplets
Item/Entity Representation
$e_i^{(l)} = \frac{1}{|N_i|} \sum_{(r,v)\in N_i} e_r \odot e_v^{(l-1)}$
i_1
i_2
i_5
r_1
r_2
r_3
v_1
v_2
v_3
Entity-Relation-Entity Triplets

Element-wise product between intent p & historical item i.

$$\mathbf{e}_u^{(1)} = \frac{1}{|\mathcal{N}_u|} \sum_{(p,i)\in\mathcal{N}_u} \beta(u,p)\mathbf{e}_p \odot \mathbf{e}_i^{(0)},$$

$$\beta(u,p) = \frac{\exp(\mathbf{e}_p^\top \mathbf{e}_u^{(0)})}{\sum_{p'\in\mathcal{P}} \exp(\mathbf{e}_{p'}^\top \mathbf{e}_u^{(0)})}$$

Generate user-specific intent representations

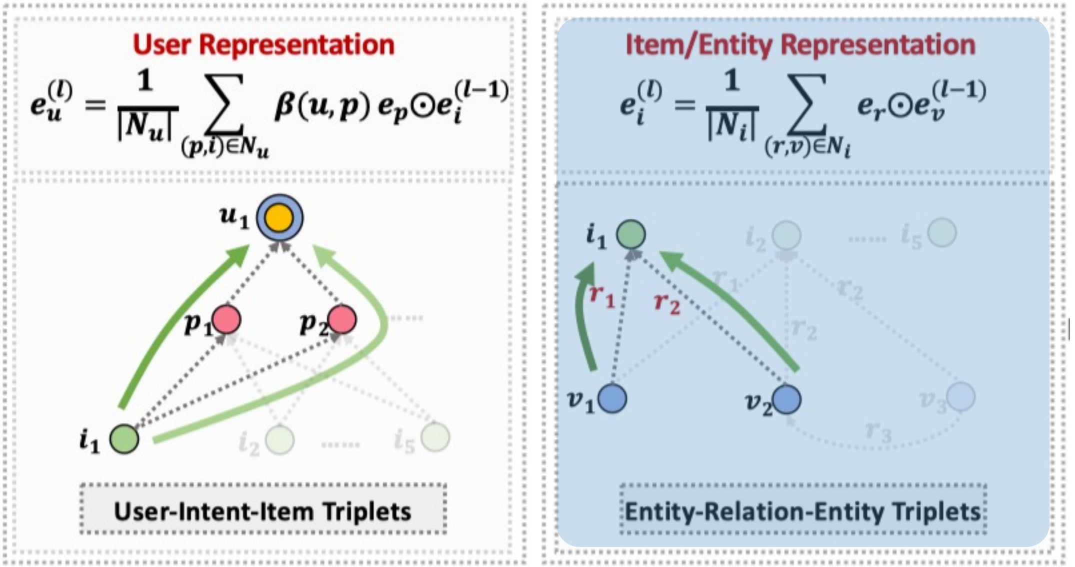
Wang et al. Learning Intents behind Interactions with Knowledge Graph for Recommendation. WWW'2021

Q2: How to Tailor GNNs for KG-aware Rec?

Modeling of Relational Paths

Step 2. Aggregation over Knowledge Graph

- **Motivation**: KG reflects content relatedness among items.
- **Idea**: each KG entity has different semantics in different relational contexts → **Relation-aware aggregation for item-relation-entity triplet** (i, r, v)


User Representation
$e_u^{(l)} = \frac{1}{|N_u|} \sum_{(p,i)\in N_u} \beta(u,p)\, e_p \odot e_i^{(l-1)}$
u_1
p_1
p_2
i_1
i_2
i_5
User-Intent-Item Triplets
Item/Entity Representation
$e_i^{(l)} = \frac{1}{|N_i|} \sum_{(r,v)\in N_i} e_r \odot e_v^{(l-1)}$
i_1
i_2
i_5
r_1
r_2
r_3
v_1
v_2
v_3
Entity-Relation-Entity Triplets

Element-wise product between relation r & connected entity v.

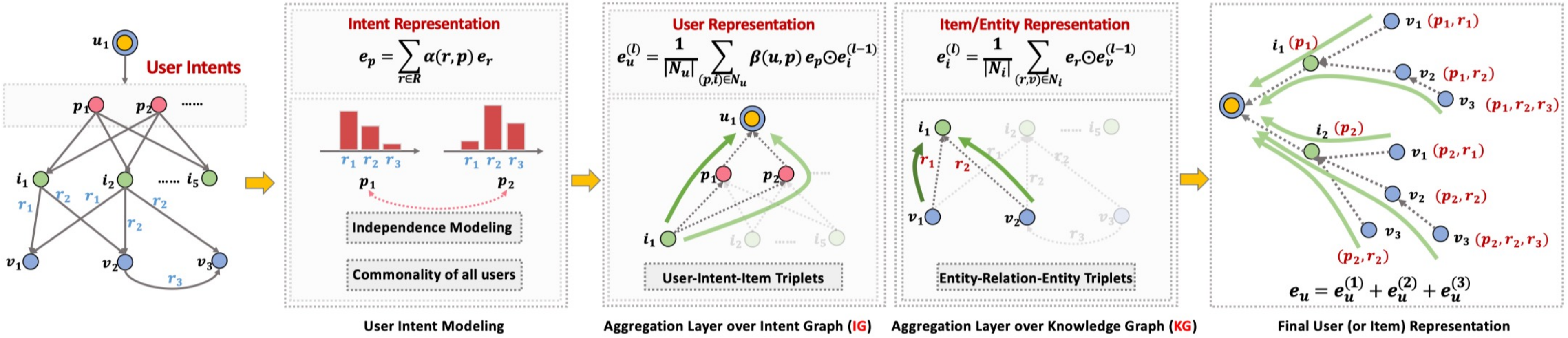
$$\mathbf{e}_i^{(1)} = \frac{1}{|\mathcal{N}_i|} \sum_{(r,v)\in\mathcal{N}_i} \mathbf{e}_r \odot \mathbf{e}_v^{(0)},$$

Wang et al. Learning Intents behind Interactions with Knowledge Graph for Recommendation. WWW'2021

Q2: How to Tailor GNNs for KG-aware Rec?

KGIN Framework

Knowledge Graph-based Intent Network (KGIN)



u_1
User Intents
p_1 p_2 ......
i_1 i_2 ...... i_5
r_1 r_2 r_3
v_1 v_2 v_3
Intent Representation
$e_p = \sum_{r \in R} \alpha(r,p)\, e_r$
r_1 r_2 r_3
p_1 p_2
Independence Modeling
Commonality of all users
User Intent Modeling
User Representation
$e_u^{(l)} = \frac{1}{|N_u|} \sum_{(p,i)\in N_u} \beta(u,p)\, e_p \odot e_i^{(l-1)}$
User-Intent-Item Triplets
Aggregation Layer over Intent Graph (IG)
Item/Entity Representation
$e_i^{(l)} = \frac{1}{|N_i|} \sum_{(r,v)\in N_i} e_r \odot e_v^{(l-1)}$
Entity-Relation-Entity Triplets
Aggregation Layer over Knowledge Graph (KG)
$i_1\ (p_1)$
$v_1\ (p_1, r_1)$
$v_2\ (p_1, r_2)$
$v_3\ (p_1, r_2, r_3)$
$i_2\ (p_2)$
$v_1\ (p_2, r_1)$
$v_2\ (p_2, r_2)$
v_3
(p_2, r_2)
$v_3\ (p_2, r_2, r_3)$
$e_u = e_u^{(1)} + e_u^{(2)} + e_u^{(3)}$
Final User (or Item) Representation

Representation of item, which memorizes the relational signals **carried by the relational paths**

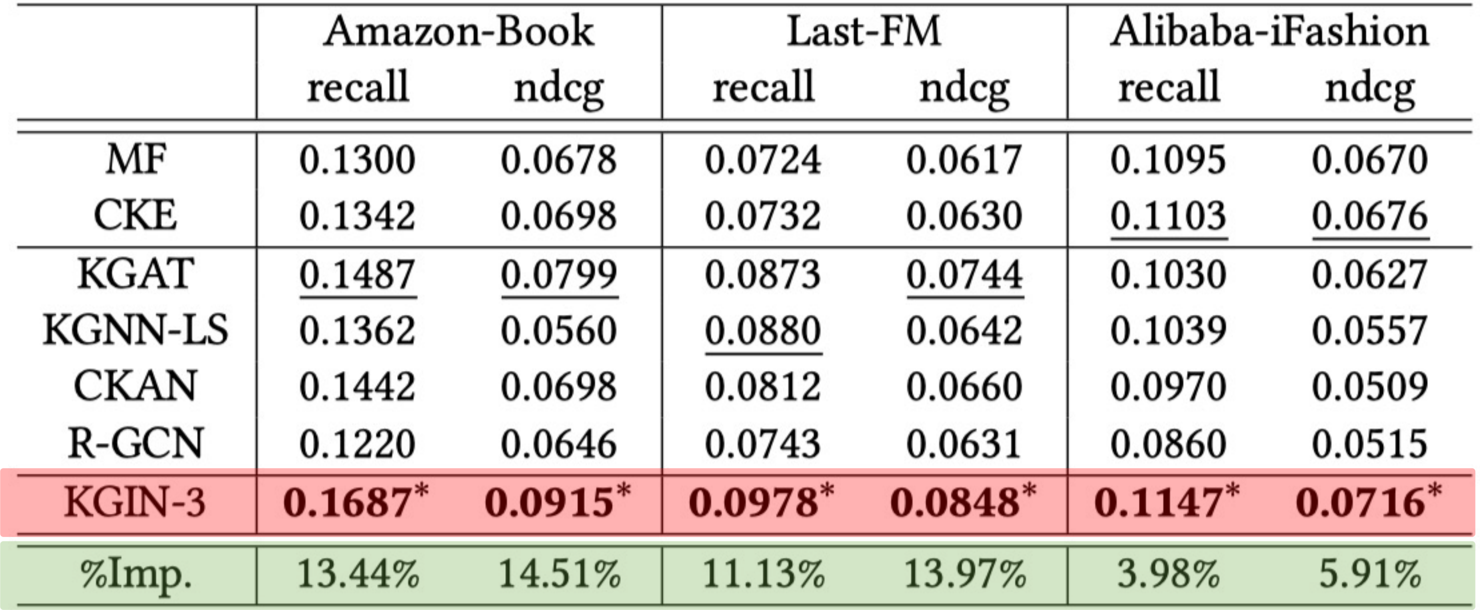
$$\mathbf{e}_i^{(l)} = \sum_{s \in \mathcal{N}_i^l} \frac{\mathbf{e}_{r_1}}{|\mathcal{N}_{s_1}|} \odot \frac{\mathbf{e}_{r_2}}{|\mathcal{N}_{s_2}|} \odot \cdots \odot \frac{\mathbf{e}_{r_l}}{|\mathcal{N}_{s_l}|} \odot \mathbf{e}_{s_l}^{(0)},$$

- reflects the interactions among relations
- preserves the holistic semantics of paths

$$s = i \xrightarrow{r_1} s_1 \xrightarrow{r_2} \cdots s_{l-1} \xrightarrow{r_l} s_l$$

Wang et al. Learning Intents behind Interactions with Knowledge Graph for Recommendation. WWW'2021

Q2: How to Tailor GNNs for KG-aware Rec?
Experimental Results



	Amazon-Book		Last-FM		Alibaba-iFashion	
	recall	ndcg	recall	ndcg	recall	ndcg
MF	0.1300	0.0678	0.0724	0.0617	0.1095	0.0670
CKE	0.1342	0.0698	0.0732	0.0630	0.1103	0.0676
KGAT	0.1487	0.0799	0.0873	0.0744	0.1030	0.0627
KGNN-LS	0.1362	0.0560	0.0880	0.0642	0.1039	0.0557
CKAN	0.1442	0.0698	0.0812	0.0660	0.0970	0.0509
R-GCN	0.1220	0.0646	0.0743	0.0631	0.0860	0.0515
KGIN-3	**0.1687***	**0.0915***	**0.0978***	**0.0848***	**0.1147***	**0.0716***
%Imp.	13.44%	14.51%	11.13%	13.97%	3.98%	5.91%

- KGIN consistently yields the **best** performance on all three datasets.

- This verifies the importance of:
  - Capturing collaborative signal in **intent-aware interaction graphs**;
  - Preserving **holistic semantics of paths**;

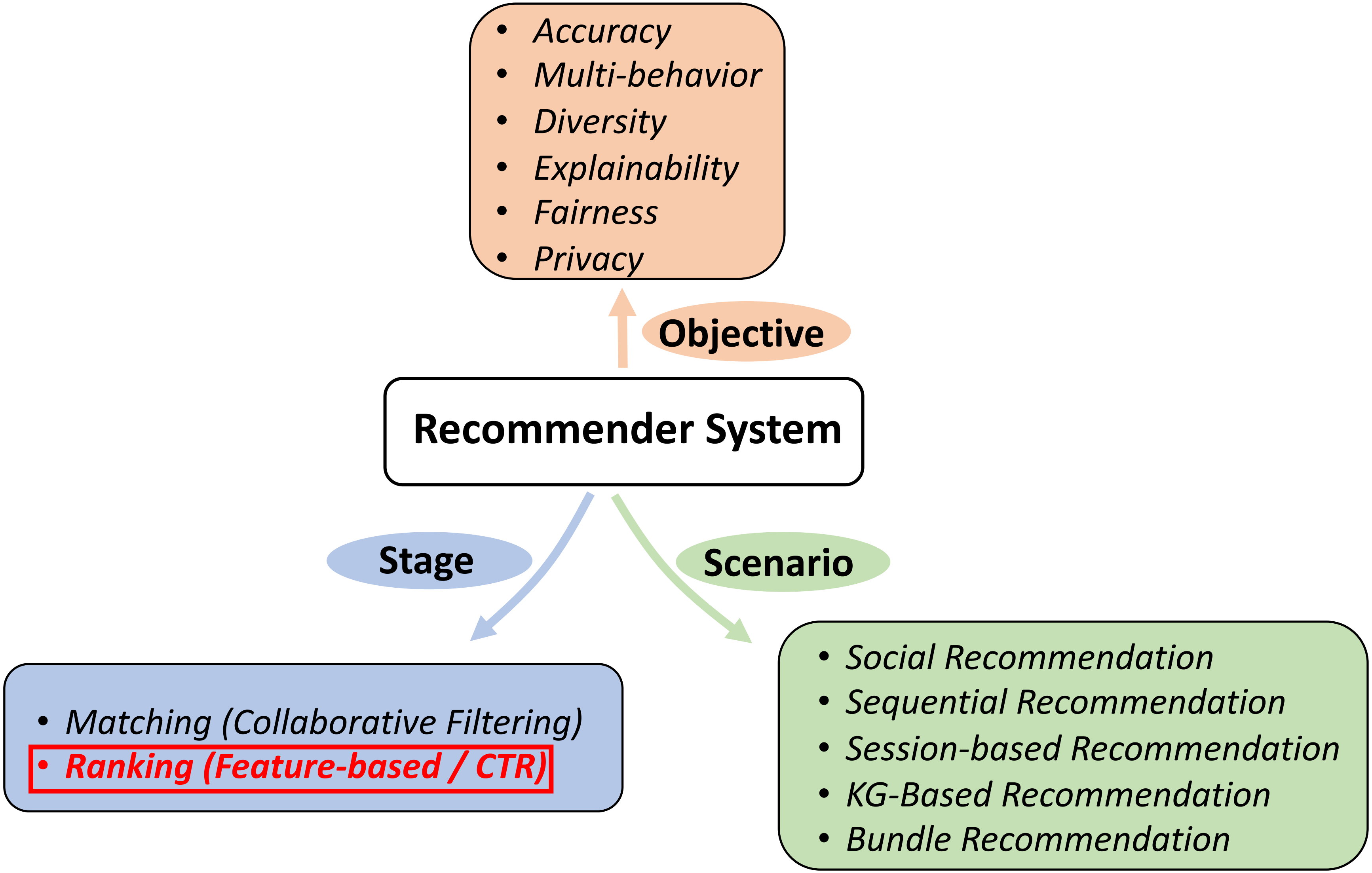
- KGIN can better encode collaborative signals & item knowledge into user and item representations.

THANK YOU!

Outline

- Background
  - Recommender System
  - Graph Neural Network
- Motivations and Challenges of GNN-based RecSys
- Recent Advances of GNN-based RecSys
- Open Problems and Future Directions

Recent advances of GNN-based RecSys



- *Accuracy*
- *Multi-behavior*
- *Diversity*
- *Explainability*
- *Fairness*
- *Privacy*

Objective

Recommender System

Stage

Scenario

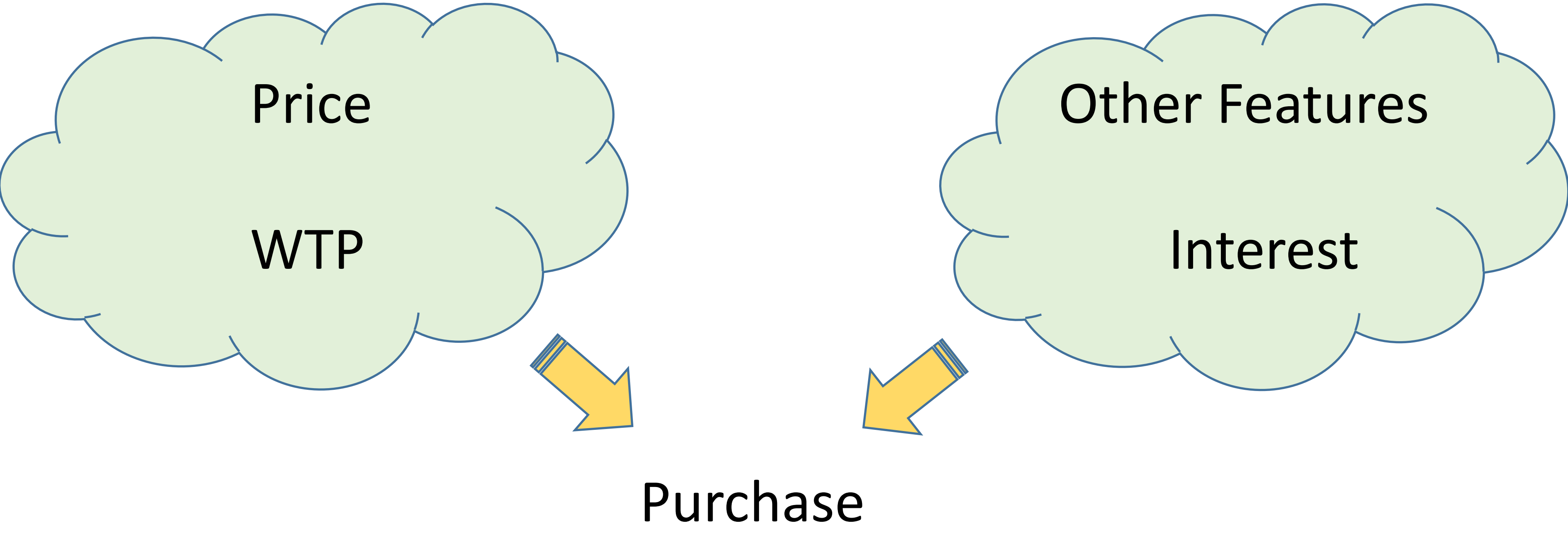
- *Matching (Collaborative Filtering)*
- ***Ranking (Feature-based / CTR)***

- *Social Recommendation*
- *Sequential Recommendation*
- *Session-based Recommendation*
- *KG-Based Recommendation*
- *Bundle Recommendation*

Price-aware recommendation with graph convolutional networks
Zheng, Y., Gao, C., He, X., Li, Y., & Jin, D. ICDE 2020

Background

- The price factor, which directly determines whether a user is willing to pay (WTP) for an item, is an important feature, while different from other features
- Price and other features play orthogonal roles in user decision making process



Price

WTP

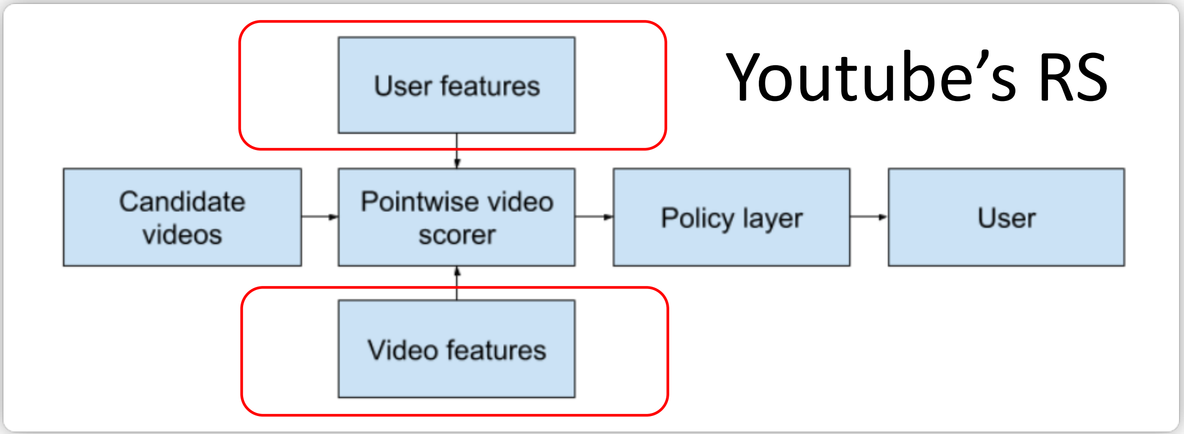
Other Features

Interest

Purchase

Background

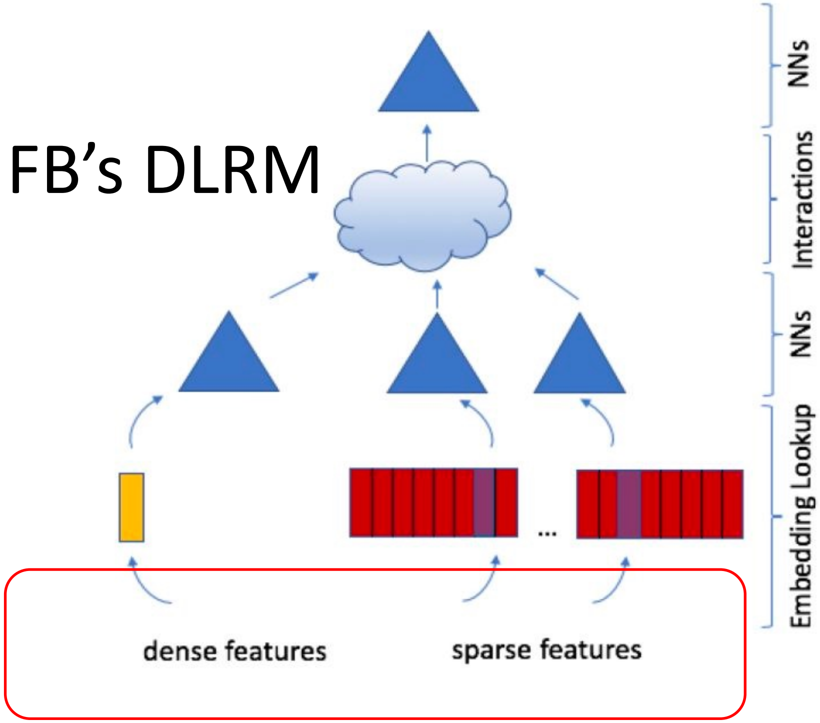
- Attribute-aware Recommendation incorporates all kinds of features into Collaborative Filtering (CF) to boost recommendation accuracy
- **Features**:
  - user feature
  - item feature
  - context feature



Background

- Trivial idea: use existing attribute-aware RS to model price
- Most attribute-aware recommendation systems treat different features equally
- Different features are captured in a generic and unified way
- e.g. FM, DeepFM, DLRM
- Features are usually fed into the model as dense features, sparse features, embedding features

FB’s DLRM



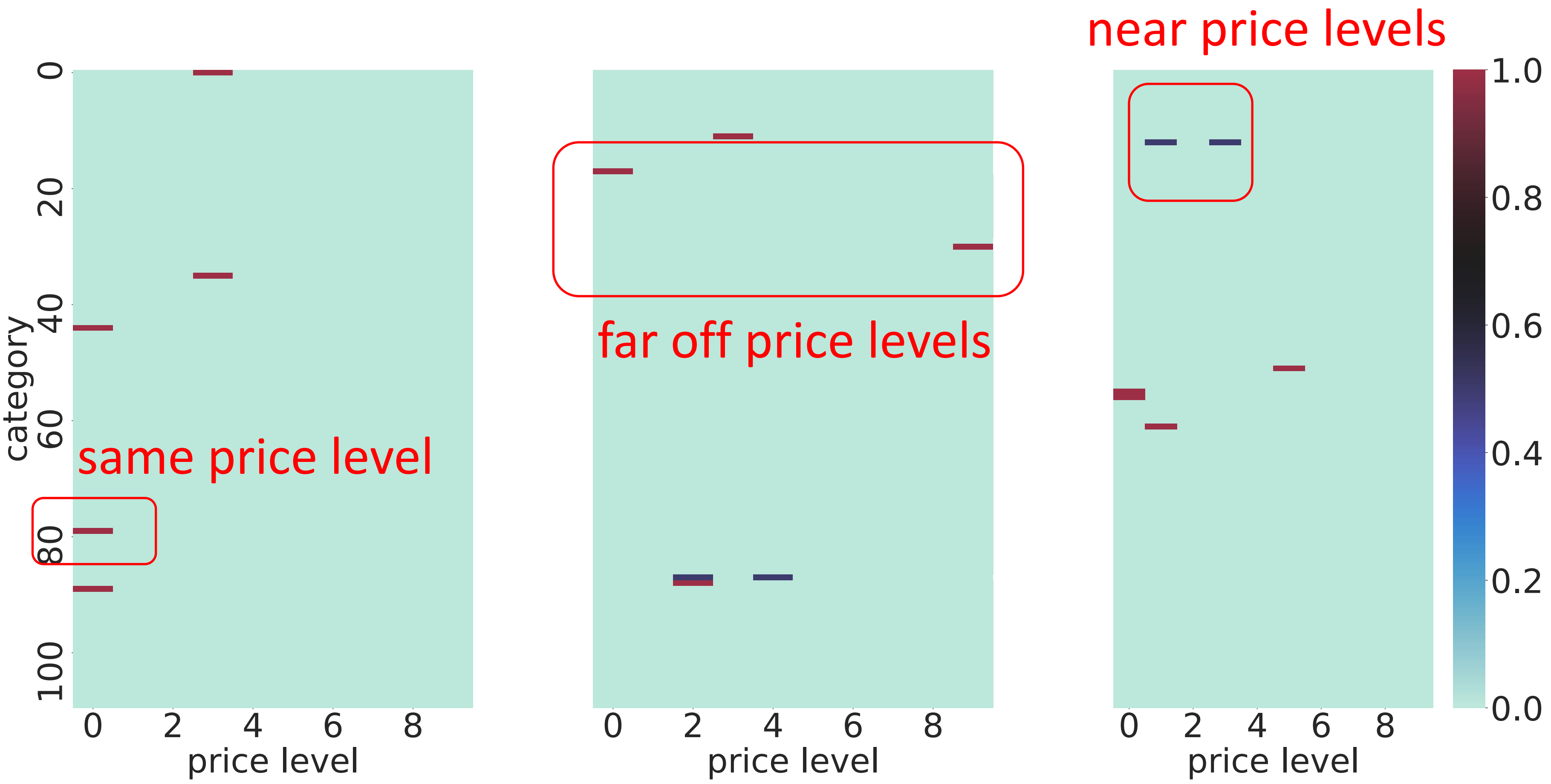
Challenges

- Implicit (unstated price awareness)
- Users seldomly speak out their preference or sensitivity on item price explicitly
- The price awareness can only be implicitly inferred from purchase history

- Complex (category-dependent influence)
- Price awareness or sensitivity is different across distinct product categories
- e.g. a sport lover would have high tolerance on the price of a sport equipment, but not on alcoholic drinks.

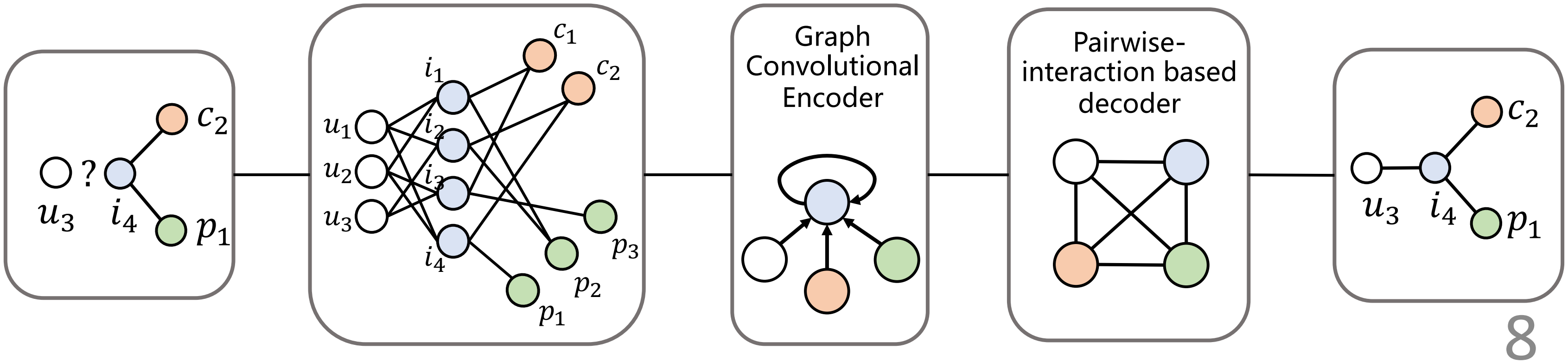
Challenges

- Purchase history as price-category heatmap of 3 randomly selected users from an e-commerce dataset



Methodology: Our PUP Model

- Price-aware User Preference-modeling (PUP)
- Input:
  - Interaction Matrix R
  - price of items p
  - category of items c
- Output:
  - estimated interaction probability given a user-item pair (u, i)



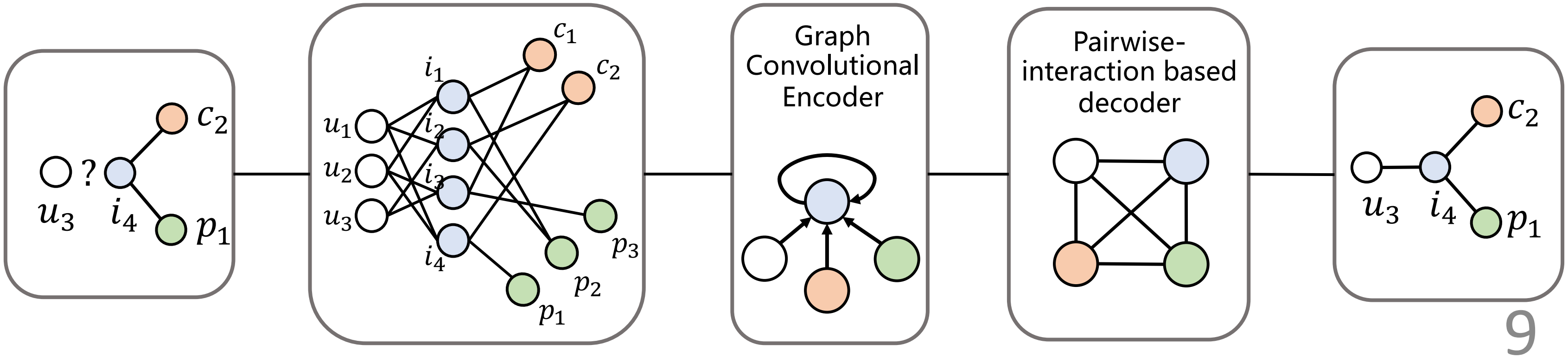
Graph Convolutional Encoder

Pairwise-interaction based decoder

Methodology: Our PUP Model

- two-branch solution
  - global branch: price as a global effect representing overall purchasing power (unrelated to category)
  - category branch: category-dependent influence of price factor

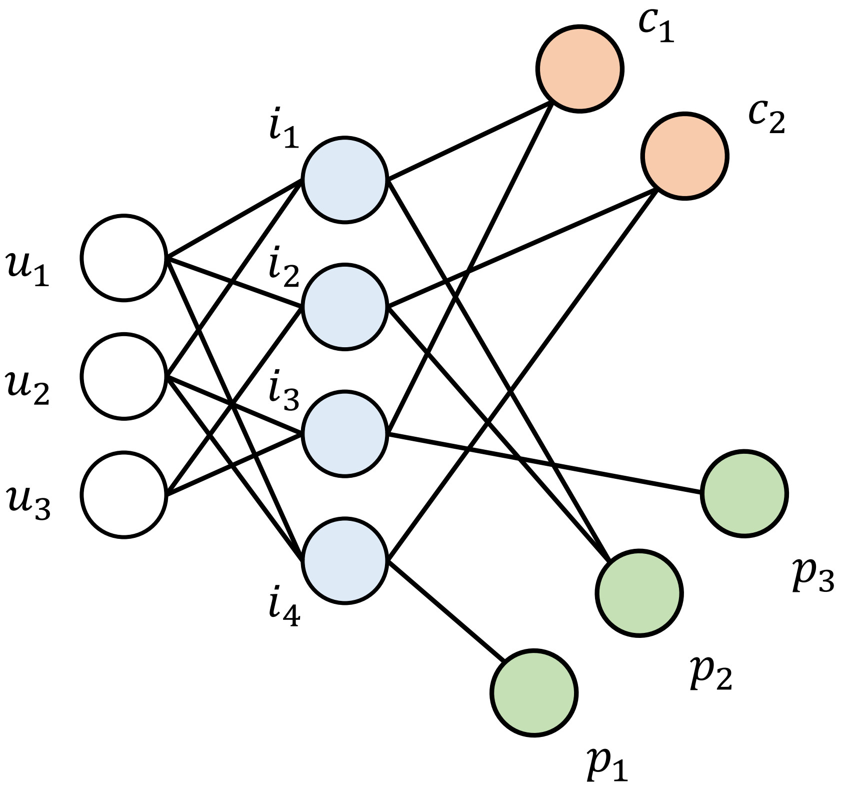
- Unified Graph Construction
- Graph Convolutional Encoder
- Pairwise-interaction Based Decoder



Methodology: Our PUP Model

- Why we use GCN?

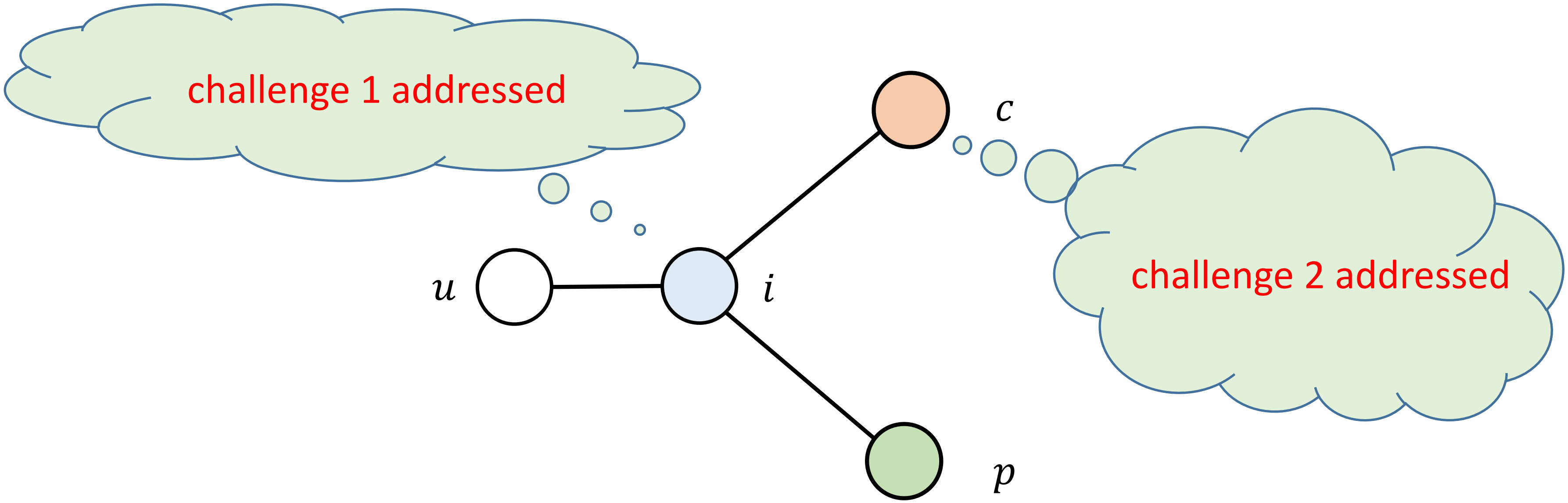
- Capture CF effect
- Learn robust representations for heterogeneous entities
- Model high-order similarity



Methodology: Our PUP Model

- Graph Construction
- Nodes: user, item, price, category
- Edges: user-item, item-price, item-category

- Price: we discretize price within each category using uniform quantization

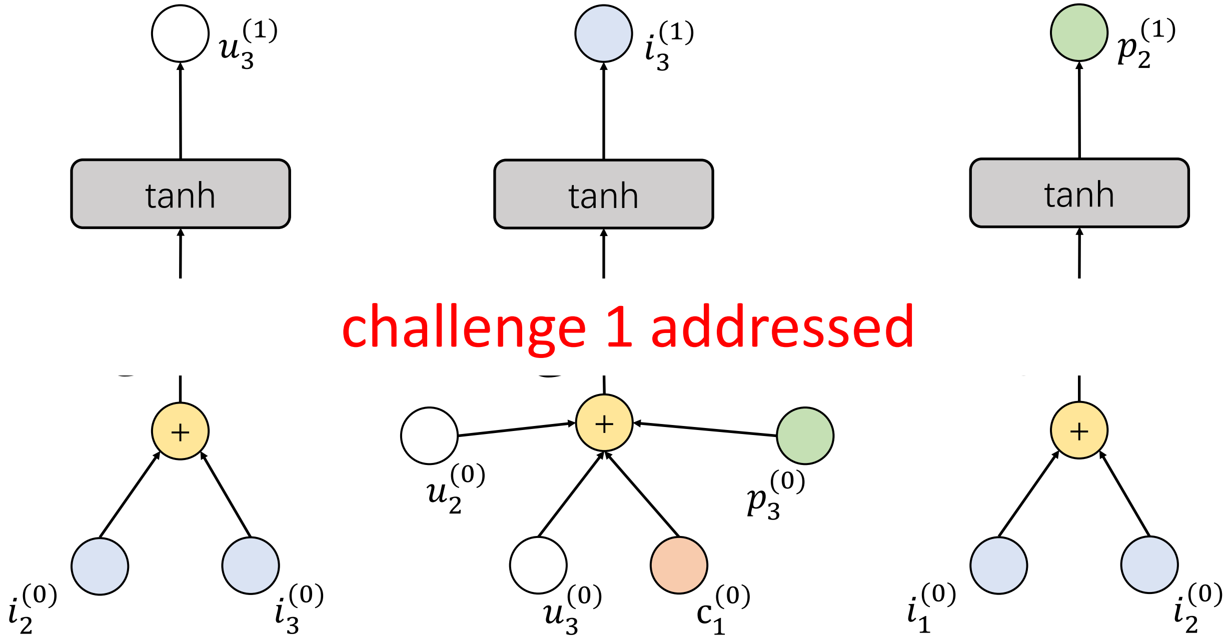


challenge 1 addressed

challenge 2 addressed

Methodology: Our PUP Model

- Graph Convolutional Encoder
- Embedding Layer: transform one-hot feature to embedding feature
- Graph Convolutional Layer: embedding propagation and neighbor aggregation



challenge 1 addressed

Methodology: Our PUP Model

- Pairwise-interaction Based Decoder
- Global branch:

user's interest and overall purchasing power

$$s_g = e_{ug}^T e_{ig} + e_{ug}^T e_{pg} + e_{ig}^T e_{pg}$$

challenge 2 addressed

- Category branch:

$$s_c = e_{uc}^T e_{pc} + e_{uc}^T e_{cc} + e_{cc}^T e_{pc}$$

user's category-dependent price awareness

- Final prediction:

$$s = \ s_g + \alpha s_c$$

balance between two aspects

Methodology: Our PUP Model

- Model training
- Semi-supervised graph encoder
  - Encoding: learn expressive representations for all kinds of nodes
  - Decoding: only focus on reconstructing user-item edges

- Loss function
  - BPR:

$$L = \sum_{(u,i,j)\in\mathcal{O}} -\ln\Big(\sigma\big(s(u,i) - s(u,j)\big)\Big) + \lambda\|\Theta\|^2$$

Experiments

- Datasets
  - Two real-world datasets: restaurant and e-commerce

dataset	#users	#items	#cate	#price	#interaction
Yelp restaurant	20637	18907	89	4	505785
Beibei	52767	39303	110	10	677065

- Evaluation protocols:
  - Top-K evaluation with two metrics Recall and NDCG.
- Baseline
  - Non-personalized: ItemPop
  - CF: BPR-MF (UAI2009), GC-MC (KDD2018 Deep Learning Day) , NGCF (SIGIR2019)
  - Attribute-aware: FM (ICDM2010), DeepFM (IJCAI2017)
  - Price-aware: PaDQ-CMF (SIGIR2014)

Experiments

- Research questions
- **RQ1:** How does PUP perform compared with other baseline methods ?
- **RQ2:** Could PUP recommend items which match users' price awareness ?
- **RQ3:** Could price help recommendation system in cold start scenarios ?

Experiments

- Overall Comparison

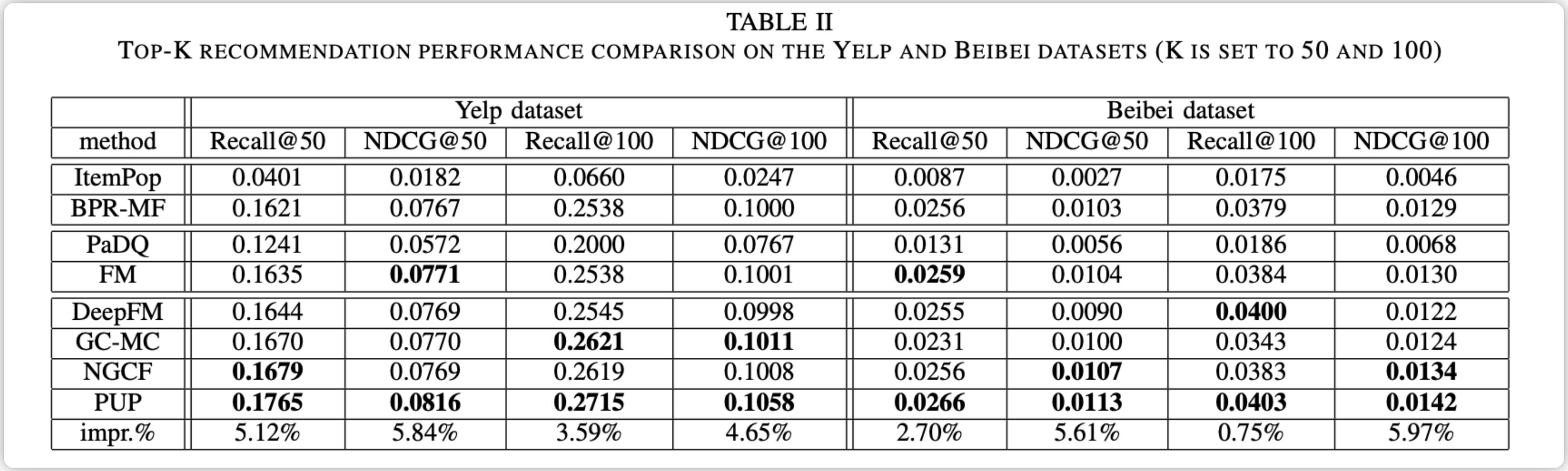


TABLE II
TOP-K RECOMMENDATION PERFORMANCE COMPARISON ON THE YELP AND BEIBEI DATASETS (K IS SET TO 50 AND 100)

	Yelp dataset				Beibei dataset			
method	Recall@50	NDCG@50	Recall@100	NDCG@100	Recall@50	NDCG@50	Recall@100	NDCG@100
ItemPop	0.0401	0.0182	0.0660	0.0247	0.0087	0.0027	0.0175	0.0046
BPR-MF	0.1621	0.0767	0.2538	0.1000	0.0256	0.0103	0.0379	0.0129
PaDQ	0.1241	0.0572	0.2000	0.0767	0.0131	0.0056	0.0186	0.0068
FM	0.1635	**0.0771**	0.2538	0.1001	**0.0259**	0.0104	0.0384	0.0130
DeepFM	0.1644	0.0769	0.2545	0.0998	0.0255	0.0090	**0.0400**	0.0122
GC-MC	0.1670	0.0770	**0.2621**	**0.1011**	0.0231	0.0100	0.0343	0.0124
NGCF	**0.1679**	0.0769	0.2619	0.1008	0.0256	**0.0107**	0.0383	**0.0134**
PUP	**0.1765**	**0.0816**	**0.2715**	**0.1058**	**0.0266**	**0.0113**	**0.0403**	**0.0142**
impr.%	5.12%	5.84%	3.59%	4.65%	2.70%	5.61%	0.75%	5.97%

Experiments

- Observations

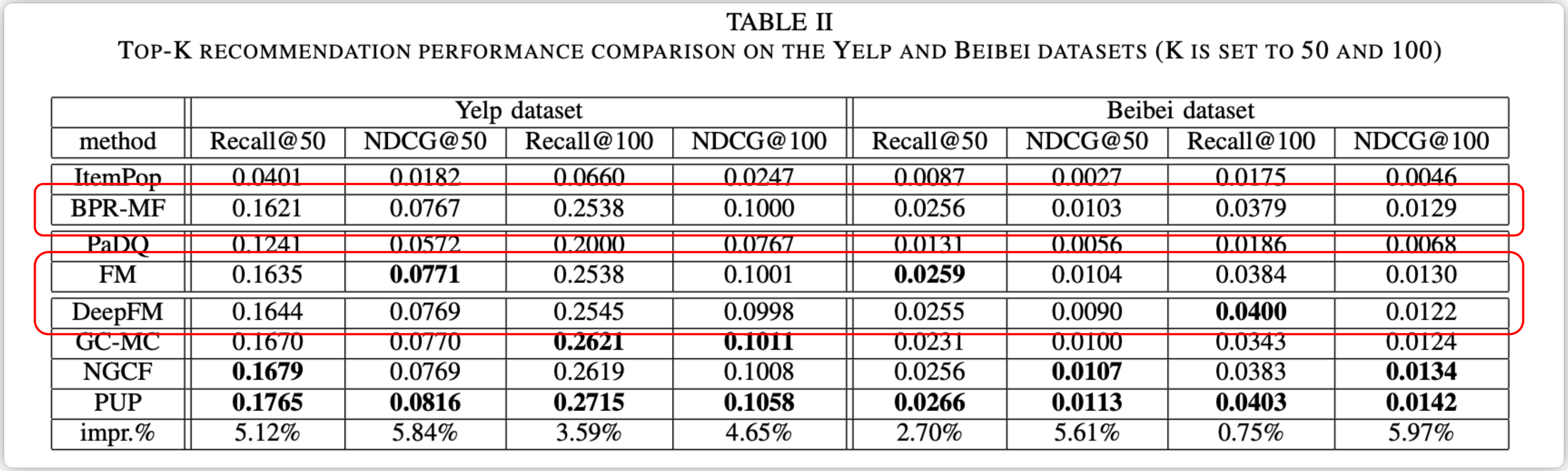


TABLE II
TOP-K RECOMMENDATION PERFORMANCE COMPARISON ON THE YELP AND BEIBEI DATASETS (K IS SET TO 50 AND 100)

	Yelp dataset				Beibei dataset			
method	Recall@50	NDCG@50	Recall@100	NDCG@100	Recall@50	NDCG@50	Recall@100	NDCG@100
ItemPop	0.0401	0.0182	0.0660	0.0247	0.0087	0.0027	0.0175	0.0046
BPR-MF	0.1621	0.0767	0.2538	0.1000	0.0256	0.0103	0.0379	0.0129
PaDQ	0.1241	0.0572	0.2000	0.0767	0.0131	0.0056	0.0186	0.0068
FM	0.1635	**0.0771**	0.2538	0.1001	**0.0259**	0.0104	0.0384	0.0130
DeepFM	0.1644	0.0769	0.2545	0.0998	0.0255	0.0090	**0.0400**	0.0122
GC-MC	0.1670	0.0770	**0.2621**	**0.1011**	0.0231	0.0100	0.0343	0.0124
NGCF	**0.1679**	0.0769	0.2619	0.1008	0.0256	**0.0107**	0.0383	**0.0134**
PUP	**0.1765**	**0.0816**	**0.2715**	**0.1058**	**0.0266**	**0.0113**	**0.0403**	**0.0142**
impr.%	5.12%	5.84%	3.59%	4.65%	2.70%	5.61%	0.75%	5.97%

- Attribute-aware methods generally outperforms trivial CF methods, e.g. FM & DeepFM vs BPR-MF. **Incorporating price into recommendation improves accuracy.**

Experiments

- Observations

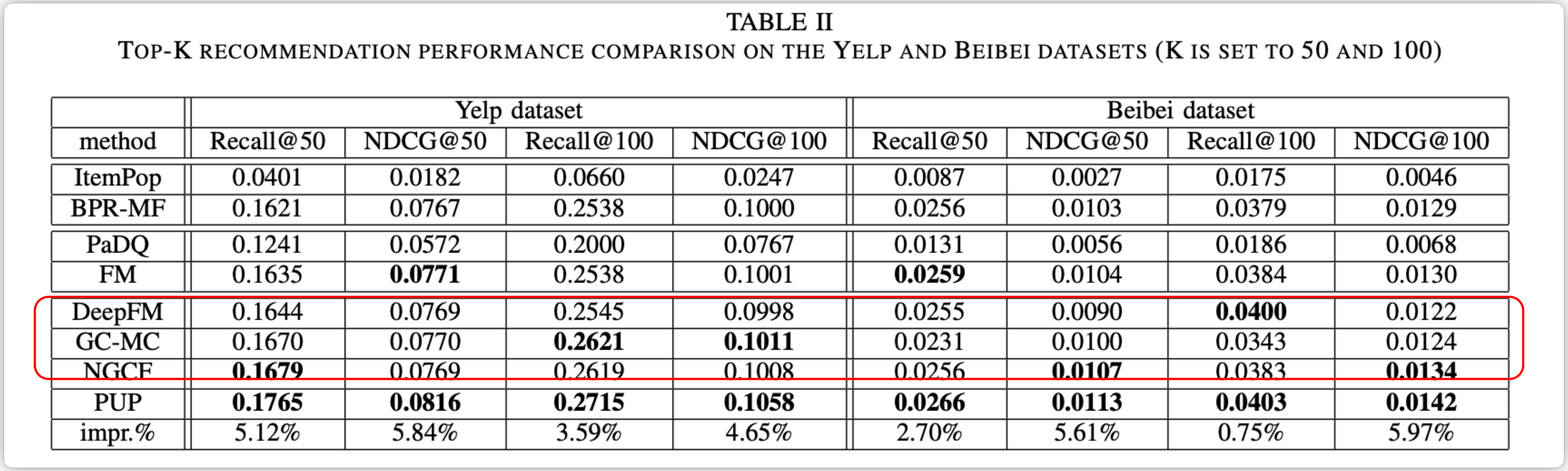


TABLE II

Top-K recommendation performance comparison on the Yelp and Beibei datasets (K is set to 50 and 100)

	Yelp dataset				Beibei dataset			
method	Recall@50	NDCG@50	Recall@100	NDCG@100	Recall@50	NDCG@50	Recall@100	NDCG@100
ItemPop	0.0401	0.0182	0.0660	0.0247	0.0087	0.0027	0.0175	0.0046
BPR-MF	0.1621	0.0767	0.2538	0.1000	0.0256	0.0103	0.0379	0.0129
PaDQ	0.1241	0.0572	0.2000	0.0767	0.0131	0.0056	0.0186	0.0068
FM	0.1635	**0.0771**	0.2538	0.1001	**0.0259**	0.0104	0.0384	0.0130
DeepFM	0.1644	0.0769	0.2545	0.0998	0.0255	0.0090	**0.0400**	0.0122
GC-MC	0.1670	0.0770	**0.2621**	**0.1011**	0.0231	0.0100	0.0343	0.0124
NGCF	**0.1679**	0.0769	0.2619	0.1008	0.0256	**0.0107**	0.0383	**0.0134**
PUP	**0.1765**	**0.0816**	**0.2715**	**0.1058**	**0.0266**	**0.0113**	**0.0403**	**0.0142**
impr.%	5.12%	5.84%	3.59%	4.65%	2.70%	5.61%	0.75%	5.97%

- Models based on neural networks and graph neural networks achieve better results than other models in most cases. **It is promising to introduce deep models, especially GNN models into representation learning.**

Experiments

- Observations

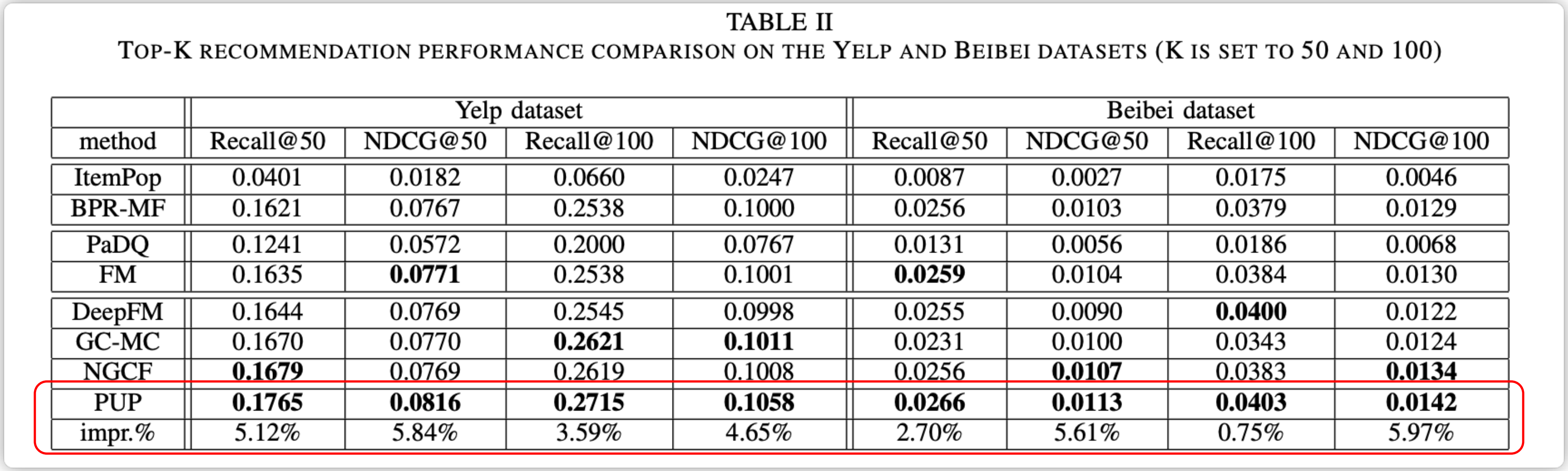


TABLE II

Top-K recommendation performance comparison on the Yelp and Beibei datasets (K is set to 50 and 100)

	Yelp dataset				Beibei dataset			
method	Recall@50	NDCG@50	Recall@100	NDCG@100	Recall@50	NDCG@50	Recall@100	NDCG@100
ItemPop	0.0401	0.0182	0.0660	0.0247	0.0087	0.0027	0.0175	0.0046
BPR-MF	0.1621	0.0767	0.2538	0.1000	0.0256	0.0103	0.0379	0.0129
PaDQ	0.1241	0.0572	0.2000	0.0767	0.0131	0.0056	0.0186	0.0068
FM	0.1635	**0.0771**	0.2538	0.1001	**0.0259**	0.0104	0.0384	0.0130
DeepFM	0.1644	0.0769	0.2545	0.0998	0.0255	0.0090	**0.0400**	0.0122
GC-MC	0.1670	0.0770	**0.2621**	**0.1011**	0.0231	0.0100	0.0343	0.0124
NGCF	**0.1679**	0.0769	0.2619	0.1008	0.0256	**0.0107**	0.0383	**0.0134**
PUP	**0.1765**	**0.0816**	**0.2715**	**0.1058**	**0.0266**	**0.0113**	**0.0403**	**0.0142**
impr.%	5.12%	5.84%	3.59%	4.65%	2.70%	5.61%	0.75%	5.97%

- **Our proposed PUP achieves the best performance.** The improvements are statistically significant for p < 0.005.

Experiments

- User study

	user 50432	user 30901	user 36035	user 12359
user	👤	👤	👤	👤
non-sensitive $ $ $	housewares	skirts	high heels	jeans
recommend avg price	4.75	2.76	2.86	3.53
sensitive $	toys	books	makeup	slipper
recommend avg price	0.51	0.08	0.30	0.58

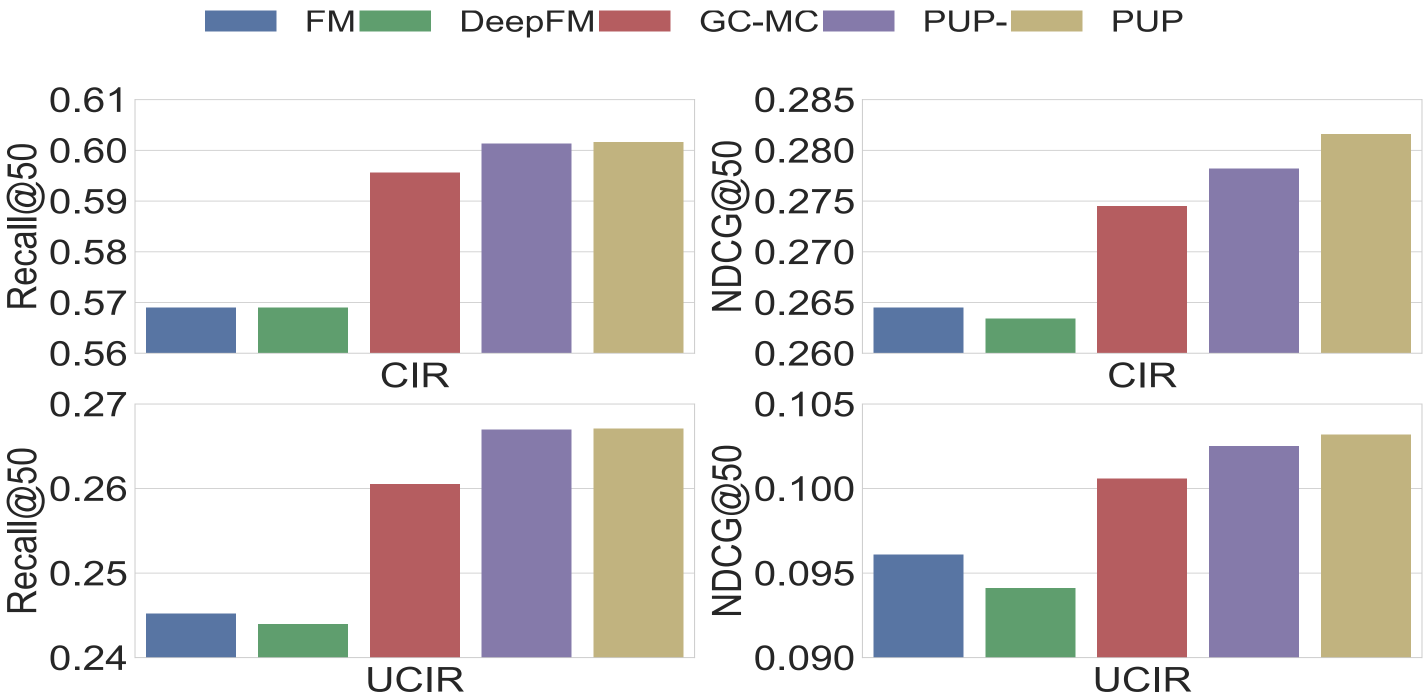
- PUP successfully recommend items matching users’ price awareness

Experiments

- Utilizing price to tackle **cold-start** problem
- Recommend items of **unexplored categories**
- CIR (Category Item Recommendation): recommend from unexplored "positive" categories in the test set
- UCIR (Unexplored Category Item Recommendation): recommend from all categories not explored in the training set
- Example:
  - All categories {A, B, C, D, E, F, G}
  - Explored categories {A, B, C} in training set
  - Explored category {E} in test set
  - CIR: recommend from all items of category E
  - UCIR: recommend from all items of category {D, E, F, G}

Experiments

- Utilizing price to tackle **cold-start** problem



- Graph based methods outperform other methods: items of cold-start categories are more reachable on the graph
- User’s price awareness could bridge the gap between explored and cold-start categories

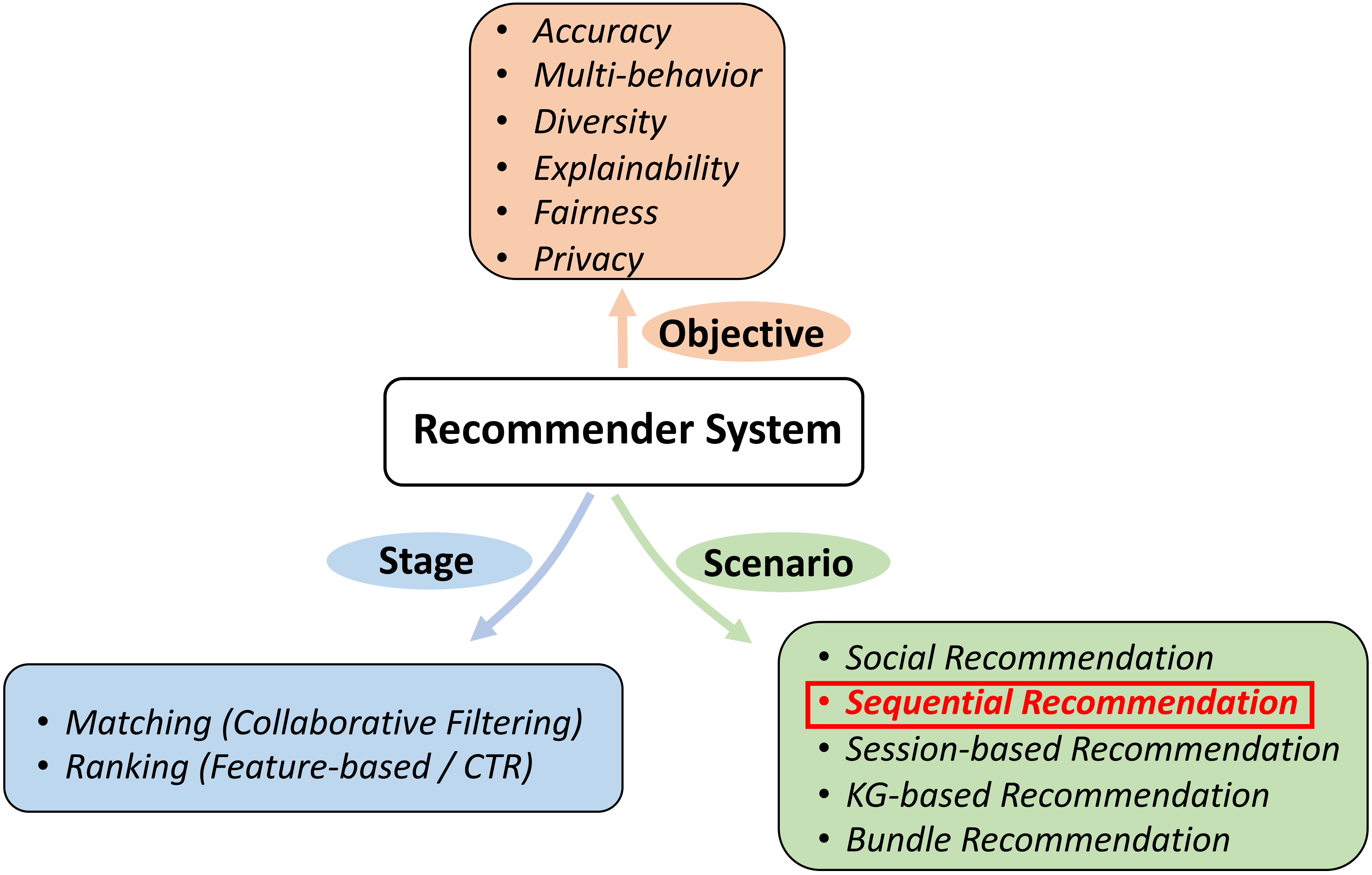
Conclusion

- 1. We highlight the significance of **incorporating price into recommendation** and analyze the two difficulties in capturing price (unstated price awareness and category dependent influence).

- 2. we propose a **GCN-based** method named PUP and adopt a **two-branch structure** which is specifically designed to separately model the global and category-dependent effect of the price awareness

- 3. Our proposed method could recommend items matching users' price awareness and alleviate the **cold-start** problem when recommending items from unexplored categories.

Recent advances of GNN-based RecSys



- *Accuracy*
- *Multi-behavior*
- *Diversity*
- *Explainability*
- *Fairness*
- *Privacy*

Objective

Recommender System

Stage

Scenario

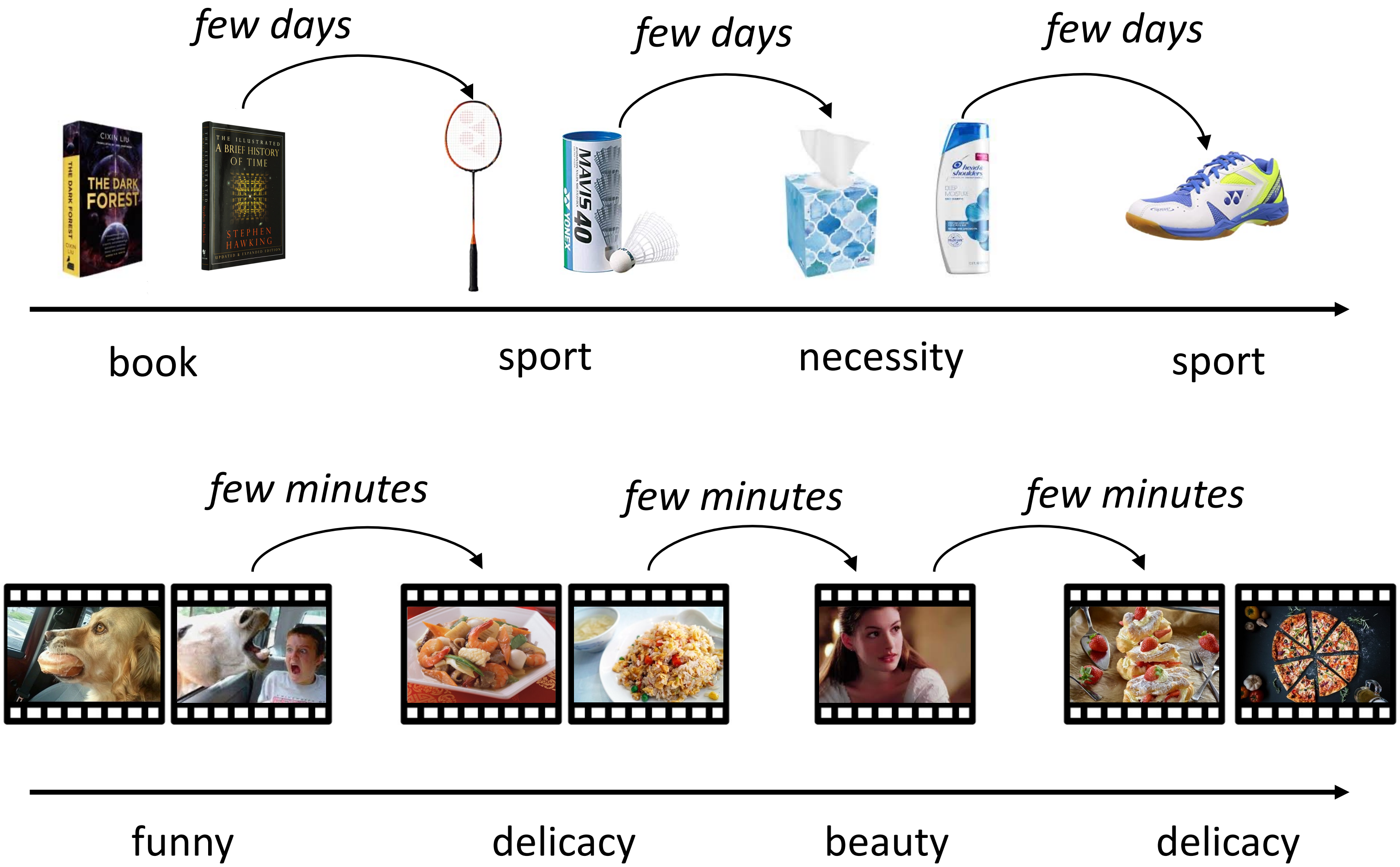
- *Matching (Collaborative Filtering)*
- *Ranking (Feature-based / CTR)*

- *Social Recommendation*
- ***Sequential Recommendation***
- *Session-based Recommendation*
- *KG-based Recommendation*
- *Bundle Recommendation*

Sequential Recommendation with Graph Neural Networks.
Chang, J., Gao, C., Zheng, Y., Hui, Y., Niu, Y., Song, Y., ... & Li, Y. SIGIR 2021

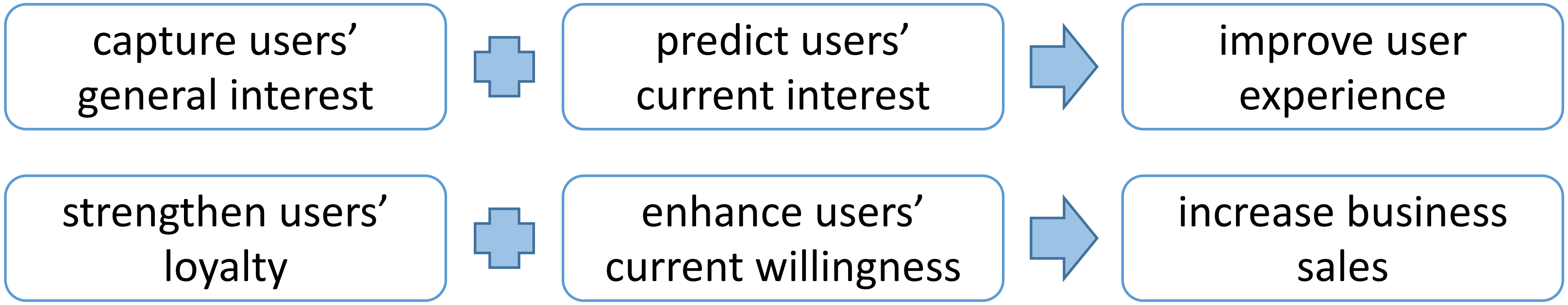
Background

- What and why is sequential recommendation(SR)?



Background

- Sequential recommendation(SR) aims to leverage users' historical behaviors to predict their next interaction.



capture users' general interest + predict users' current interest → improve user experience

strengthen users' loyalty + enhance users' current willingness → increase business sales

- The accumulation of user behaviors on e-commerce and content platforms makes it become an important task.

e-commerce

content

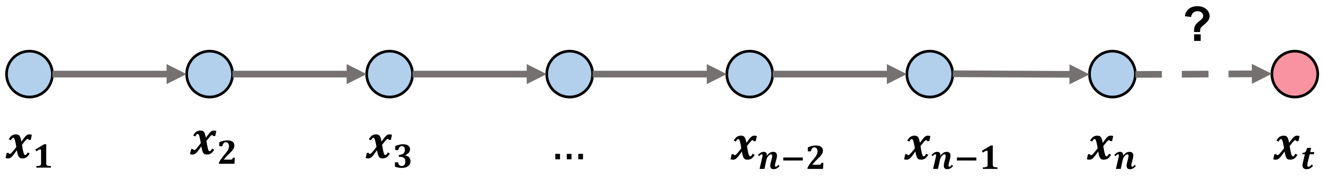
Problem Definition

❑ Input:

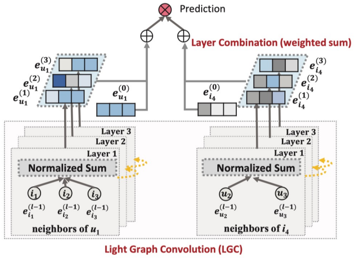
- the interaction history $\{x_1, x_2, \ldots\ x_n\}$ for each user

❑ Output:

- the probability that a user with interaction history $\{x_1, x_2, \ldots\ x_n\}$ will interact with the target item x_t at the $(\mathrm{n}+1){-}th$ step



Related Work



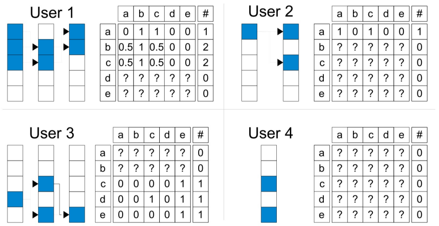
NCF [WWW'17]
LightGCN [SIGIR'20]

Traditional Recommendations:

- model user-item interaction in a static fashion.

Early efforts in SR:

- use designed rules or attention mechanism to assign weights.



User 1

	a	b	c	d	e	#
a	0	1	1	0	0	1
b	0.5	1	0.5	0	0	2
c	0.5	1	0.5	0	0	2
d	?	?	?	?	?	0
e	?	?	?	?	?	0

User 2

	a	b	c	d	e	#
a	1	0	1	0	0	1
b	?	?	?	?	?	0
c	?	?	?	?	?	0
d	?	?	?	?	?	0
e	?	?	?	?	?	0

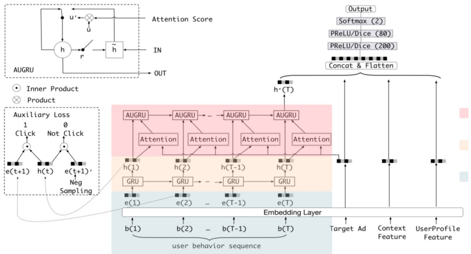
User 3

	a	b	c	d	e	#
a	?	?	?	?	?	0
b	?	?	?	?	?	0
c	0	0	0	0	1	1
d	0	0	1	0	1	1
e	0	0	0	0	1	1

User 4

	a	b	c	d	e	#
a	?	?	?	?	?	0
b	?	?	?	?	?	0
c	?	?	?	?	?	0
d	?	?	?	?	?	0
e	?	?	?	?	?	0

FPMC [WWW'10]
DIN [KDD'18]



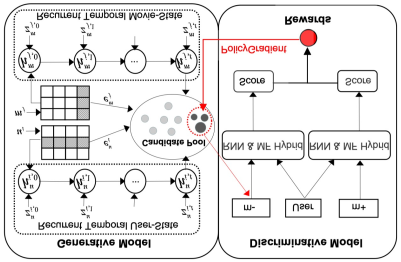
GRU4REC[ICLR'16]
Caser[WSDM'18]
DIEN [AAAI'19]

Mainstream methods in SR:

- leverage RNN/CNN to summarize the behavioral sequences.

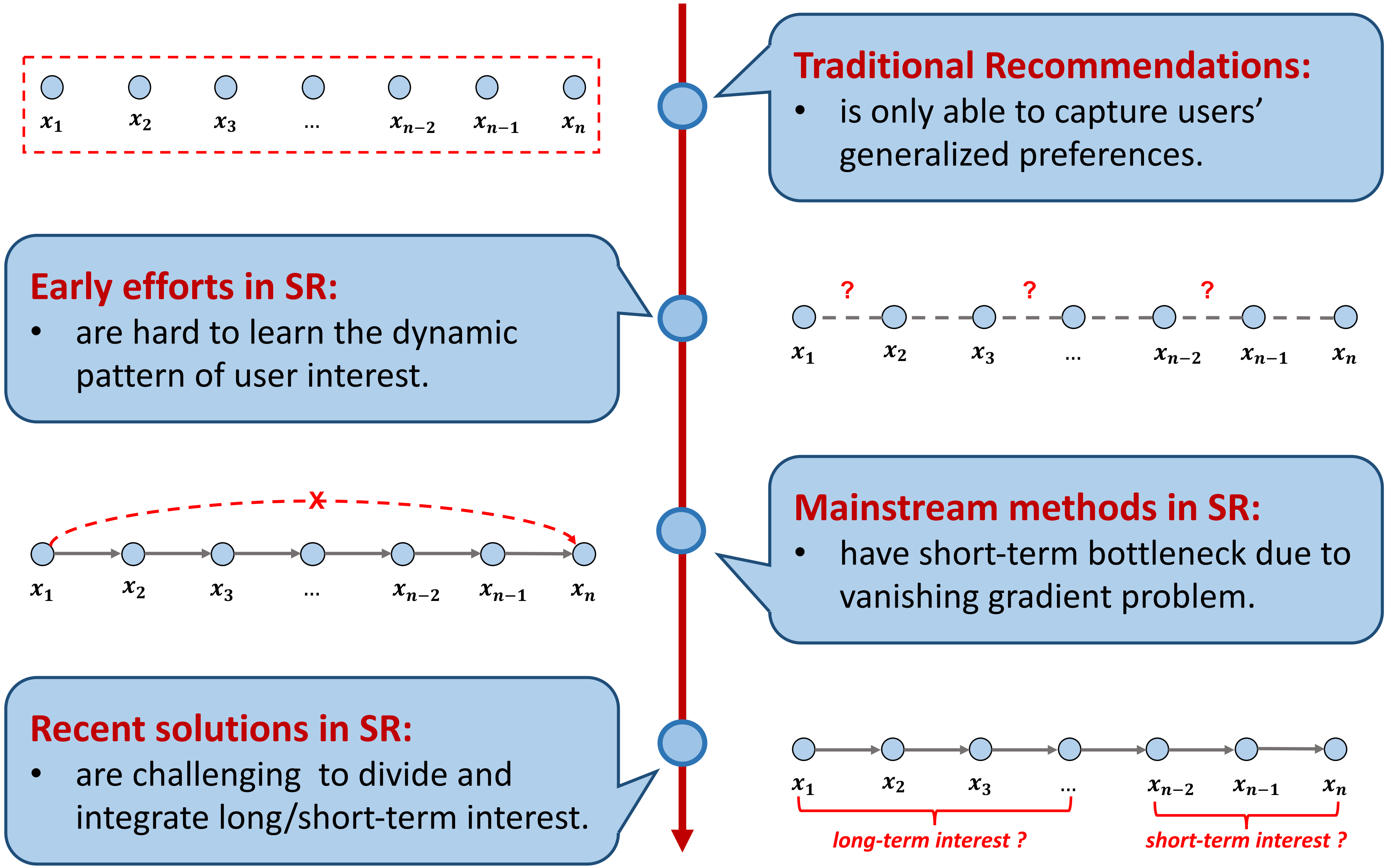
Recent solutions in SR:

- jointly model long/short-term interests to avoid forgetting.



PLASTIC [IJCAI'18]
SLi-Rec [IJCAI'19]

Limitations of Existing Work



Traditional Recommendations:

- is only able to capture users' generalized preferences.

Early efforts in SR:

- are hard to learn the dynamic pattern of user interest.

Mainstream methods in SR:

- have short-term bottleneck due to vanishing gradient problem.

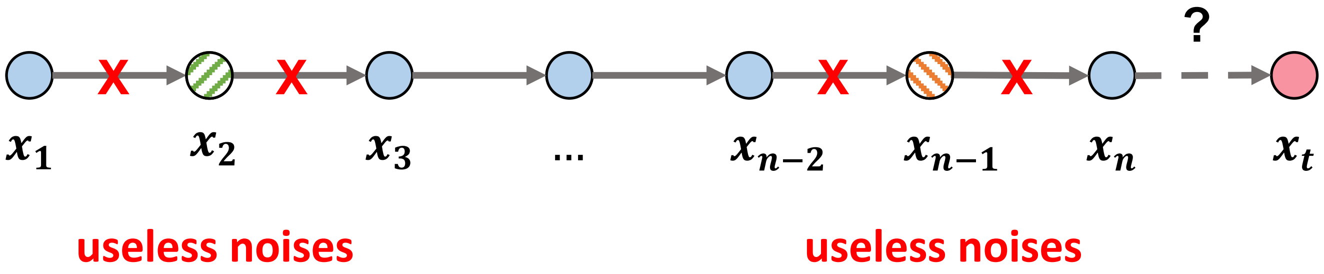
Recent solutions in SR:

- are challenging to divide and integrate long/short-term interest.

x_1 x_2 x_3 … x_{n-2} x_{n-1} x_n
?
?
?
x_1 x_2 x_3 … x_{n-2} x_{n-1} x_n
x
x_1 x_2 x_3 … x_{n-2} x_{n-1} x_n
x_1 x_2 x_3 … x_{n-2} x_{n-1} x_n
long-term interest ?
short-term interest ?

Challenges

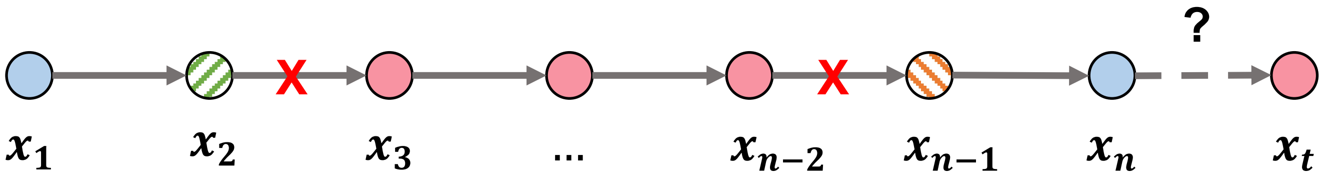
1. **User behaviors in long sequences reflect implicit and noisy preference signals.**
   - Users may interact with many items with **implicit feedback**, such as clicks and watches.
   - Unlike explicit feedback that can infer preferences, single ones cannot reflect user's **real interest**.
   - Useless records will serve as **noises in behavior history**, worsening the modeling of real interests.



Challenges

2. **User preferences are always drifting over time due to their diversity.**

   - User preferences are changing due to their **diversity**, no matter slow or fast.
   - Some preferences may be **activated** and some others may have been **deactivated**.
   - It is challenging to model **how they change** in the implicit and noisy history sequence.



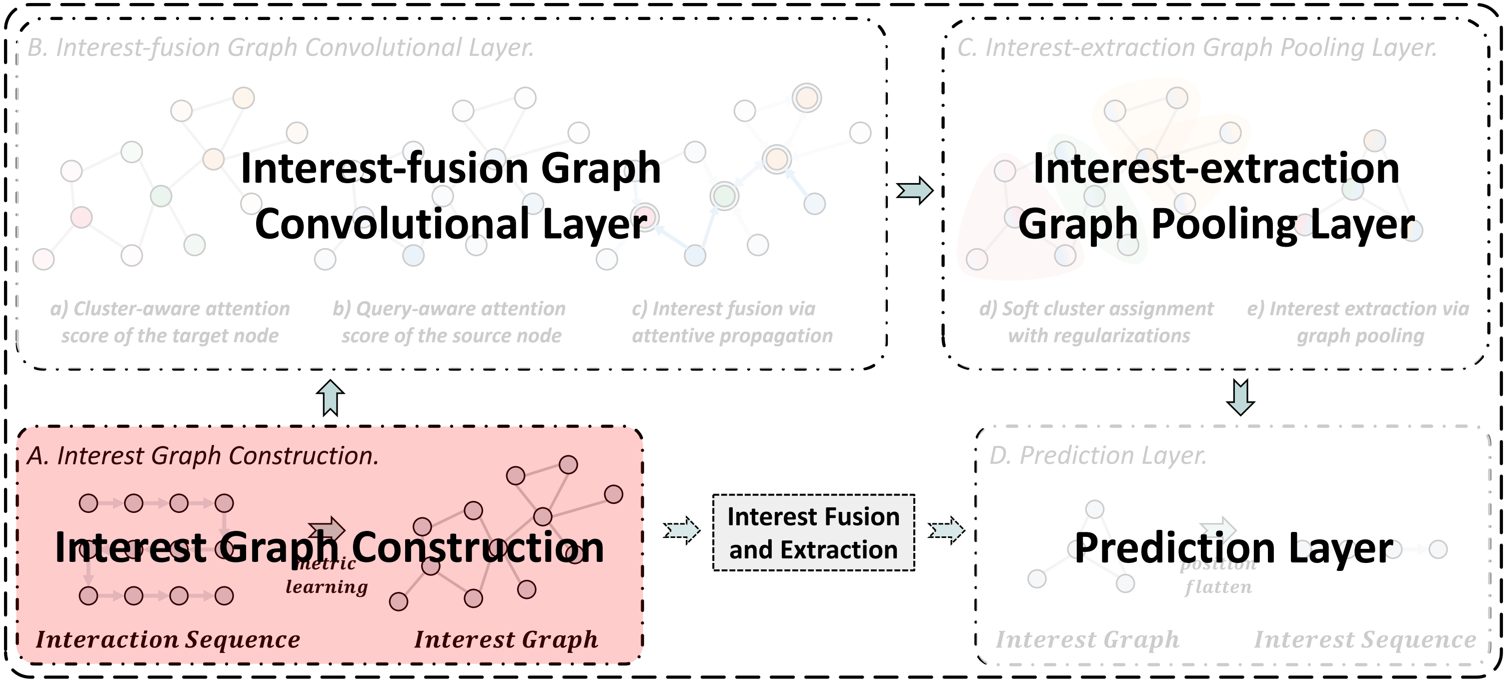
activated interest

deactivated interest

SURGE Model Framework

- ❑ Interest Graph Construction

Solve Challenge 1: by explicitly integrating and distinguishing different types of preferences


B. Interest-fusion Graph Convolutional Layer.
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
metric learning
Interaction Sequence
Interest Graph
D. Prediction Layer.
position flatten
Interest Graph
Interest Sequence

Interest-fusion Graph Convolutional Layer

Interest-extraction Graph Pooling Layer

Interest Graph Construction

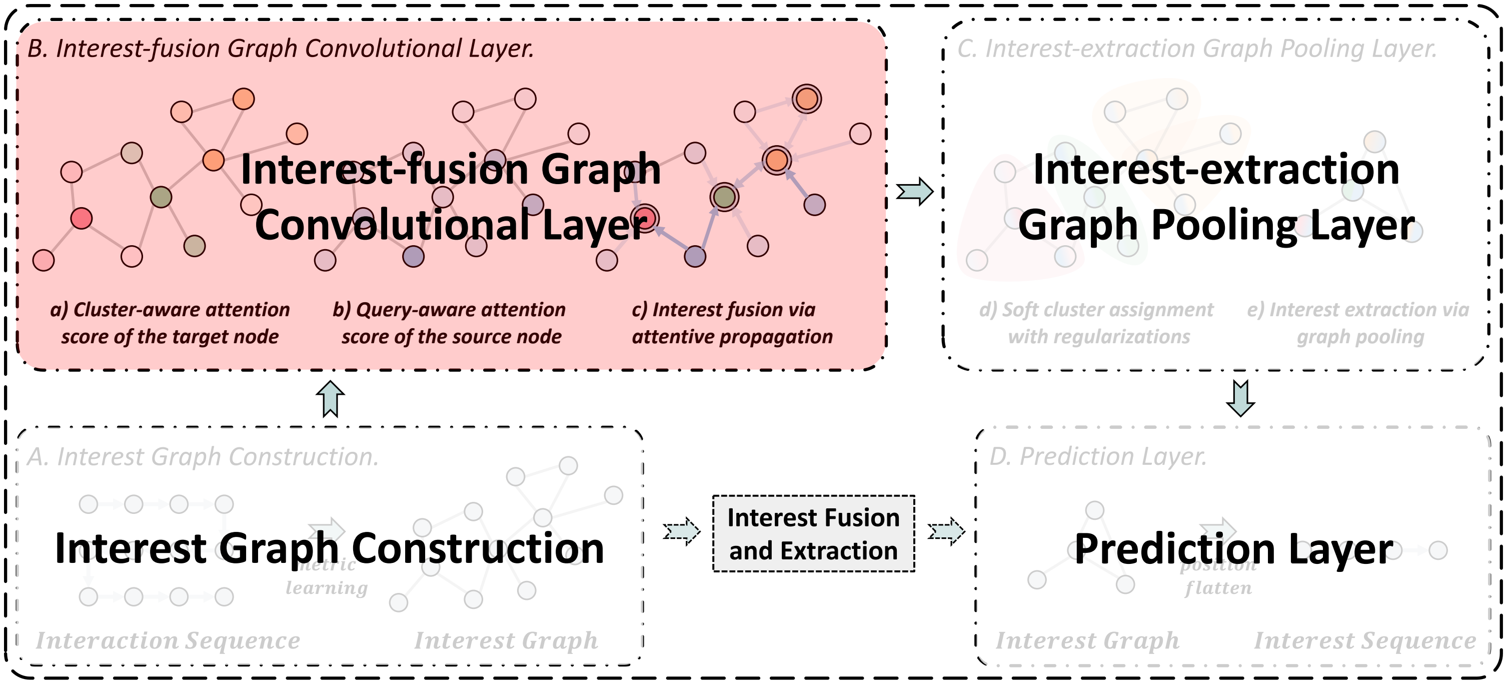
Interest Fusion and Extraction

Prediction Layer

SURGE Model Framework

- ❑ Interest-fusion Graph Convolutional Layer

Solve Challenge 1: by strengthening important behaviors and weakening noise behaviors.



B. Interest-fusion Graph Convolutional Layer.

Interest-fusion Graph Convolutional Layer

a) Cluster-aware attention score of the target node

b) Query-aware attention score of the source node

c) Interest fusion via attentive propagation

C. Interest-extraction Graph Pooling Layer.

Interest-extraction Graph Pooling Layer

d) Soft cluster assignment with regularizations

e) Interest extraction via graph pooling

A. Interest Graph Construction.

Interest Graph Construction

metric learning

Interaction Sequence

Interest Graph

Interest Fusion and Extraction

D. Prediction Layer.

Prediction Layer

position flatten

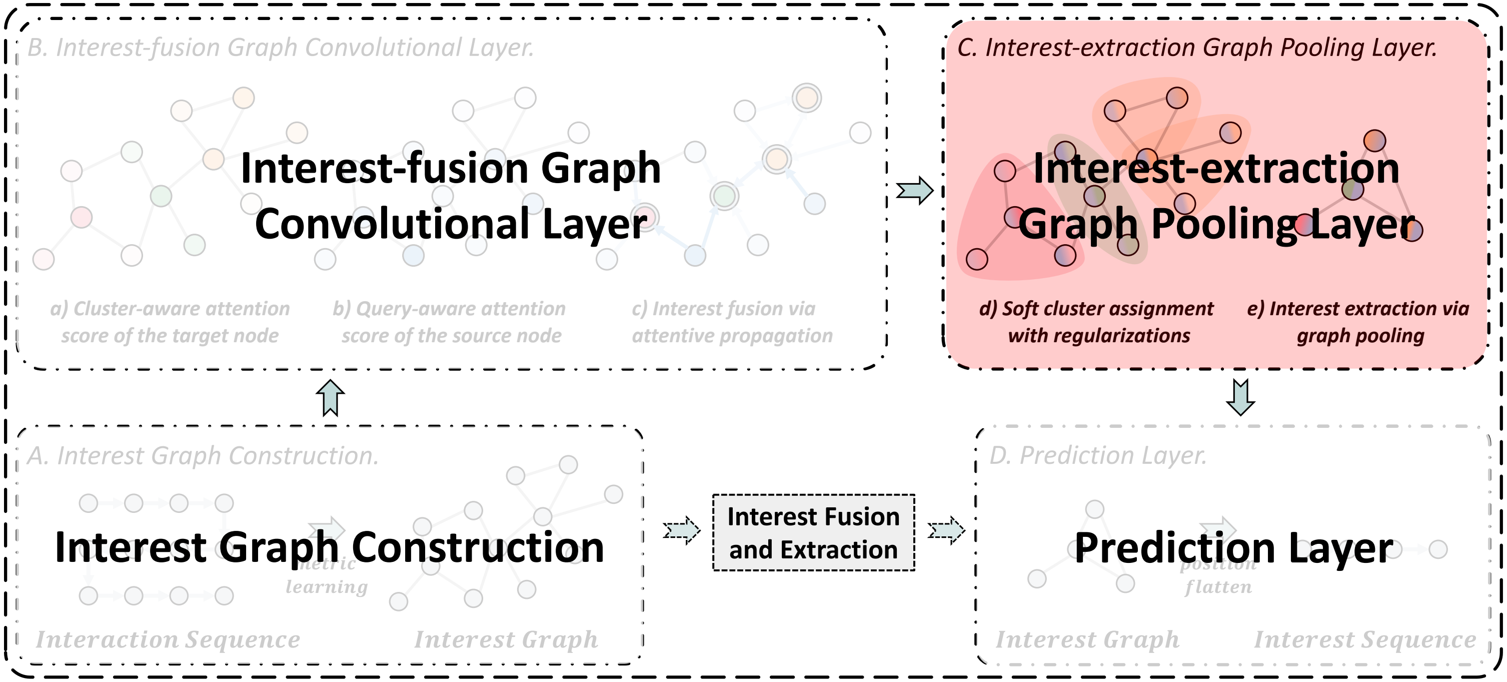
Interest Graph

Interest Sequence

SURGE Model Framework

- Interest-extraction Graph Pooling Layer

Solve Challenge 2: by adaptively reserving dynamically-activated core preferences.



B. Interest-fusion Graph Convolutional Layer.

Interest-fusion Graph Convolutional Layer

a) Cluster-aware attention score of the target node

b) Query-aware attention score of the source node

c) Interest fusion via attentive propagation

C. Interest-extraction Graph Pooling Layer.

Interest-extraction Graph Pooling Layer

d) Soft cluster assignment with regularizations

e) Interest extraction via graph pooling

A. Interest Graph Construction.

Interest Graph Construction

metric learning

Interaction Sequence

Interest Graph

Interest Fusion and Extraction

D. Prediction Layer.

Prediction Layer

position flatten

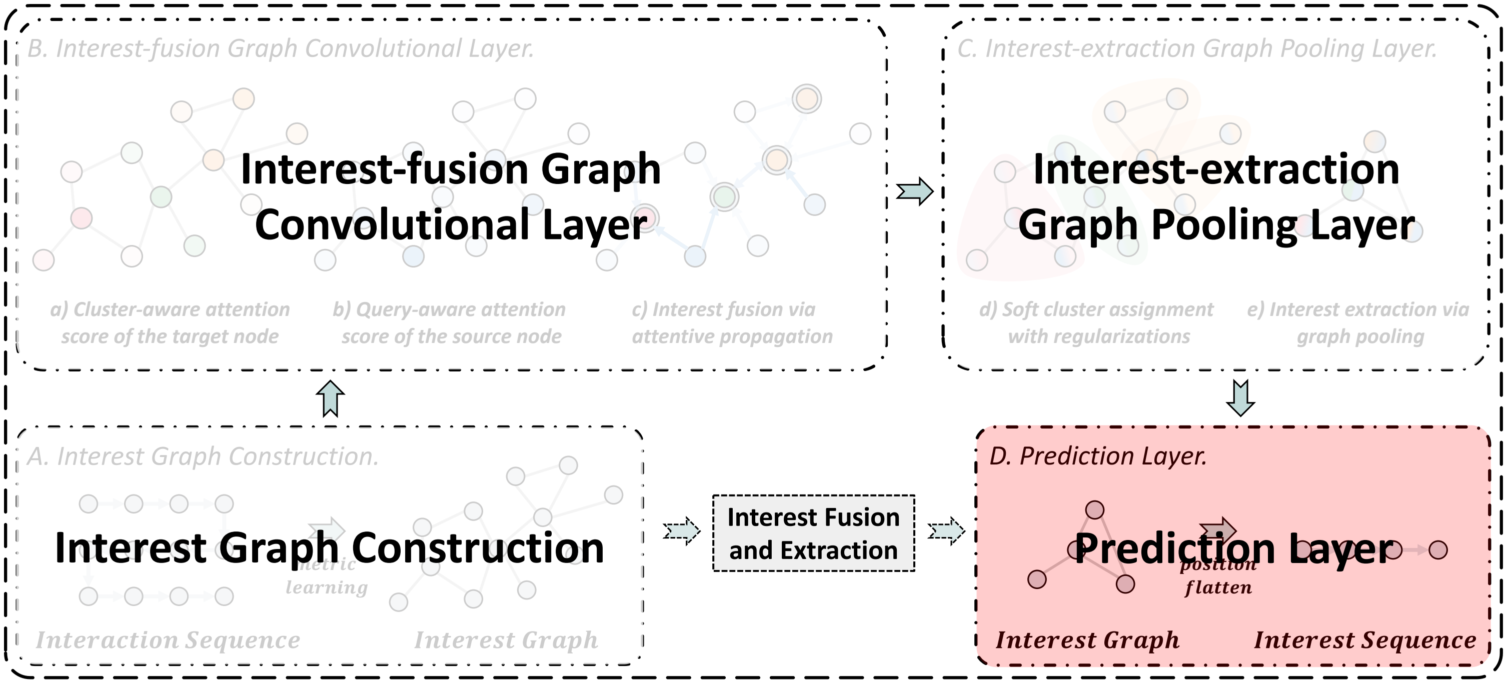
Interest Graph

Interest Sequence

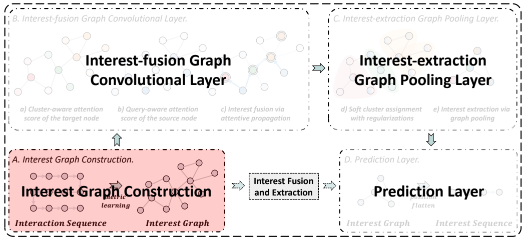
SURGE Model Framework

- ❑ Prediction Layer

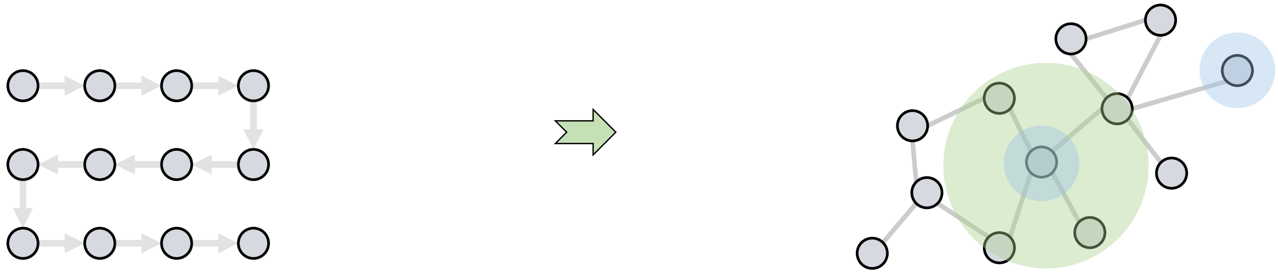
Solve Challenge 2: by modeling evolution on reduced sequence flattened from pooled graph.


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
metric learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
position flatten
Interest Graph
Interest Sequence

Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
metric learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
position flatten
Interest Graph
Interest Sequence

❑ A. Interest Graph Construction.



Interaction Sequence

Each interacted item is converted to an **vertex** $\boldsymbol{v} \in \boldsymbol{V}$ with $|\boldsymbol{V}| = \boldsymbol{n}$

Interest Graph

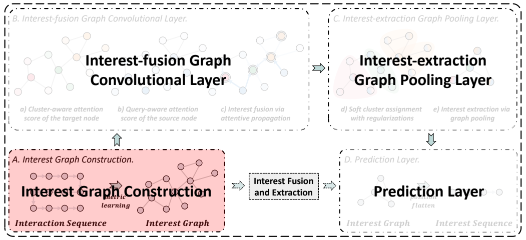
Each **edge** $(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{A}_{\boldsymbol{i},\boldsymbol{j}}) \in \boldsymbol{E}$ indicates whether item $\boldsymbol{i}$ is related to item $\boldsymbol{j}$.

- **1. Raw graph construction:**
  - learns an undirected **graph** $\boldsymbol{G} = \{\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{A}\}$ for each interaction sequence.

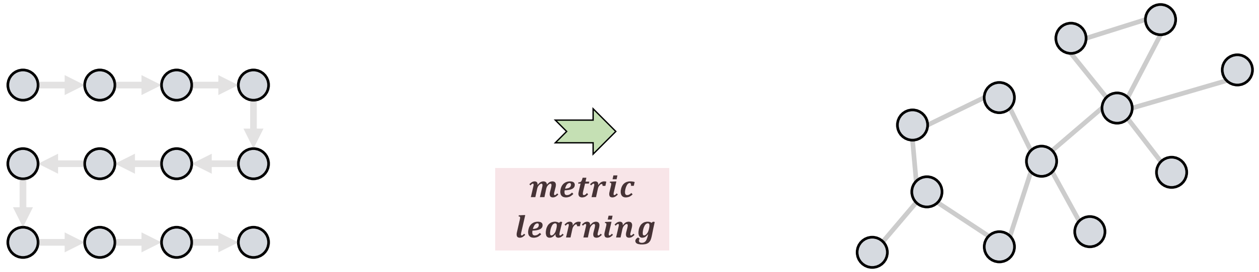
It is easier to distinguish users' core and peripheral interests.

The core interest nodes have higher degree and form denser subgraph.

Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
metric learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
flatten
Interest Graph
Interest Sequence

❑ A. Interest Graph Construction.



Interaction Sequence

Interest Graph

Each interacted item is converted to an **vertex** $\boldsymbol{v} \in \boldsymbol{V}$ with $|\boldsymbol{V}| = \boldsymbol{n}$

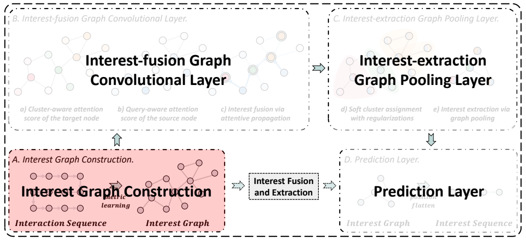
Each **edge** $(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{A}_{\boldsymbol{i},\boldsymbol{j}}) \in \boldsymbol{E}$ indicates whether item $\boldsymbol{i}$ is related to item $\boldsymbol{j}$.

- **2. Node similarity metric learning:**
  - Metric function: $M_{ij}^{\delta} = \cos(\vec{w}_{\delta} \odot \vec{h}_i, \vec{w}_{\delta} \odot \vec{h}_j)$, $M_{ij} = \frac{1}{\delta} \sum_{\delta=1}^{\phi} M_{ij}^{\delta}$,

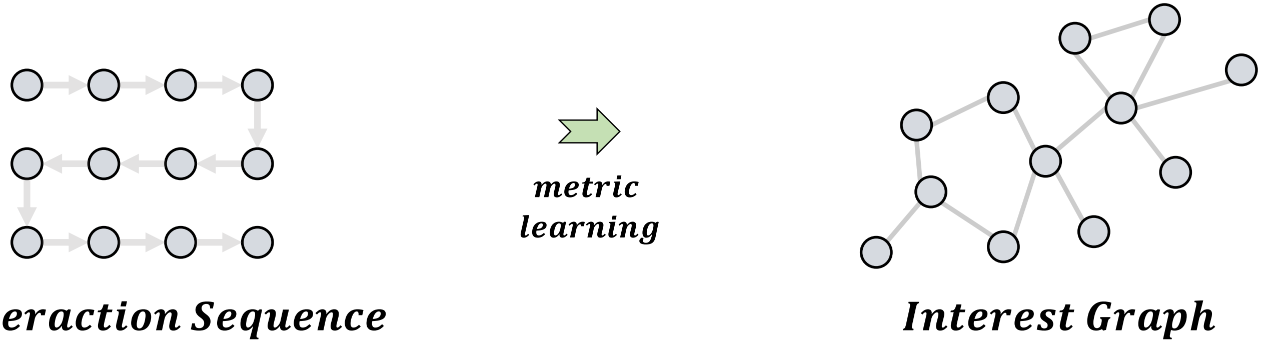
Multi-head metric increases the expressive power and stabilize the learning process.

Trainable weight $\vec{w}$ adaptively highlights different dimensions.

Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
metric learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
position flatten
Interest Graph
Interest Sequence

❑ A. Interest Graph Construction.



Each interacted item is converted to an **vertex** $\boldsymbol{v} \in \boldsymbol{V}$ with $|\boldsymbol{V}| = \boldsymbol{n}$

Each **edge** $(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{A}_{i,j}) \in \boldsymbol{E}$ indicates whether item $\boldsymbol{i}$ is related to item $\boldsymbol{j}$.

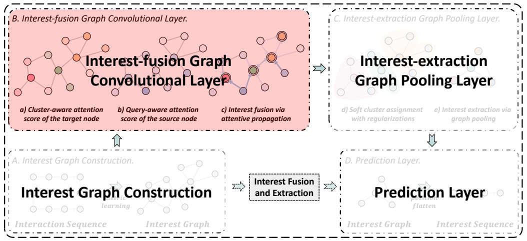
- **3. Graph sparsification via ε-sparseness:**
  - Relative ranking strategy of the entire graph:

$$A_{ij} = \begin{cases} 1, & M_{ij} >= \text{Rank}_{\varepsilon n^2}(M); \\ 0, & \text{otherwise}; \end{cases}$$

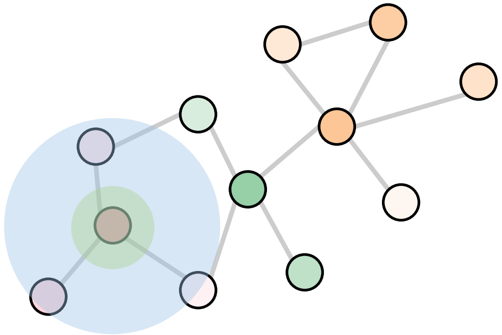
Rank **returns the value of the εn^2-th largest value in the metric matrix M.**

$\boldsymbol{n}$ is the number of nodes and ε controls the overall sparsity.

Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
metric learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
position flatten
Interest Graph
Interest Sequence

❑ B. Interest-fusion Graph Convolutional Layer



a) Cluster-aware attention score of the target node

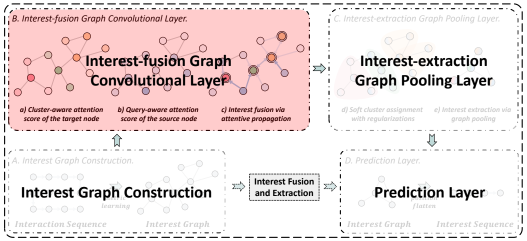
- **1. Cluster-aware attention:**
  - identifies whether the target node is the center of the cluster.

$$\alpha_i = \mathbf{Attention_c}(\mathbf{W_c}\vec{h}_i \,||\, \vec{h}_{i_c} \,||\, \mathbf{W_c}\vec{h}_i \odot \vec{h}_{i_c}),$$

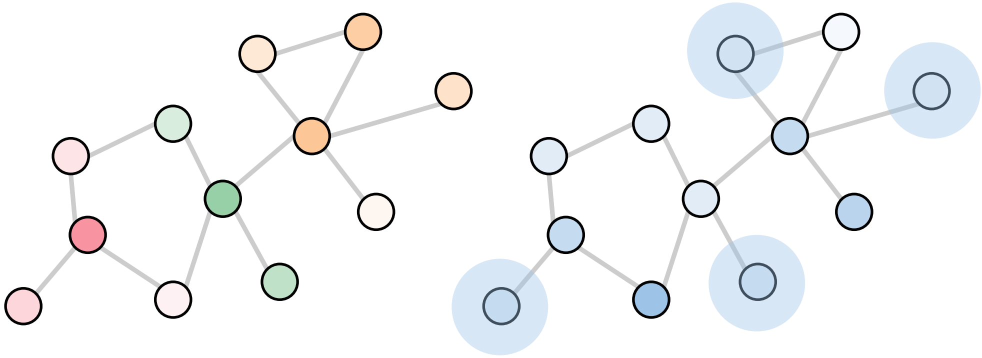
The target node v_i is regarded as a medoid of a cluster $c(v_i)$.

k-hop neighborhood of the target node v_i is the receptive field of the cluster $c(v_i)$.

Methodology


Interest-fusion Graph Convolutional Layer
Interest-extraction Graph Pooling Layer
Interest Graph Construction
Interest Fusion and Extraction
Prediction Layer

❑ B. Interest-fusion Graph Convolutional Layer



a) Cluster-aware attention score of the target node

b) Query-aware attention score of the source node

▪ 1. Query-aware attention:

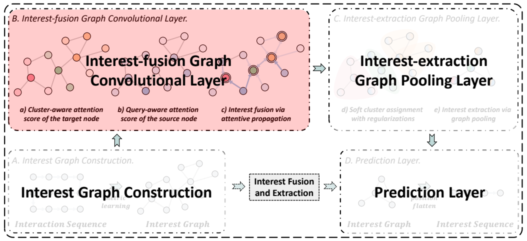
- identifies interests' independent evolution for different target item.

$$\beta_j = \text{Attention}_q(\mathbf{W}_\mathbf{q}\vec{h}_j \,\|\, \vec{h}_t \,\|\, \mathbf{W}_\mathbf{q}\vec{h}_j \odot \vec{h}_t),$$

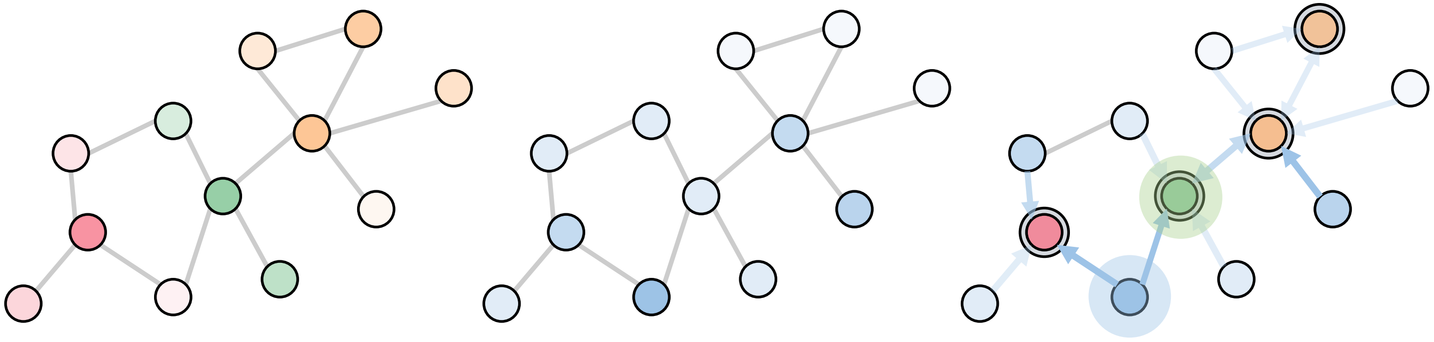
Only relevants with target item x_t can play a role in the prediction.

Irrelevant source node v_j information will be discarded during aggregation.

Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
metric learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
position flatten
Interest Graph
Interest Sequence

❑ B. Interest-fusion Graph Convolutional Layer



a) Cluster-aware attention score of the target node

b) Query-aware attention score of the source node

c) Interest fusion via attentive propagation

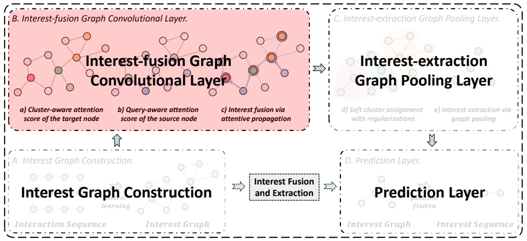
- **1. Cluster- and query-aware attention:**
  - maps the importance of target node v_i on it’s neighbor source node v_j.

$$E_{ij} = \mathrm{softmax}_j(\alpha_i + \beta_j) = \frac{\exp(\alpha_i + \beta_j)}{\sum_{k \in \mathcal{N}_i} \exp(\alpha_i + \beta_k)},$$

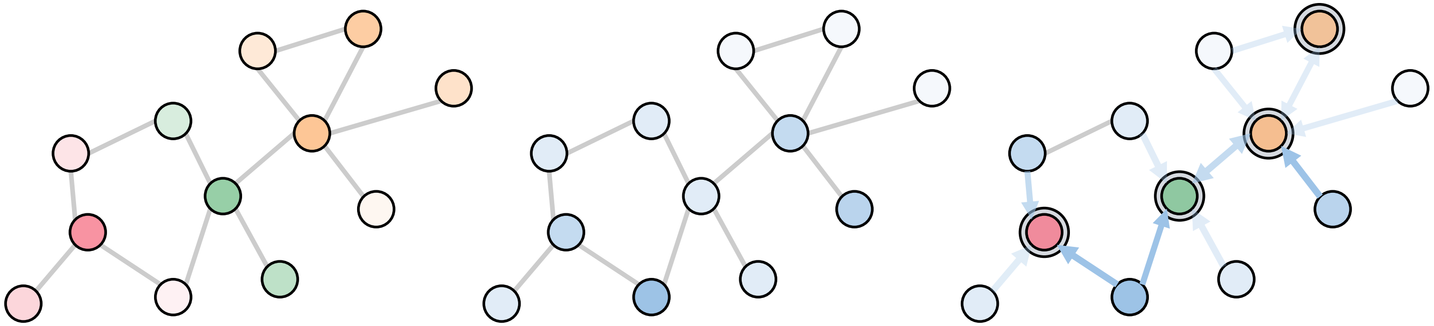
α_i controls how much information the target node v_i can receive.

β_j controls how much information the source node v_j can send.

Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
Interest Graph
Interest Sequence

❑ B. Interest-fusion Graph Convolutional Layer



a) Cluster-aware attention score of the target node

b) Query-aware attention score of the source node

c) Interest fusion via attentive propagation

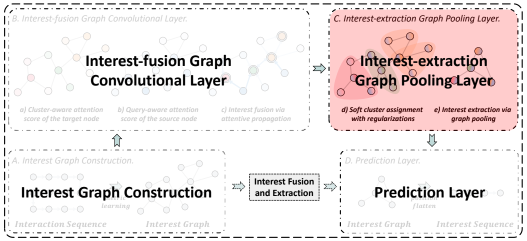
- **1. Interest fusion via graph attentive convolution:**
  - refines output embeddings by gathering weak signals to strong ones.

$$\vec{h}_i' = \Big\|_{\delta=1}^{\phi} \sigma\left(\mathrm{W_a}^{\delta} \cdot \mathrm{Aggregate}\left(E_{ij}^{\delta} * \vec{h}_j | j \in \mathcal{N}_i\right) + \vec{h}_i\right),$$

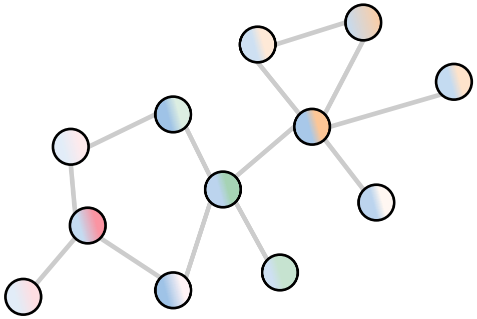
Multi-head attention mechanism increases the expressive power.

$E_{i,j}$ perceives users' core interest and the interest related to query interest.

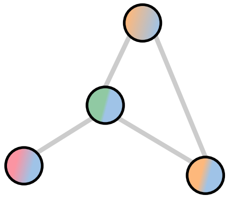
Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
metric learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
position flatten
Interest Graph
Interest Sequence

❑ C. Interest-extraction Graph Pooling Layer



d) Soft cluster assignment with regularizations



e) Interest extraction via graph pooling

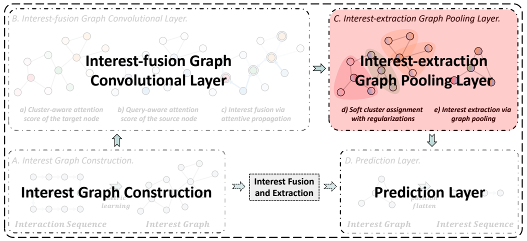
- **1. Interest extraction via graph pooling :**
  - downsizes the graph reasonably to further extract the fused interest.

$$\{\vec{h}_1^*, \vec{h}_2^*, \ldots, \vec{h}_m^*\} = S^T \{\vec{h}_1', \vec{h}_2', \ldots, \vec{h}_n'\},$$
$$\{\gamma_1^*, \gamma_2^*, \ldots, \gamma_m^*\} = S^T \{\gamma_1, \gamma_2, \ldots, \gamma_n\},$$

Cluster assignment matrix $S \in R^{n \times m}$ pools node embedding $\vec{h}'_i$ and score γ_i into cluster.

n loose interests are transformed into m tight interests and their distribution is maintained.

Methodology



Interest-fusion Graph Convolutional Layer

Interest-extraction Graph Pooling Layer

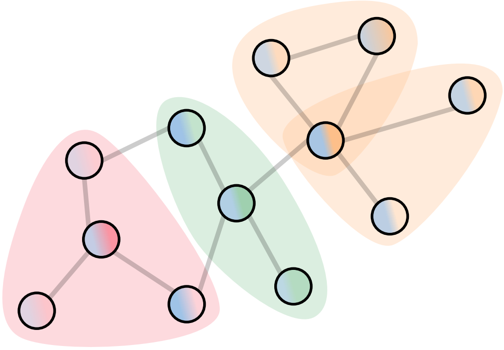
Interest Graph Construction

Interest Fusion and Extraction

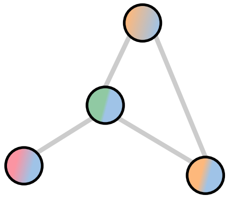
Prediction Layer

B. Interest-fusion Graph Convolutional Layer.
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
metric learning
Interaction Sequence
Interest Graph
D. Prediction Layer.
position flatten
Interest Graph
Interest Sequence

❑ C. Interest-extraction Graph Pooling Layer



d) Soft cluster assignment with regularizations



e) Interest extraction via graph pooling

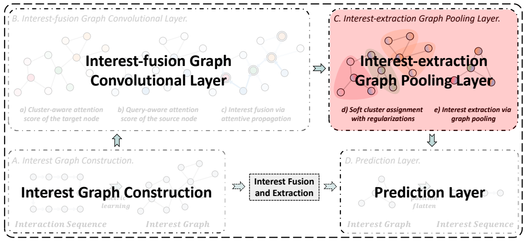
- **1. Interest extraction via graph pooling :**
  - uses the GNN architecture to generate the assignment matrix.

$$S_{i:} = \text{softmax}\left(\mathbf{W}_{\text{p}} \cdot \textbf{Aggregate}\left(A_{ij} * \vec{h}'_j | j \in \mathcal{N}_i\right)\right),$$

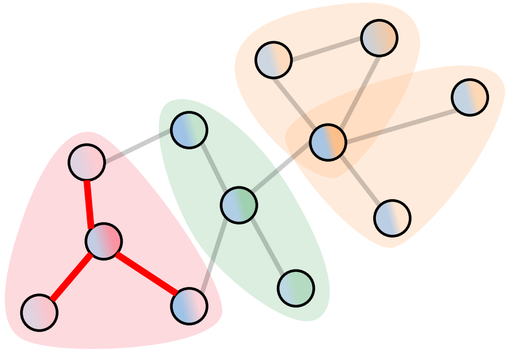
Softmax is used to obtain the probability of the i-th node being divided into m clusters.

The output dimension of weight W_p is set as the number of clusters m.

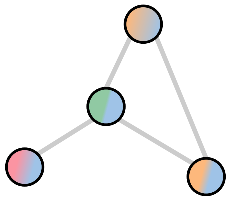
Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
Interest Graph
Interest Sequence

❑ C. Interest-extraction Graph Pooling Layer



d) Soft cluster assignment with regularizations



e) Interest extraction via graph pooling

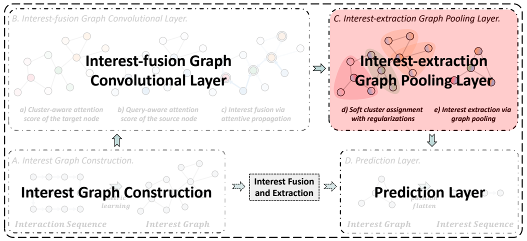
- **2. Assignment regularization :**
  - Same mapping regularization with Frobenius norm.

$$L_{\mathrm{M}} = \|A, SS^T\|_F,$$

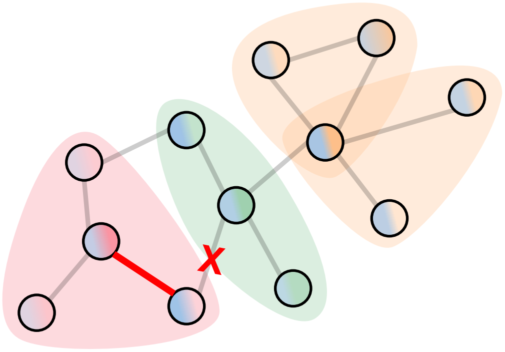
Each element in A represents the connection strength between two nodes.

Each element in SS^T represents the probability that two nodes are divided to the same cluster.

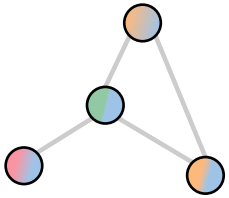
Methodology


C. Interest-extraction Graph Pooling Layer.
Interest-fusion Graph Convolutional Layer
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
Interest Graph Construction
Interest Fusion and Extraction
Prediction Layer

❑ C. Interest-extraction Graph Pooling Layer



d) Soft cluster assignment with regularizations



e) Interest extraction via graph pooling

▪ 2. Assignment regularization :

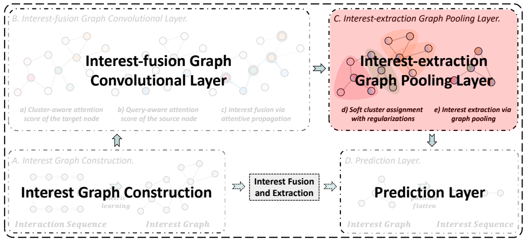
- Single affiliation regularization with entropy function.

$$L_{\mathrm{A}} = \frac{1}{n}\sum_{i=1}^{n} H\left(S_{i:}\right),$$

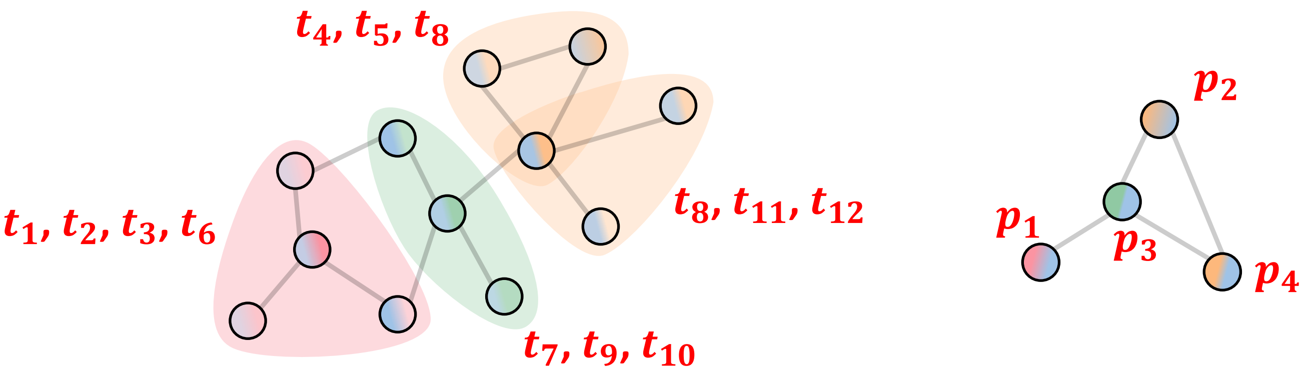
It makes each row $S_{i:}$ in assignment matrix approach a one-hot vector.

$H(\cdot)$ is the entropy function that can reduce the uncertainty of the mapping distribution.

Methodology


B. Interest-fusion Graph Convolutional Layer
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction
Interest Graph Construction
learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer
Prediction Layer
flatten
Interest Graph
Interest Sequence

❑ C. Interest-extraction Graph Pooling Layer



d) Soft cluster assignment with regularizations

e) Interest extraction via graph pooling

▪ 2. Assignment regularization :

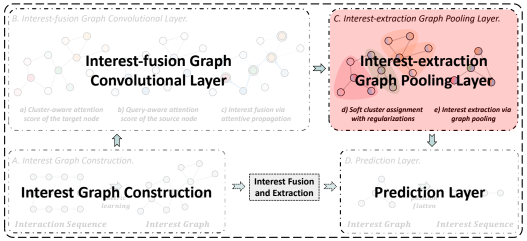
- Relative position regularization with L2 norm.

$$L_{\mathrm{P}} = \|P_n S, P_m\|_2,$$

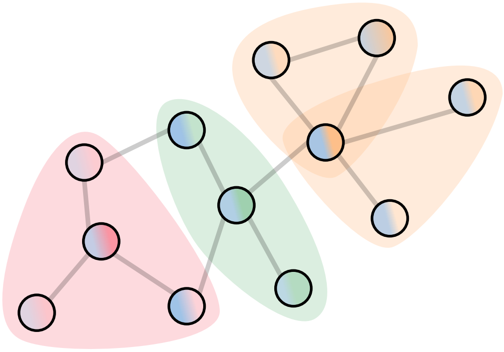
It makes the position of the non-zero elements in S closer to the main diagonal elements.

P_n and P_m are position encoding vectors, like $\{1, 2, \ldots, n\}$ and $\{1, 2, \ldots, m\}$.

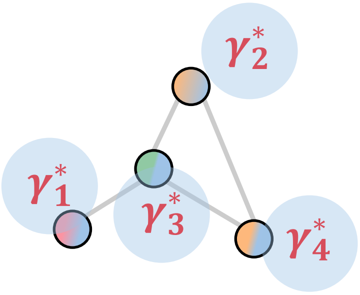
Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
metric learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
position flatten
Interest Graph
Interest Sequence

❑ C. Interest-extraction Graph Pooling Layer



d) Soft cluster assignment with regularizations



e) Interest extraction via graph pooling

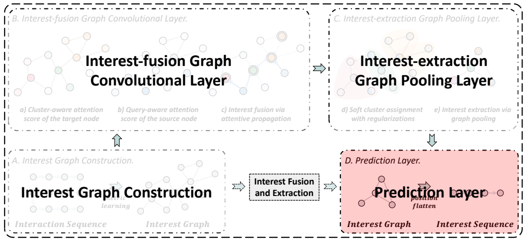
- **3. Graph readout :**
  - feeds graph-level representation into the final prediction layer.

$$\vec{h}_g = \mathbf{Readout}(\{\gamma_i * \vec{h}'_i, i \in \mathcal{G}\}),$$

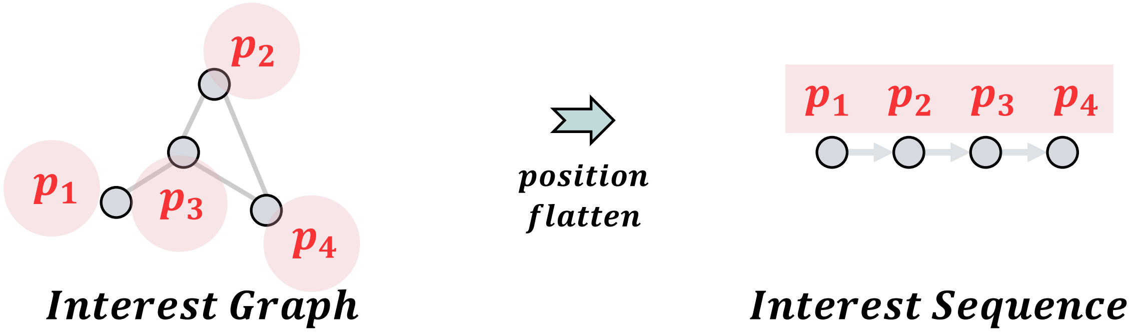
The constrains of node information can better extract each cluster's importance.

The weighted readout on raw graph constrains each node's importance.

Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
A. Interest Graph Construction.
Interest Graph Construction
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
Interaction Sequence
Interest Graph
Interest Graph
Interest Sequence

❑ D. Prediction Layer



The relative position regularization avoids the time order's bias.

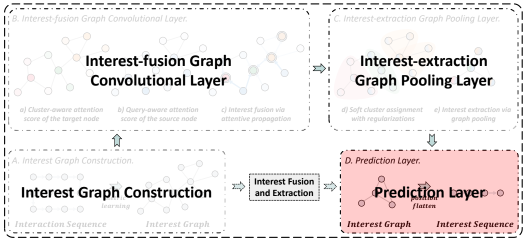
- **1. Interest evolution modeling :**
  - supplies the final interest with more relative historical information;

$$\vec{h}_s = \text{AUGRU}(\{\vec{h}_1^*, \vec{h}_2^*, \ldots, \vec{h}_m^*\}).$$

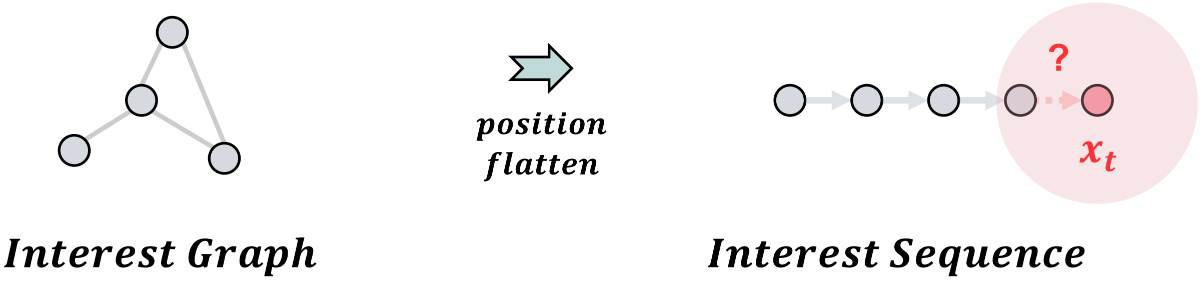
It is easier to model evolution on reduced sequence with enhanced interest signals.

AUGRU uses cluster score γ_i^* to scale all dimensions of the update gate in GRU.

Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
metric learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
position flatten
Interest Graph
Interest Sequence

❑ D. Prediction Layer



The probability of the user interacting with next item is estimated.

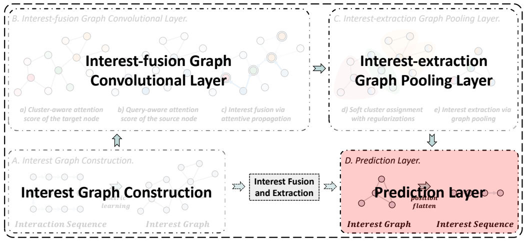
- **2. Prediction :**
  - uses MLPS to automatically learn the combination of embeddings;

$$\hat{y} = \text{Predict}(\vec{h}_s \| \vec{h}_g \| \vec{h}_t \| \vec{h}_g \odot \vec{h}_t).$$

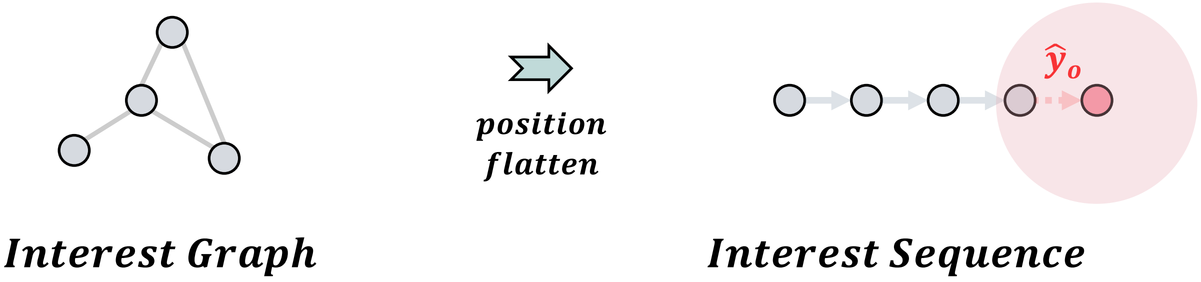
Graph representation, evolution output and target item embedding are considered.

Two-layer feedforward neural network is used as the prediction function.

Methodology


B. Interest-fusion Graph Convolutional Layer.
Interest-fusion Graph Convolutional Layer
a) Cluster-aware attention score of the target node
b) Query-aware attention score of the source node
c) Interest fusion via attentive propagation
C. Interest-extraction Graph Pooling Layer.
Interest-extraction Graph Pooling Layer
d) Soft cluster assignment with regularizations
e) Interest extraction via graph pooling
A. Interest Graph Construction.
Interest Graph Construction
metric learning
Interaction Sequence
Interest Graph
Interest Fusion and Extraction
D. Prediction Layer.
Prediction Layer
position flatten
Interest Graph
Interest Sequence

❑ D. Prediction Layer



$\hat{y}_o$ stands for the network's output after the softmax layer.

- **3. Optimization objective :**
  - uses the negative log-likelihood function as the loss function;

$$L = -\frac{1}{|\mathcal{O}|}\sum_{o\in\mathcal{O}}\left(y_o \log \hat{y}_o + (1 - y_o)\log(1-\hat{y}_o)\right) + \lambda\|\Theta\|_2,$$

L2 regularization is used to prevent over-fitting and λ controls the penalty strength.

$y_o = 1$ indicates positive instances and $y_o = 0$ indicates negative ones.

Experiment Settings

❑ Datasets

- E-commerce Platform: **Taobao**
- Short-video Platform: **Kuaishou**

淘宝网 Taobao.com 淘

快手 KUAISHOU TECHNOLOGY

❑ Evaluation Metrics:

- Accuracy Metrics: **AUC, GAUC**
- Ranking Metrics: **MRR, NDCG@K**

❑ Dataset partition:

- Taobao: **2017.11. 25 ~ 2017.12.3**, first 7 days as training set, the 8th day as validation set, and the last day as test set.
- Kuaishou: **2020.10.11 ~ 2020.10.28**, first 6 days as training set, before 12 pm of the last day as validation set, and after 12 pm as test set.

Dataset	Users	Items	Instances	Average Length
Taobao	36,915	64,138	1,471,155	39.85
Kuaishou	60,813	292,286	14,952,659	245.88

Compared Methods

- **NCF** [He et al. WWW'17]
  - matrix factorization and multilayer perceptrons
- **DIN** [Zhou et al. KDD'18]
  - attention mechanism with target item as query
- **LightGCN** [Zhou et al. SIGIR'20]
  - uses GCN to extract higher-order connectivity

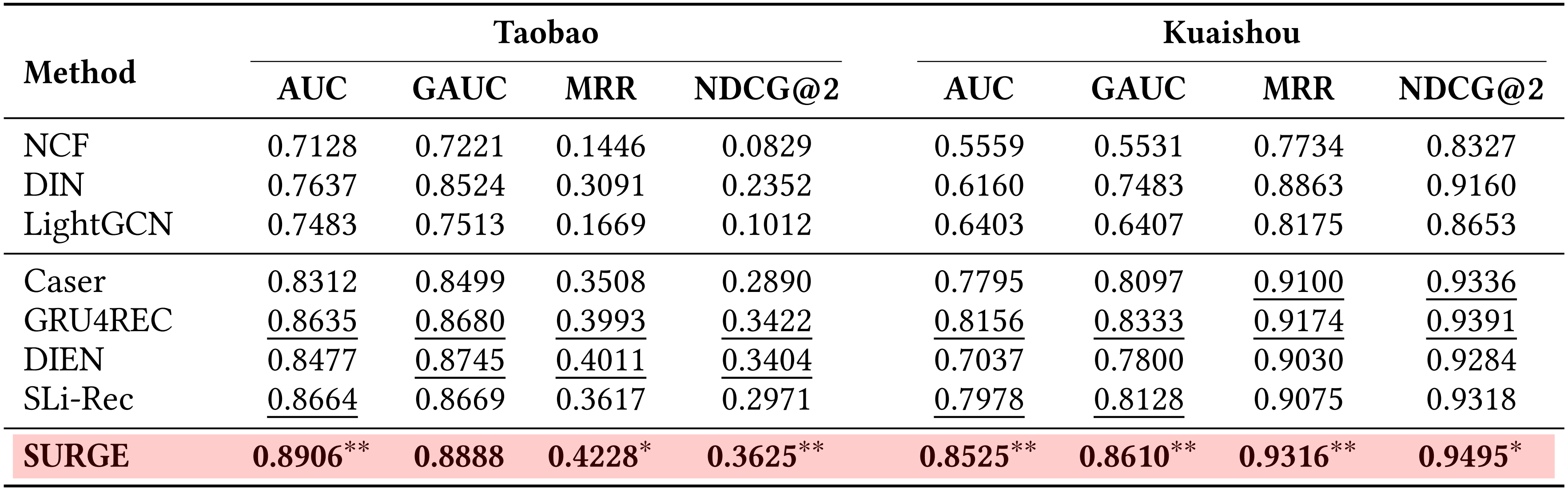
General Models

- **GRU4REC** [Hidasi et al. ICLR'16]
  - encodes user interest into GRU’s final state
- **Caser** [Tang et al. WSDM'18]
  - uses CNN to learn sequence patterns
- **DIEN** [Zhou et al. AAAI'19]
  - interest extraction and evolution GRUs
- **SLi-Rec** [Yu et al. IJCAI’19]
  - jointly models long and short-term interests

Sequential Models

Experiments

❑ Overal Performance



Method	Taobao				Kuaishou			
	AUC	GAUC	MRR	NDCG@2	AUC	GAUC	MRR	NDCG@2
NCF	0.7128	0.7221	0.1446	0.0829	0.5559	0.5531	0.7734	0.8327
DIN	0.7637	0.8524	0.3091	0.2352	0.6160	0.7483	0.8863	0.9160
LightGCN	0.7483	0.7513	0.1669	0.1012	0.6403	0.6407	0.8175	0.8653
Caser	0.8312	0.8499	0.3508	0.2890	0.7795	0.8097	0.9100	0.9336
GRU4REC	0.8635	0.8680	0.3993	0.3422	0.8156	0.8333	0.9174	0.9391
DIEN	0.8477	0.8745	0.4011	0.3404	0.7037	0.7800	0.9030	0.9284
SLi-Rec	0.8664	0.8669	0.3617	0.2971	0.7978	0.8128	0.9075	0.9318
SURGE	**0.8906****	**0.8888**	**0.4228***	**0.3625****	**0.8525****	**0.8610****	**0.9316****	**0.9495***

Performance comparisons (bold means p-value < 0.05, bold* means p-value < 0.01, and bold** means p-value < 0.001.)

1. Our proposed method consistently achieves the best performance;

Experiments

□ Overal Performance

Method	Taobao				Kuaishou			
	AUC	GAUC	MRR	NDCG@2	AUC	GAUC	MRR	NDCG@2
NCF	0.7128	0.7221	0.1446	0.0829	0.5559	0.5531	0.7734	0.8327
DIN	0.7637	0.8524	0.3091	0.2352	0.6160	0.7483	0.8863	0.9160
LightGCN	0.7483	0.7513	0.1669	0.1012	0.6403	0.6407	0.8175	0.8653
Caser	0.8312	0.8499	0.3508	0.2890	0.7795	0.8097	0.9100	0.9336
GRU4REC	0.8635	0.8680	0.3993	0.3422	0.8156	0.8333	0.9174	0.9391
DIEN	0.8477	0.8745	0.4011	0.3404	0.7037	0.7800	0.9030	0.9284
SLi-Rec	0.8664	0.8669	0.3617	0.2971	0.7978	0.8128	0.9075	0.9318
SURGE	**0.8906****	**0.8888**	**0.4228***	**0.3625****	**0.8525****	**0.8610****	**0.9316****	**0.9495***

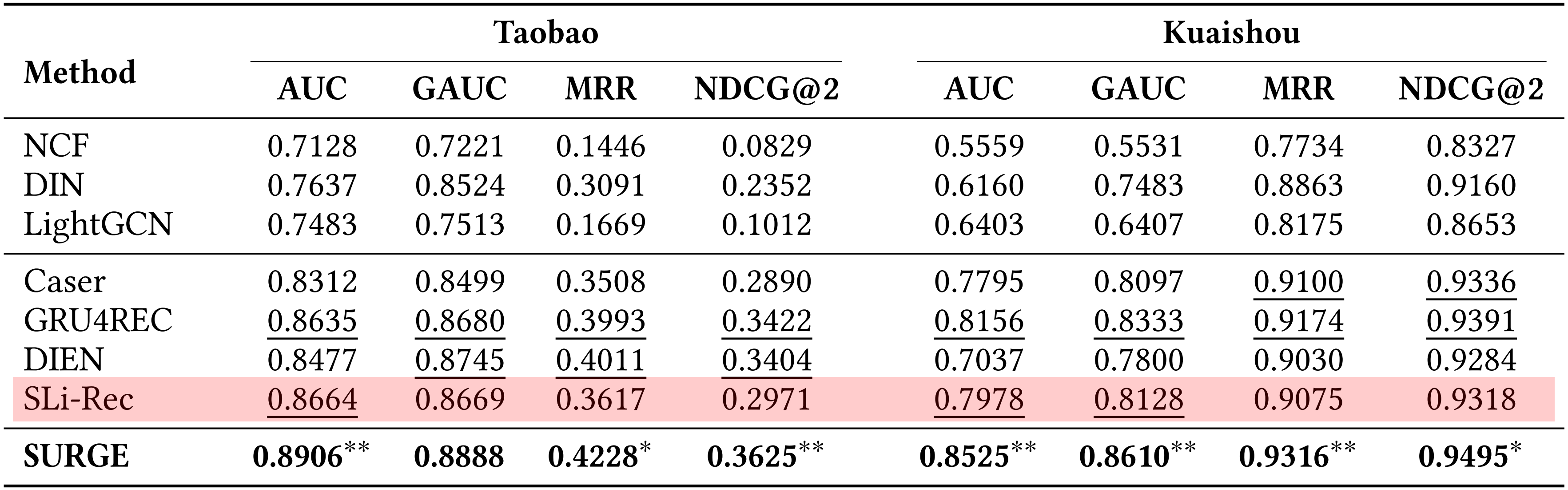
Performance comparisons (bold means p-value < 0.05, bold* means p-value < 0.01, and bold** means p-value < 0.001.)

1. Our proposed method consistently achieves the best performance;

2. Sequential models are effective but have a short-term bottleneck;

Experiments

❑ Overal Performance



Method	Taobao				Kuaishou			
	AUC	GAUC	MRR	NDCG@2	AUC	GAUC	MRR	NDCG@2
NCF	0.7128	0.7221	0.1446	0.0829	0.5559	0.5531	0.7734	0.8327
DIN	0.7637	0.8524	0.3091	0.2352	0.6160	0.7483	0.8863	0.9160
LightGCN	0.7483	0.7513	0.1669	0.1012	0.6403	0.6407	0.8175	0.8653
Caser	0.8312	0.8499	0.3508	0.2890	0.7795	0.8097	0.9100	0.9336
GRU4REC	0.8635	0.8680	0.3993	0.3422	0.8156	0.8333	0.9174	0.9391
DIEN	0.8477	0.8745	0.4011	0.3404	0.7037	0.7800	0.9030	0.9284
SLi-Rec	0.8664	0.8669	0.3617	0.2971	0.7978	0.8128	0.9075	0.9318
SURGE	**0.8906****	**0.8888**	**0.4228***	**0.3625****	**0.8525****	**0.8610****	**0.9316****	**0.9495***

Performance comparisons (bold means p-value < 0.05, bold* means p-value < 0.01, and bold** means p-value < 0.001.)

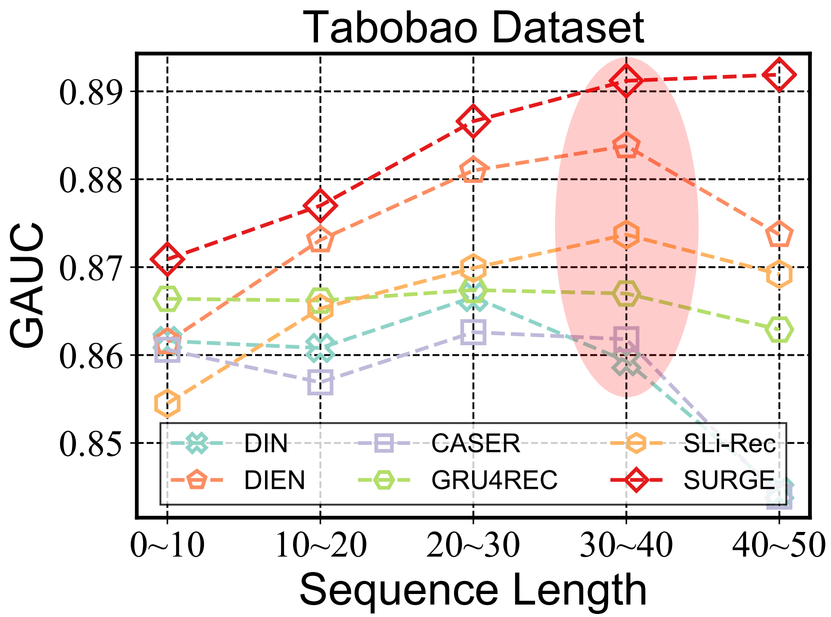
1. Our proposed method consistently achieves the best performance;

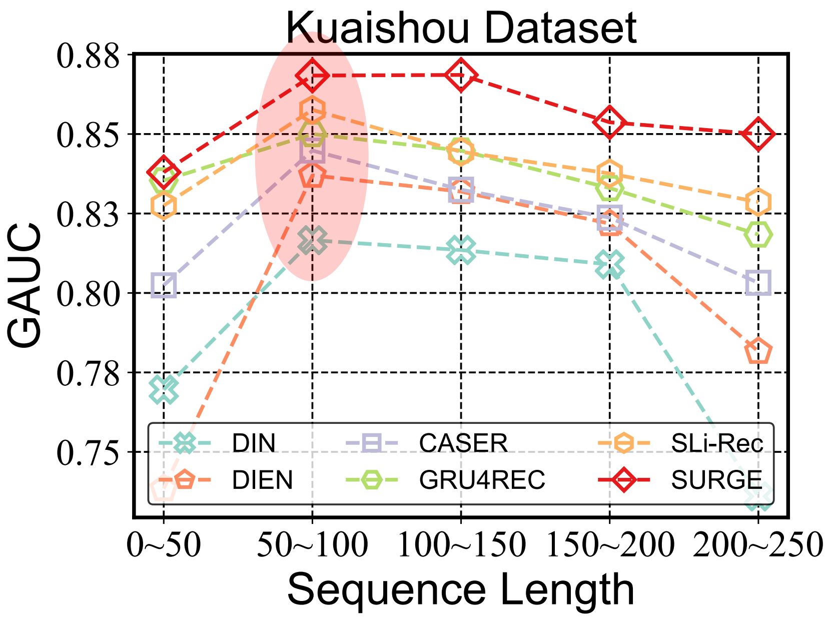
2. Sequential models are effective but have a short-term bottleneck;

3. Joint modeling long and short-term interests are not always better.

Experiments

❑ Study on Sequence Length

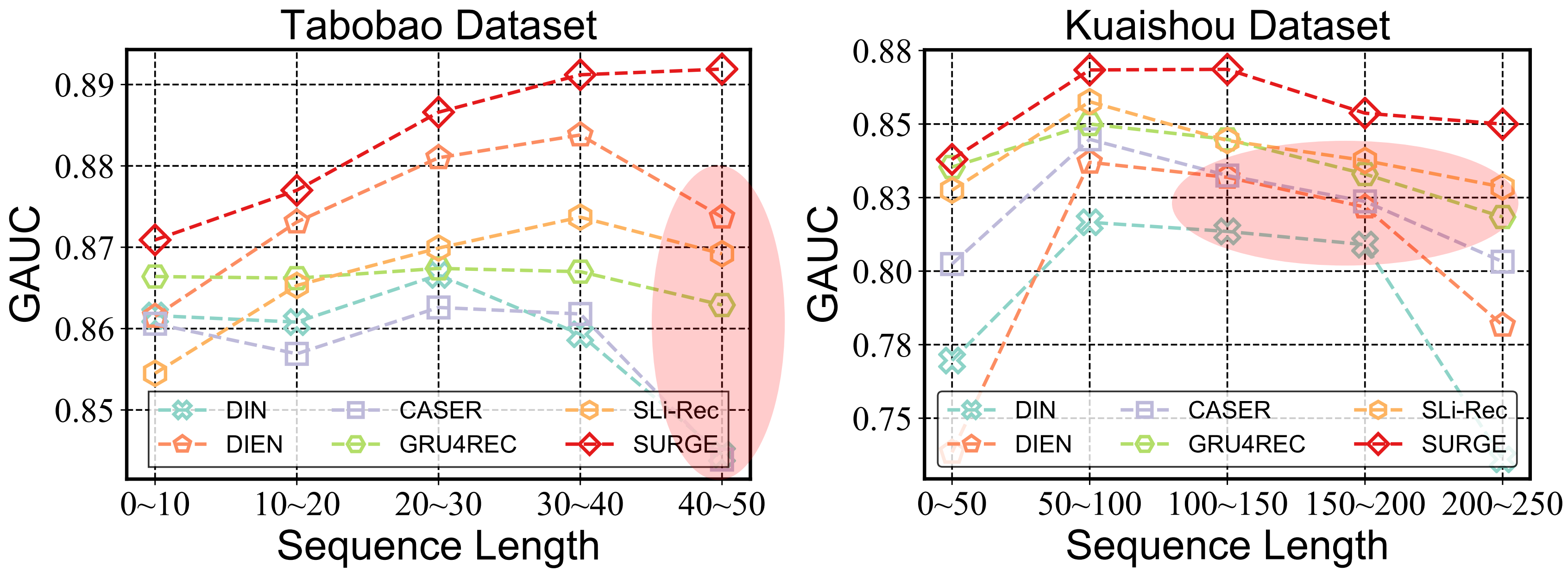




1. As the length increases, each model's performance reaches its peak;

Experiments

❑ Study on Sequence Length

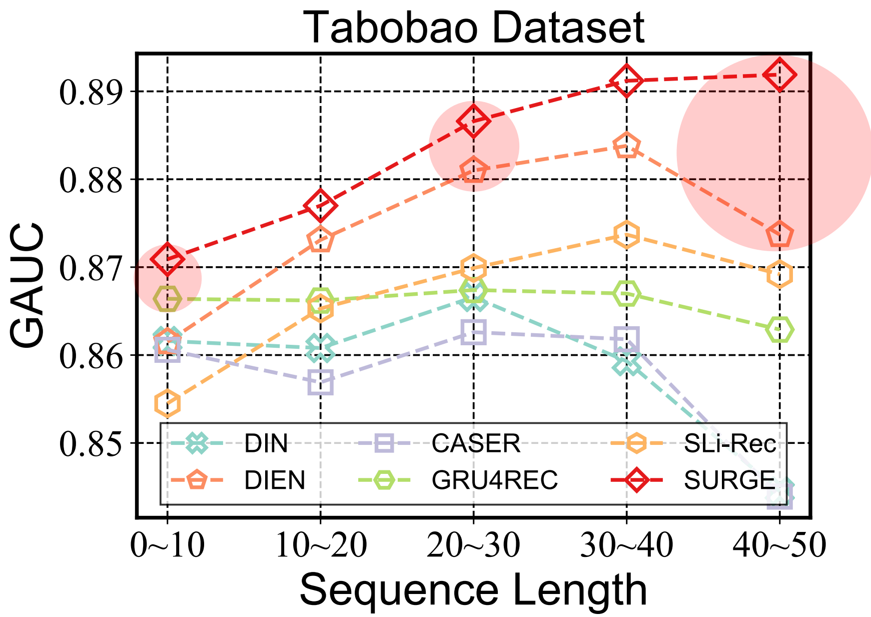


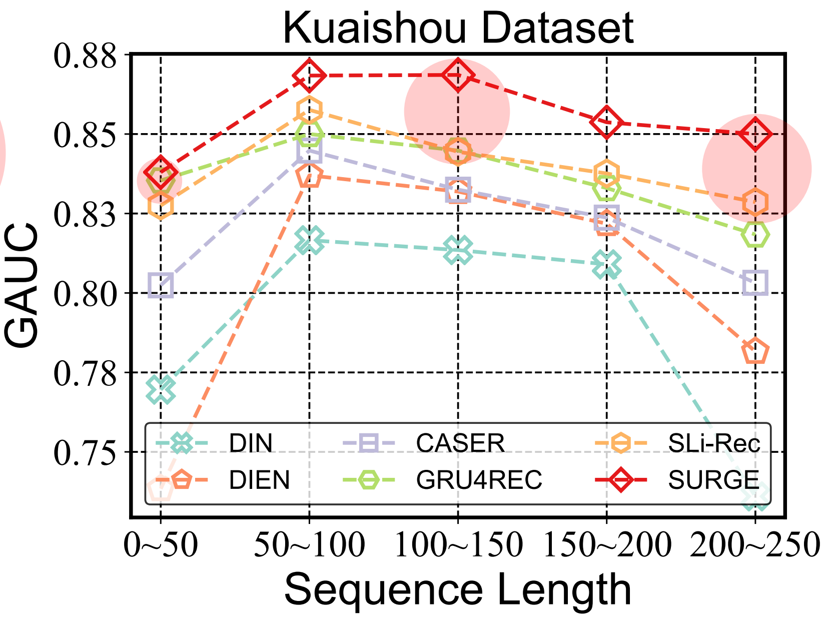
1. As the length increases, each model's performance reaches its peak;

2. As the length continues to increase, most models' performance decline;

Experiments

❑ Study on Sequence Length





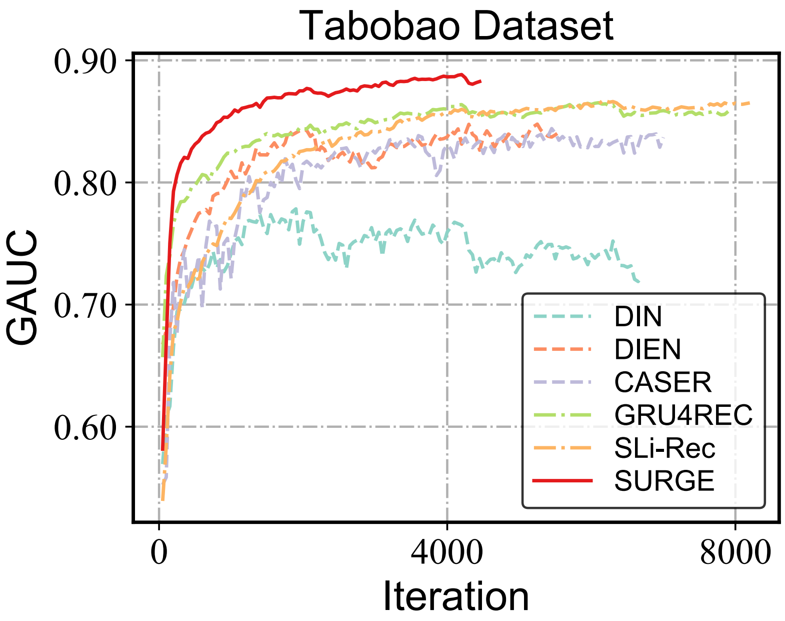
1. As the length increases, each model's performance reaches its peak;

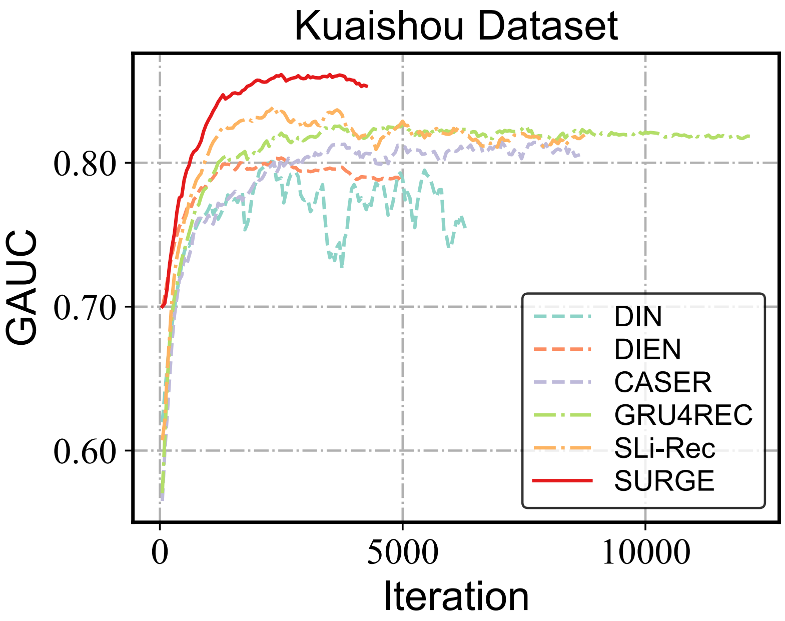
2. As the length continues to increase, most models' performance decline;

3. The performance gap with SURGE is larger when sequences become longer.

Experiments

❑ Efficiency Comparison



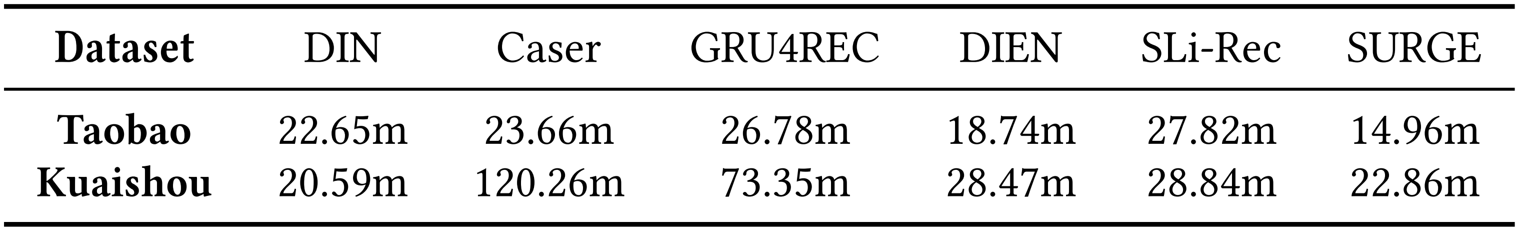


SURGE's another advantage is that the convergence process during training is more stable and fast.

- Other methods either **continually fluctuate** and are difficult to converge, or **increase slowly** and are difficult to stop early.

Experiments

❑ Efficiency Comparison



Dataset	DIN	Caser	GRU4REC	DIEN	SLi-Rec	SURGE
Taobao	22.65m	23.66m	26.78m	18.74m	27.82m	14.96m
Kuaishou	20.59m	120.26m	73.35m	28.47m	28.84m	22.86m

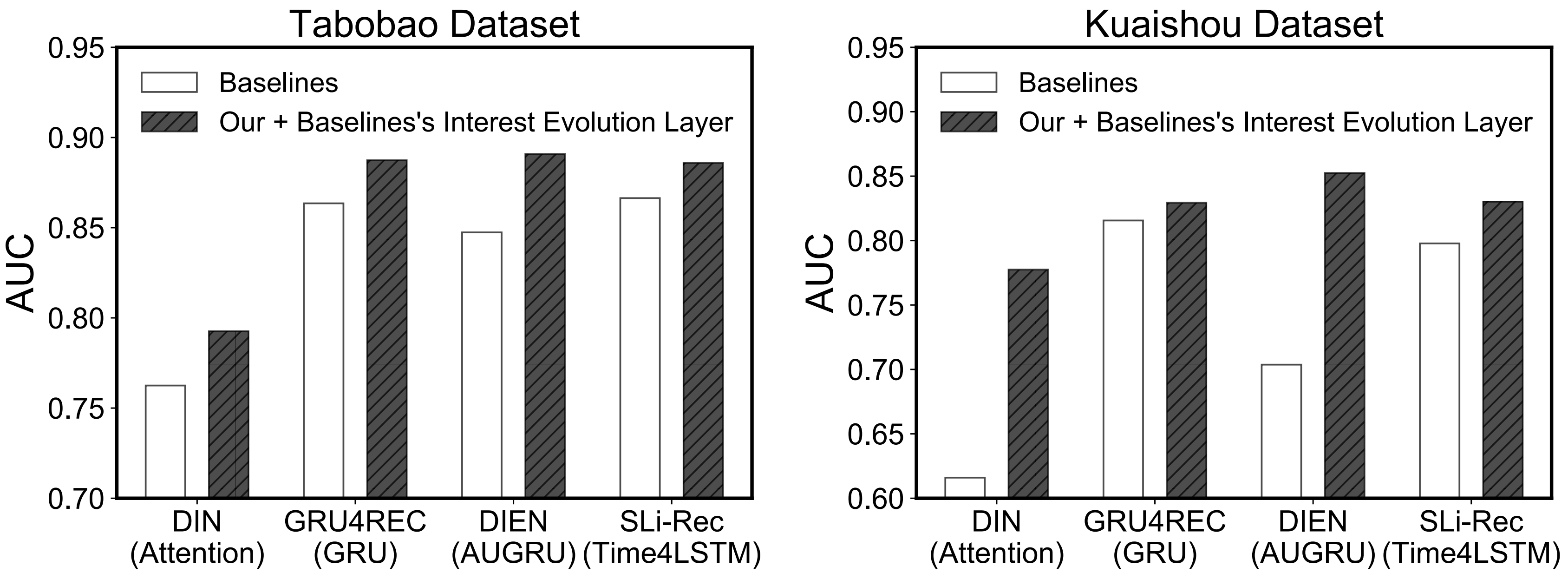
Total training time until convergence of baselines on two real-world datasets, where *m* indicates minutes.

Except for the non-sequential model of DIN, SURGE's efficiency improvement compared with all baselines is more than 20%.

- SURGE **compresses the sequence** before feeding the embedded sequence into the recurrent neural network, which greatly **reduces the number of recurrent steps**.
- Since most of the noise is filtered, the pooled sequence **only contains the core interest**, which will undoubtedly help **speed up the model's convergence**.

Experiments

❑ Design choices for interest evolution



SURGE's third advantage is that the framework can bring benefits to some existing methods.

- Modeling on the compressed sequence will significantly reduce the difficulty of capturing user interests.

Conclusion & Future Work

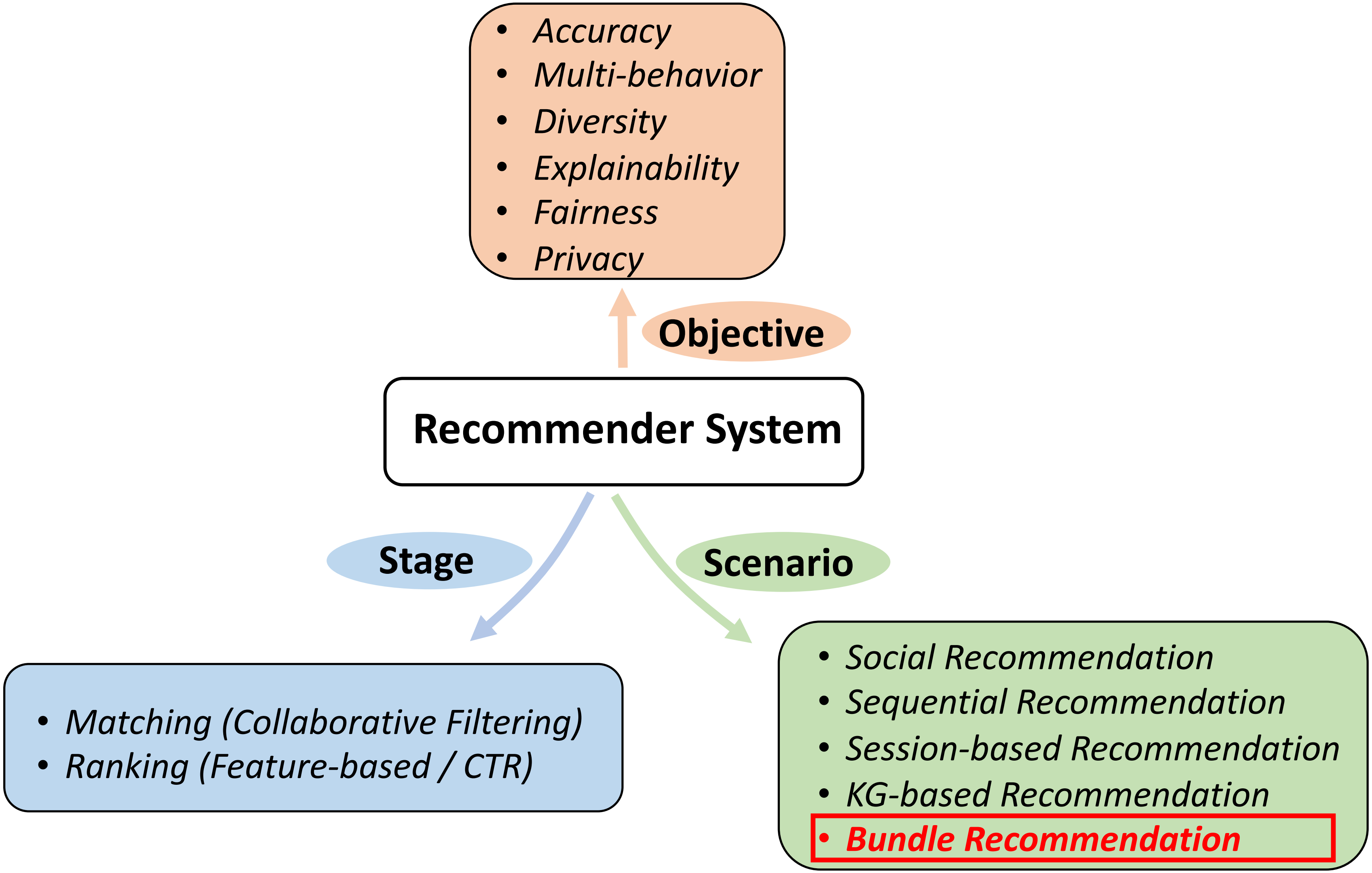
❑ Conclusion

- We approach **sequential recommendation from a new perspective**.
- We propose to **aggregate implicit signals into explicit ones** by designing graph neural network-based models on constructed item-item interest graphs.
- We design dynamic-pooling to **filter and reserve activated core preferences** for recommendation.

❑ Future Work

- We plan to use different **behaviors** to explore **fine-grained** multiple interactions **from noisy historical sequences.**

Recent advances of GNN-based RecSys



- *Accuracy*
- *Multi-behavior*
- *Diversity*
- *Explainability*
- *Fairness*
- *Privacy*

Objective

Recommender System

Stage

Scenario

- *Matching (Collaborative Filtering)*
- *Ranking (Feature-based / CTR)*

- *Social Recommendation*
- *Sequential Recommendation*
- *Session-based Recommendation*
- *KG-based Recommendation*
- ***Bundle Recommendation***

Bundle recommendation with graph convolutional networks.
Chang, J., Gao, C., He, X., Jin, D., & Li, Y. SIGIR 2020

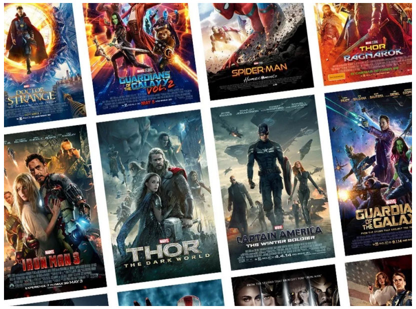
What is a bundle?

Nursery Bundle

Suit Bundle

Computer Bundle

App Bundle

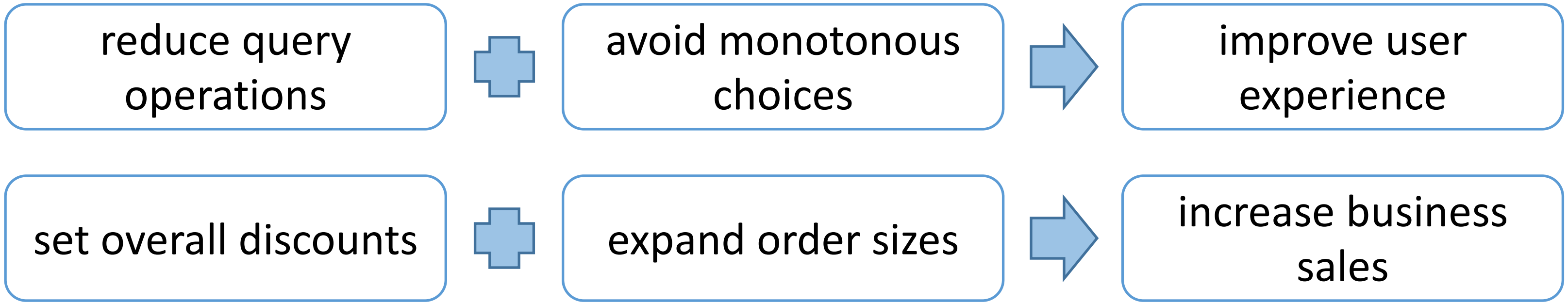


Movie Bundle

Game Bundle

Background

- Bundle recommendation aims to recommend a bundle of items for a user to consume as a whole.



reduce query operations + avoid monotonous choices → improve user experience

set overall discounts + expand order sizes → increase business sales

- The prevalence of bundled items on e-commerce and content platforms makes it become an important task.

e-commerce

content

Challenges

□ Model

- The attractiveness of a bundle depends on its items.
- The users need to be satisfied with most items in a bundle.
- The items matching degree will affect the user's choice.

Challenges

❑ Data

- On the existing platforms, the item is still the main form to buy.
- The number of bundles that the user has interacted with is limited.
- There is a sparser interaction between the user and bundle.

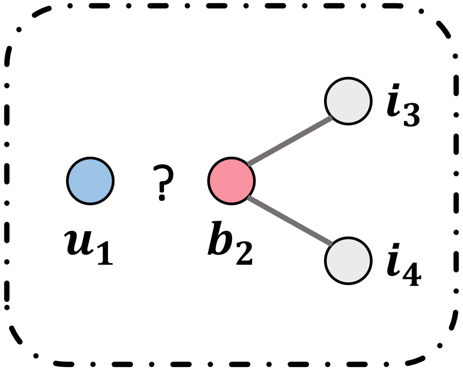
Problem Definition

❑ Input:

- user-bundle interaction records
- user-item interaction records
- bundle-item affiliation information

❑ Output:

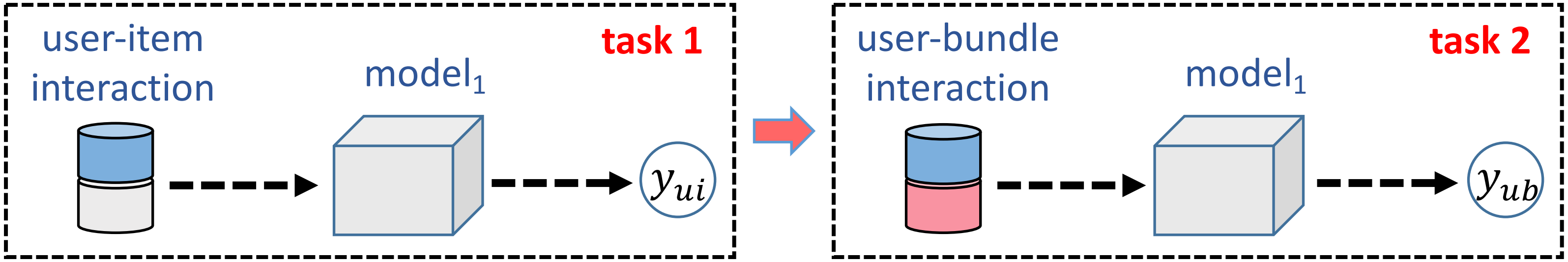
- user-bundle interaction probability



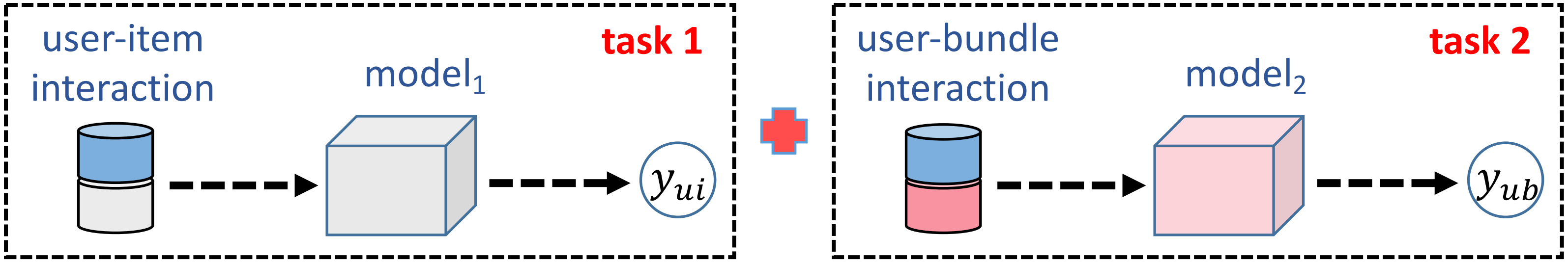
Limitations of Existing works

1. Separated modeling of two affiliated entities.

- reuse model parameters



- share model parameters



- It is difficult to balance the weights of the main task and auxiliary task.

Limitations of Existing works

2. Substitution of bundles is not considered.

- They only consider the correlation between items in a bundle to enhance the item task.

- The association between the bundles as the recommended target is even more critical.

Limitations of Existing works

3. Decision-making is ignored in bundle scenarios.

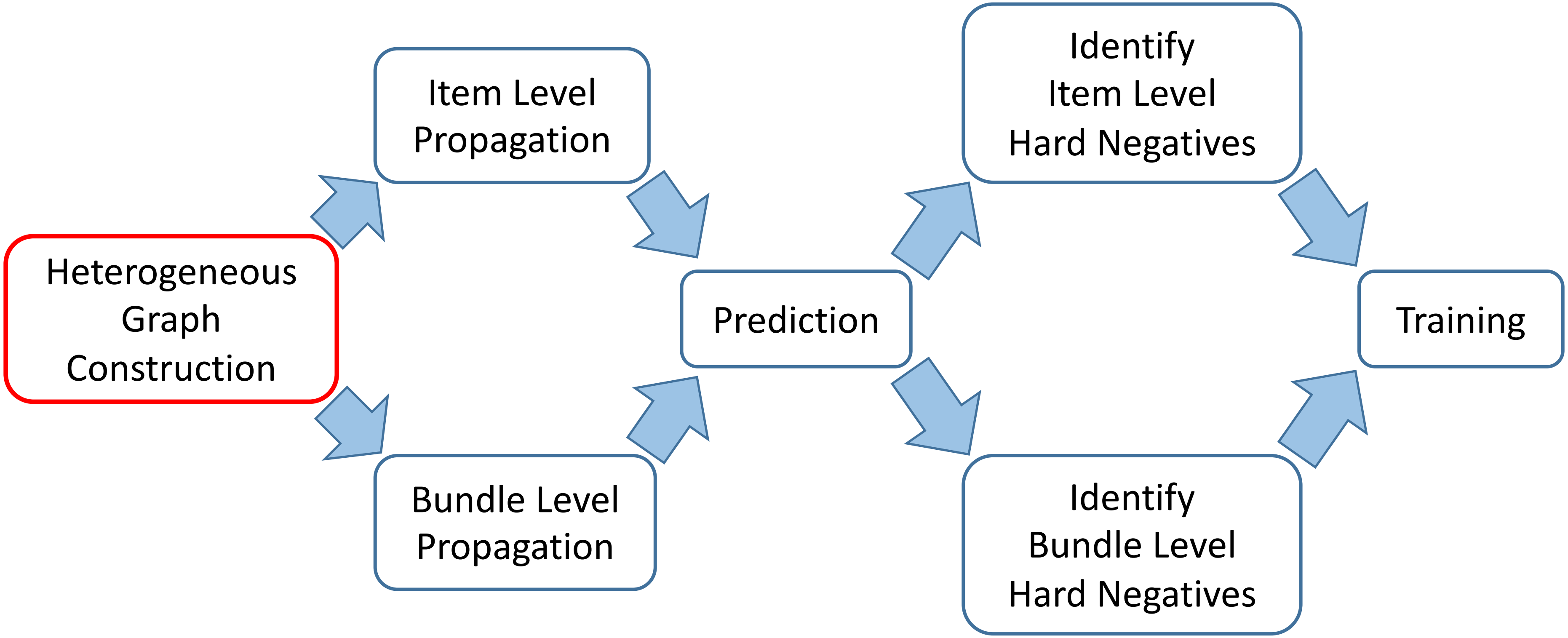
- Even though a user likes most items in a bundle, but may refuse the bundle.

- For two highly similar bundles, the key to the user’s final selection is non-overlapping parts.

BGCN Model Framework

- ❑ Heterogeneous Graph Construction

Solve Limitation 1: Separated modeling of two affiliated entities.

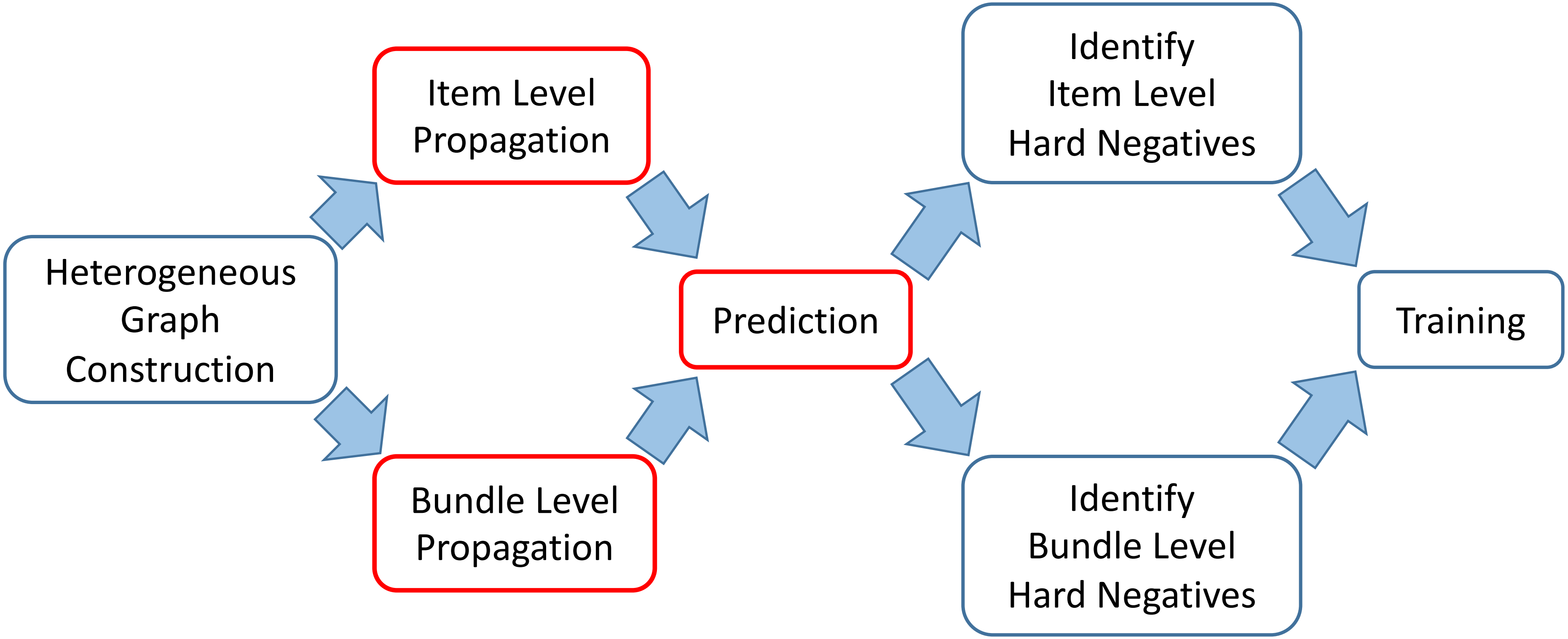

Heterogeneous Graph Construction
Item Level Propagation
Bundle Level Propagation
Prediction
Identify Item Level Hard Negatives
Identify Bundle Level Hard Negatives
Training

BGCN Model Framework

- ❑ Levels Propagation

Solve Limitation 1 and Limitation 2:

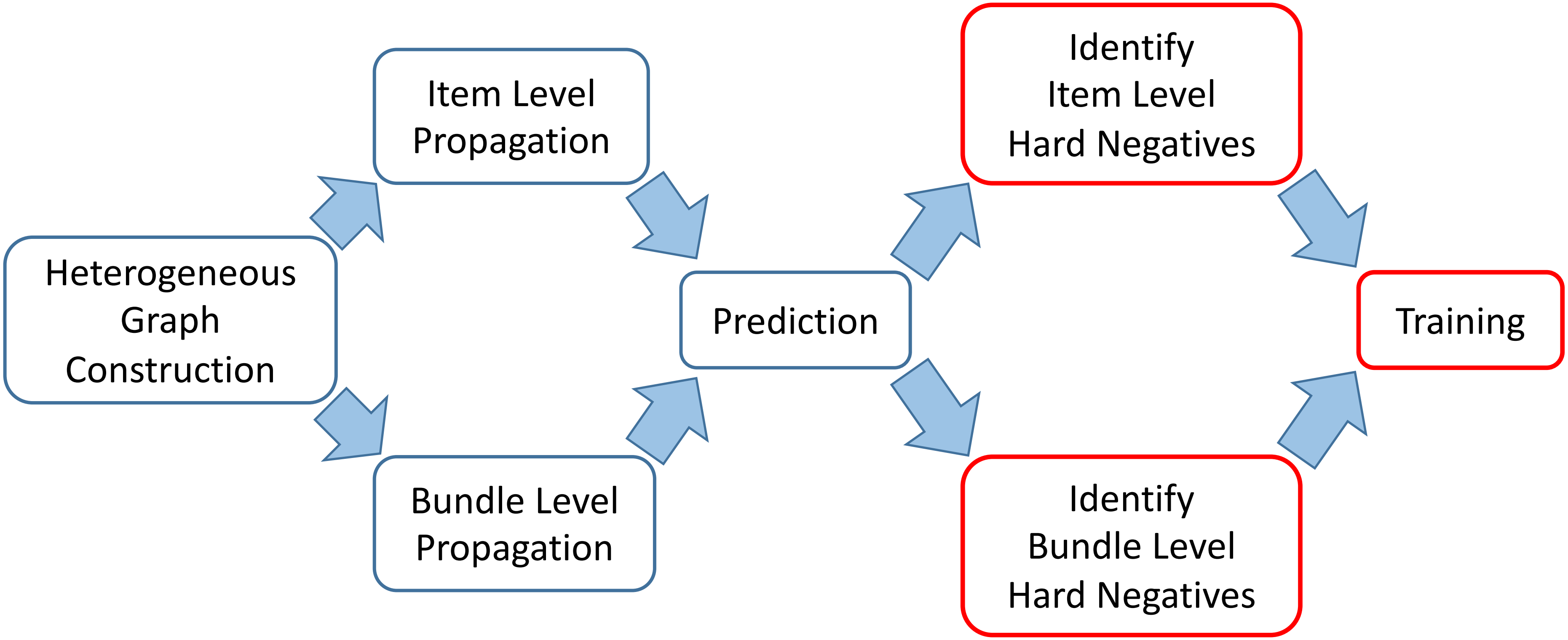
Substitution of bundles is not considered.


Item Level Propagation
Identify Item Level Hard Negatives
Heterogeneous Graph Construction
Prediction
Training
Bundle Level Propagation
Identify Bundle Level Hard Negatives

BGCN Model Framework

- ❑ Training with Hard Negatives

Solve Limitation 3: Decision-making is ignored in bundle scenarios.



Heterogeneous Graph Construction

Item Level Propagation

Bundle Level Propagation

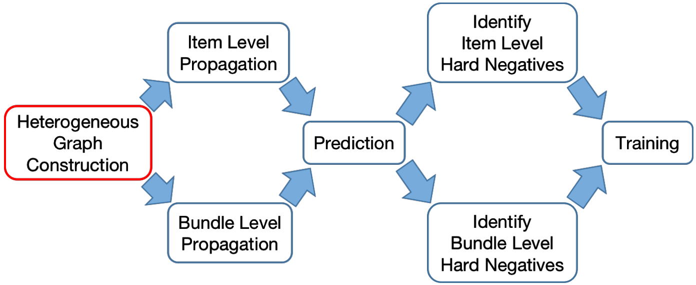
Prediction

Identify Item Level Hard Negatives

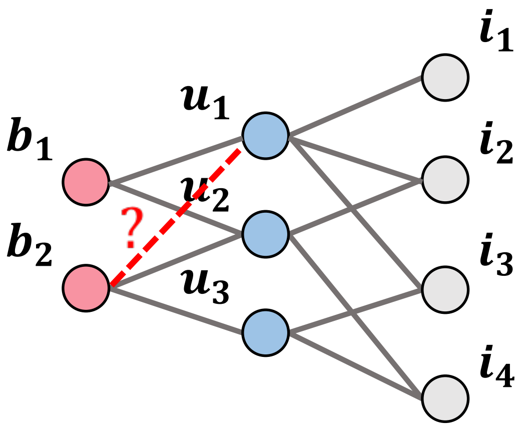
Identify Bundle Level Hard Negatives

Training

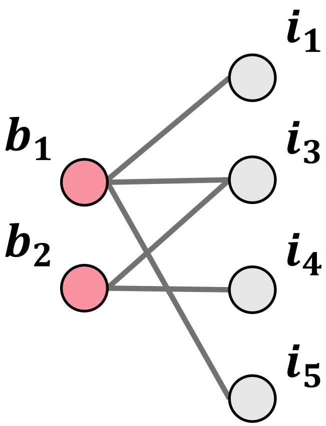
Methodology


Heterogeneous Graph Construction
Item Level Propagation
Bundle Level Propagation
Prediction
Identify Item Level Hard Negatives
Identify Bundle Level Hard Negatives
Training

❑ Heterogeneous Graph Construction



Interaction Relation
an observed link means **user *u* once purchased bundle *b* or item *i*.**



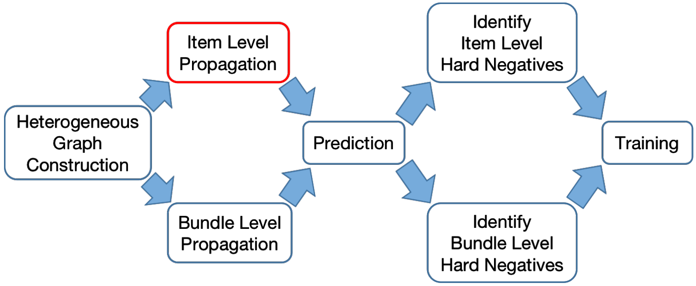
Affiliation Relation
an observed link means **bundle *b* contains item *i*.**

- Our target:

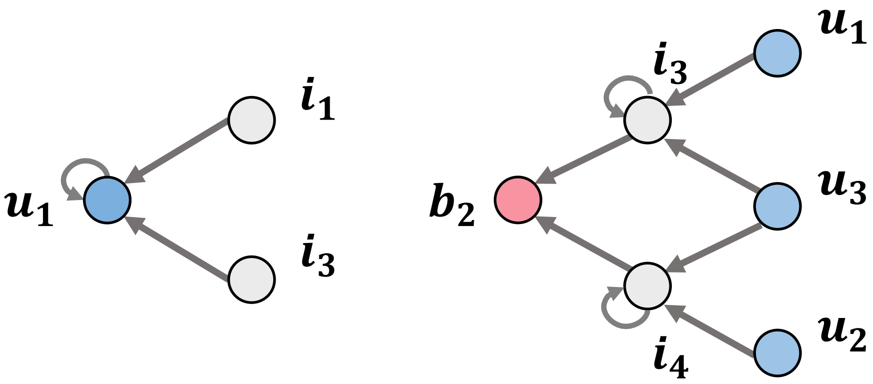
Limitation 1 is addressed!

  - predict any possible unobserved links between u and b.
  - e.g., will user 1 interact with bundle 2?

Methodology

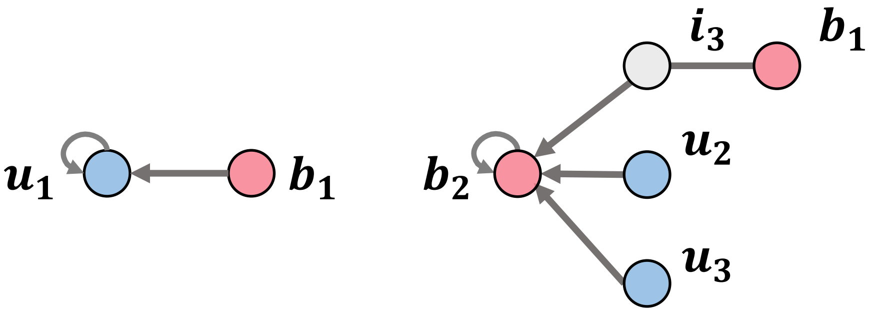

Item Level Propagation
Identify Item Level Hard Negatives
Heterogeneous Graph Construction
Prediction
Training
Bundle Level Propagation
Identify Bundle Level Hard Negatives

□ Item Level Propagation

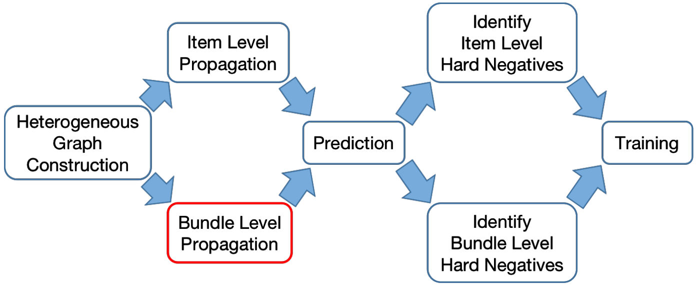


Methodology

❑ Bundle Level Propagation



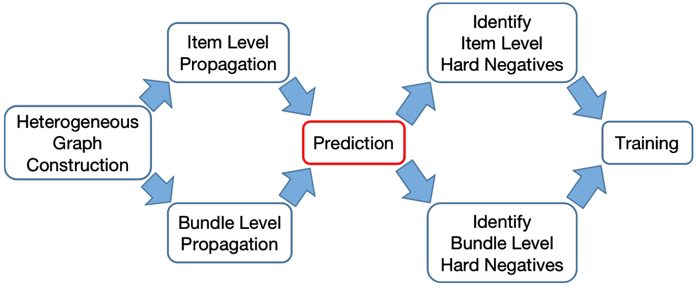
Limitation 2 is addressed!


Item Level Propagation
Identify Item Level Hard Negatives
Heterogeneous Graph Construction
Prediction
Training
Bundle Level Propagation
Identify Bundle Level Hard Negatives

v.s.

v.s.

Methodology


Heterogeneous Graph Construction

Item Level Propagation

Bundle Level Propagation

Prediction

Identify Item Level Hard Negatives

Identify Bundle Level Hard Negatives

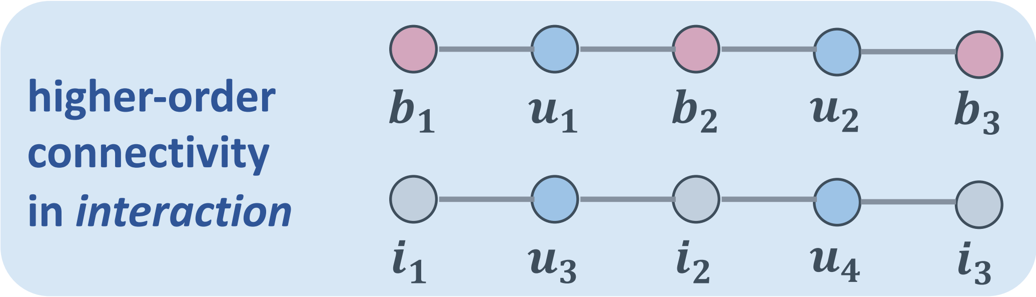
Training

❑ Prediction

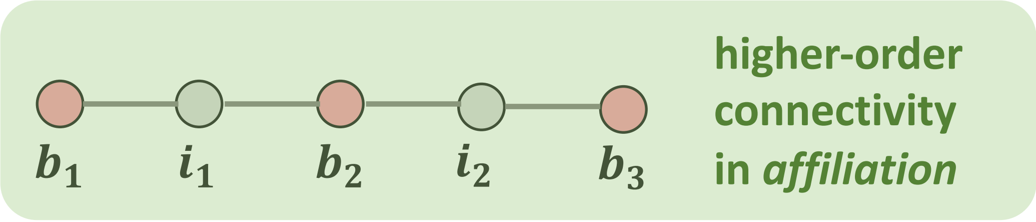
- propagate iteratively for *L* times;
- concatenate *L* layers' embeddings.

$$\mathbf{p}^*_{u,1} = \mathbf{p}^{(0)}_{u,2} || \cdots || \mathbf{p}^{(L)}_{u,1}, \quad \mathbf{r}^*_{b,1} = \mathbf{r}^{(0)}_{b,1} || \cdots || \mathbf{r}^{(L)}_{b,1},$$
$$\mathbf{p}^*_{u,2} = \mathbf{p}^{(0)}_{u,2} || \cdots || \mathbf{p}^{(L)}_{u,2}, \quad \mathbf{r}^*_{b,2} = \mathbf{r}^{(0)}_{b,2} || \cdots || \mathbf{r}^{(L)}_{b,2}.$$

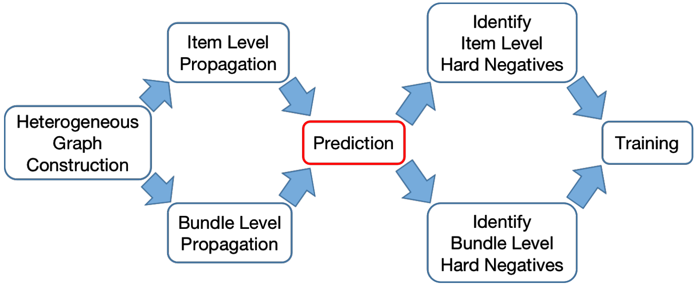
- combine the information from different depths


higher-order connectivity in *interaction*

cannot be modeled in CF model!


higher-order connectivity in *affiliation*

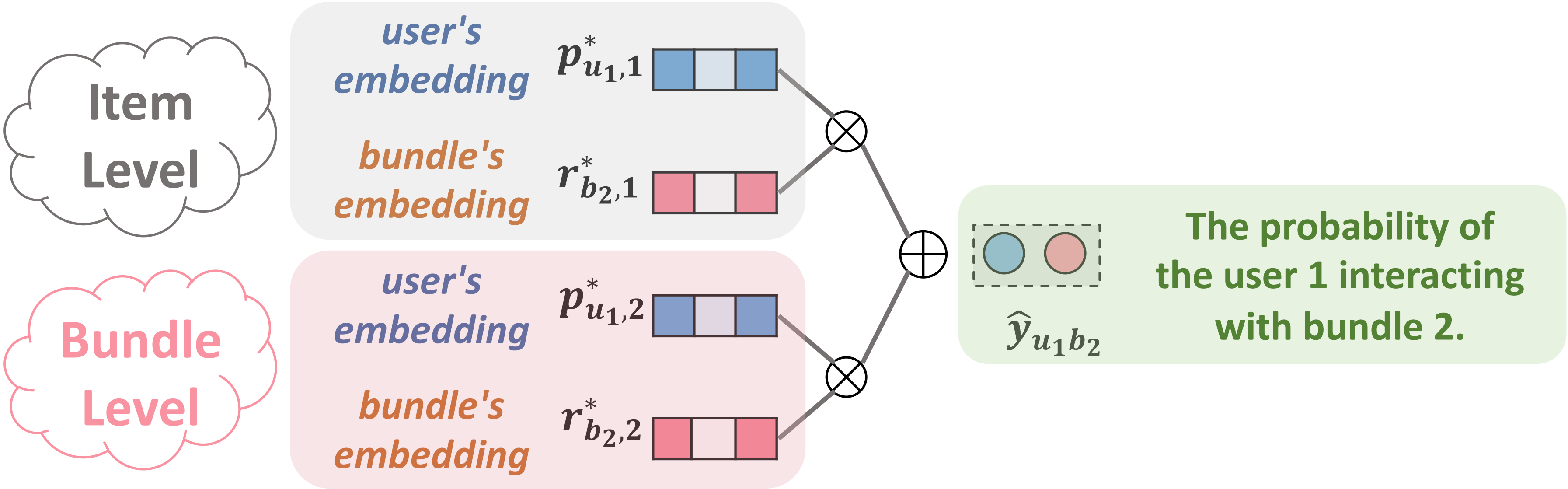
Methodology


Heterogeneous Graph Construction
Item Level Propagation
Bundle Level Propagation
Prediction
Identify Item Level Hard Negatives
Identify Bundle Level Hard Negatives
Training

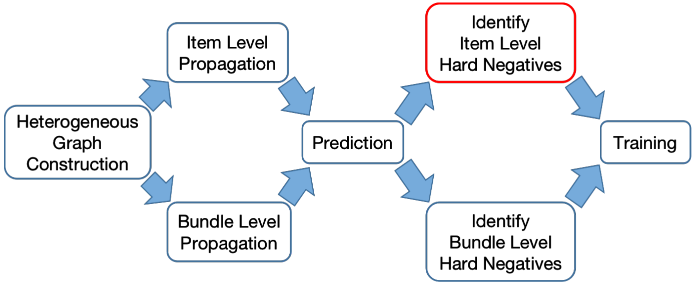
❑ Prediction

- adopt inner product
- combine bundle and item levels

$$\hat{y}_{ub} = \mathbf{p}^{*}_{u,1}{}^{\top}\mathbf{r}^{*}_{b,1} + \mathbf{p}^{*}_{u,2}{}^{\top}\mathbf{r}^{*}_{b,2}.$$



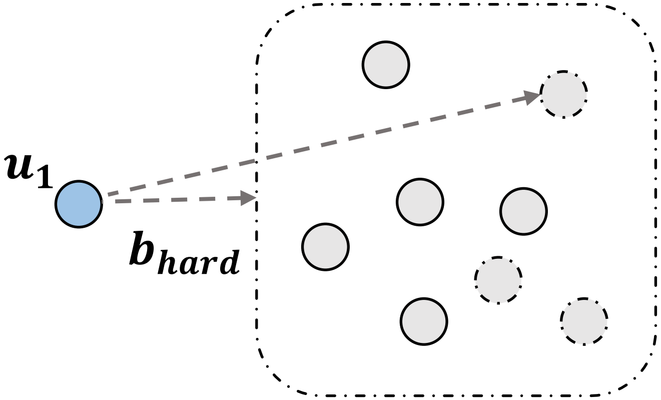
Methodology


Heterogeneous Graph Construction
Item Level Propagation
Bundle Level Propagation
Prediction
Identify Item Level Hard Negatives
Identify Bundle Level Hard Negatives
Training

❑ Training with Hard Negatives

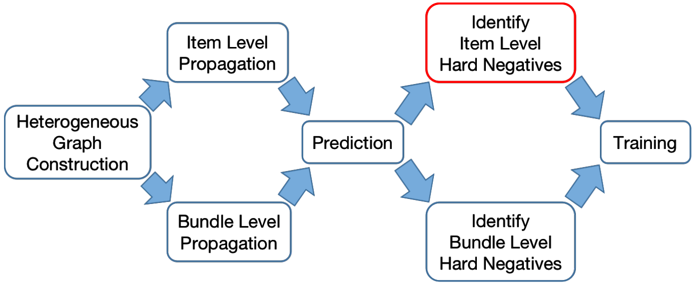
bundles contain more items and have higher prices → users are often cautious to avoid unnecessary risks

- Identify Item Level Hard Negatives

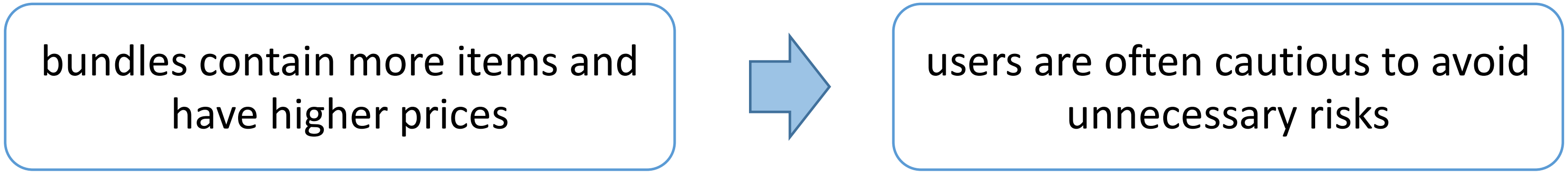


- Even though a user likes most items in a bundle,
- but may refuse it because of the existence of one disliked item.

Methodology


Heterogeneous Graph Construction
Item Level Propagation
Bundle Level Propagation
Prediction
Identify Item Level Hard Negatives
Identify Bundle Level Hard Negatives
Training

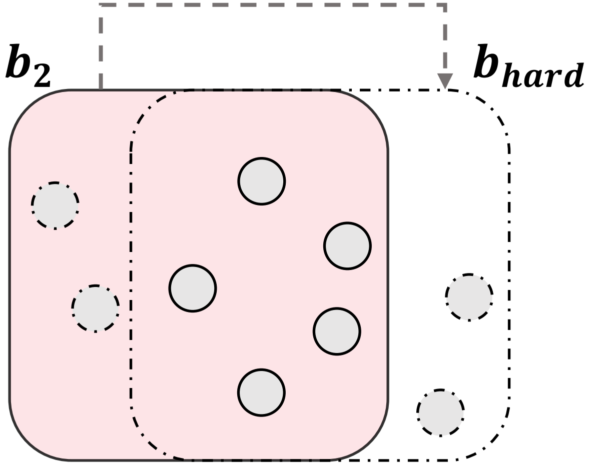
❑ Training with Hard Negatives



bundles contain more items and have higher prices

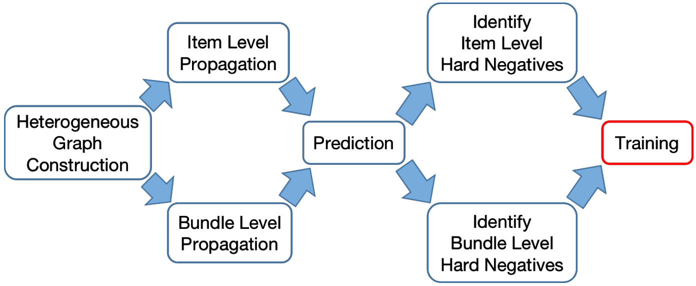
users are often cautious to avoid unnecessary risks

- ▪ Identify Bundle Level Hard Negatives



- For two highly similar bundles,
- the key to the user’s final selection is their non-overlapping parts.

Methodology



Heterogeneous Graph Construction

Item Level Propagation

Bundle Level Propagation

Prediction

Identify Item Level Hard Negatives

Identify Bundle Level Hard Negatives

Training

❑ Training with Hard Negatives

bundles contain more items and have higher prices

→

users are often cautious to avoid unnecessary risks.

- Training

Limitation 3 is addressed!

$$\text{Loss} = \sum_{(u,b,c)\in Q} -\ln\sigma(\hat{y}_{ub} - \hat{y}_{uc}) + \beta \cdot \|\Theta\|^2,$$

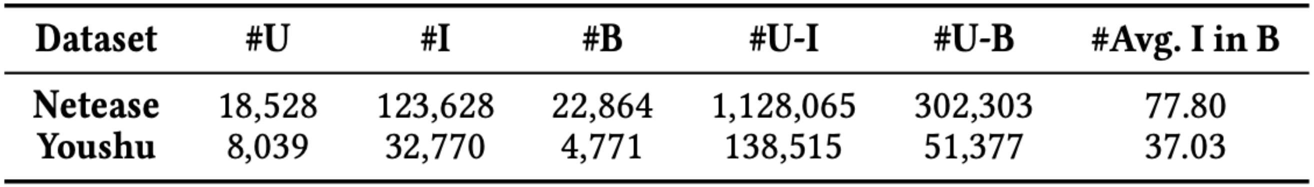
$$Q = \{(u,b,c) | (u,b) \in \mathcal{Y}^+, (u,c) \in \mathcal{Y}^-\}$$

- Bayesian Personalized Ranking pairwise learning.
- To prevent over-fitting, we adopt L2 regularization.
- After the model converges, the hard-negative samples are selected with a certain probability(80%) for training.

Experiments

❑ Datasets

- Two real-world datasets



Dataset	#U	#I	#B	#U-I	#U-B	#Avg. I in B
Netease	18,528	123,628	22,864	1,128,065	302,303	77.80
Youshu	8,039	32,770	4,771	138,515	51,377	37.03

❑ Top-K Evaluation Metrics:

- Recall@K and NDCG@K

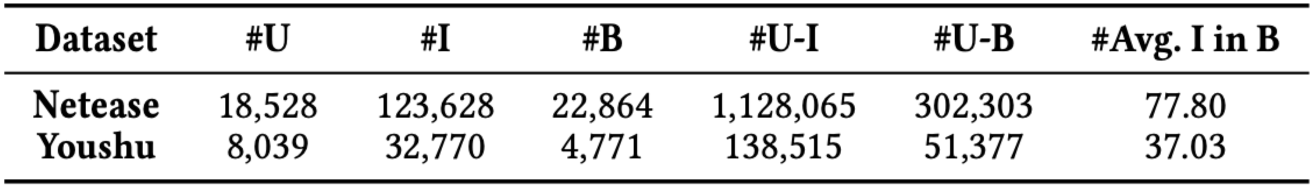
❑ Baseline

Model	Bundle Level	Item Level	Graph	Propagation
MFBPR	√			
GCN-BG	√		√	
GCN-TG	√	√	√	
NGCF-BG	√		√	
NGCF-TG	√	√	√	
DAM	√	√		

Experiments

❑ Datasets

- Two real-world datasets

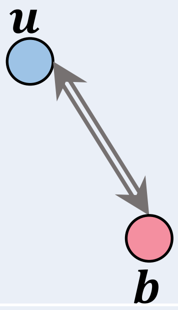


Dataset	#U	#I	#B	#U-I	#U-B	#Avg. I in B
Netease	18,528	123,628	22,864	1,128,065	302,303	77.80
Youshu	8,039	32,770	4,771	138,515	51,377	37.03

❑ Top-K Evaluation Metrics:

- Recall@K and NDCG@K

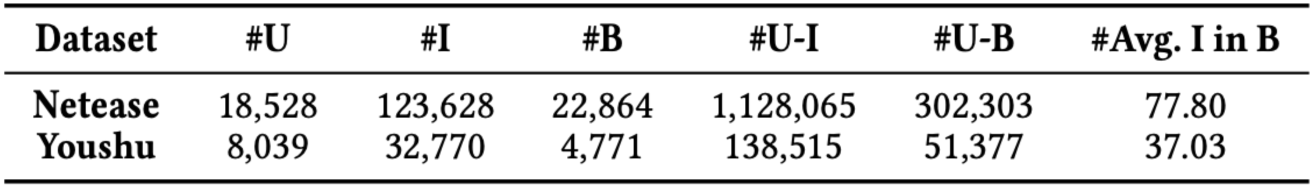
❑ Baseline

Model	Bundle Level	Item Level	Graph	Propagation
MFBPR	√			
GCN-BG	√		√	
GCN-TG	√	√	√	
NGCF-BG	√		√	
NGCF-TG	√	√	√	
DAM	√	√		

Experiments

❑ Datasets

- Two real-world datasets

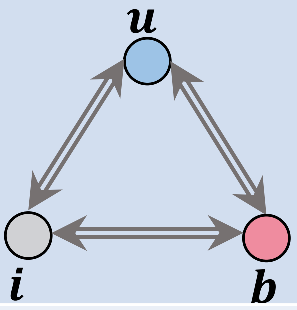


Dataset	#U	#I	#B	#U-I	#U-B	#Avg. I in B
Netease	18,528	123,628	22,864	1,128,065	302,303	77.80
Youshu	8,039	32,770	4,771	138,515	51,377	37.03

❑ Top-K Evaluation Metrics:

- Recall@K and NDCG@K

❑ Baseline

Model	Bundle Level	Item Level	Graph	Propagation
MFBPR	√			
GCN-BG	√		√	
GCN-TG	√	√	√	
NGCF-BG	√		√	
NGCF-TG	√	√	√	
DAM	√	√		

Experiments

❑ Overal Performance

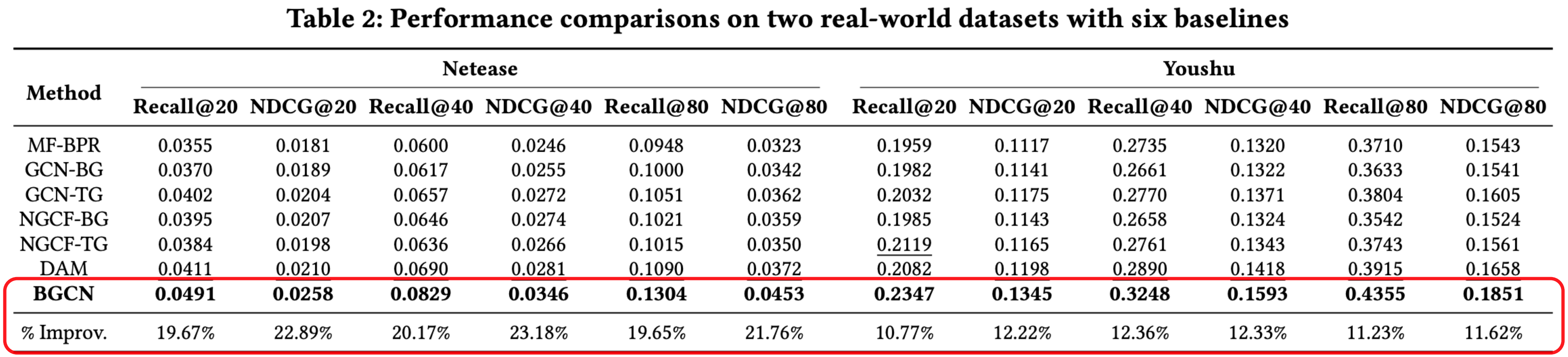


Table 2: Performance comparisons on two real-world datasets with six baselines

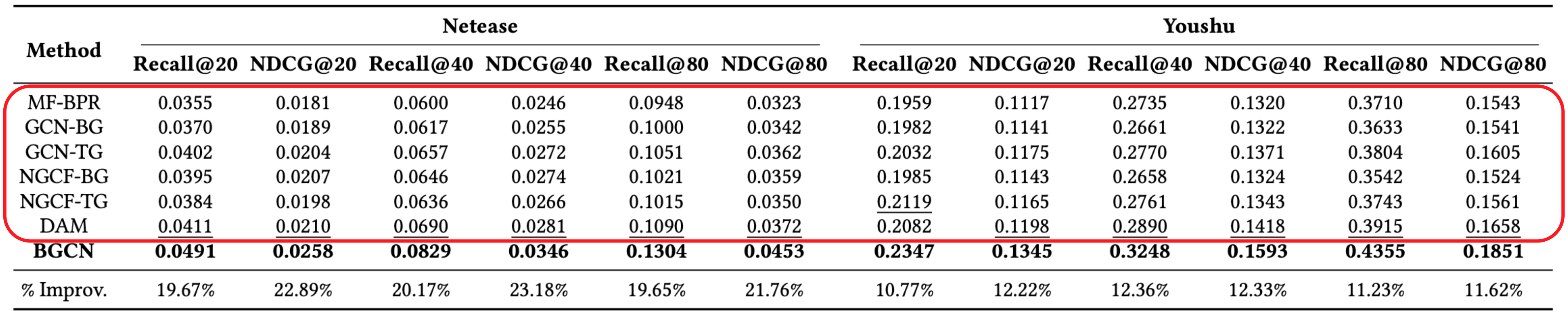
Method	Netease						Youshu					
	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80
MF-BPR	0.0355	0.0181	0.0600	0.0246	0.0948	0.0323	0.1959	0.1117	0.2735	0.1320	0.3710	0.1543
GCN-BG	0.0370	0.0189	0.0617	0.0255	0.1000	0.0342	0.1982	0.1141	0.2661	0.1322	0.3633	0.1541
GCN-TG	0.0402	0.0204	0.0657	0.0272	0.1051	0.0362	0.2032	0.1175	0.2770	0.1371	0.3804	0.1605
NGCF-BG	0.0395	0.0207	0.0646	0.0274	0.1021	0.0359	0.1985	0.1143	0.2658	0.1324	0.3542	0.1524
NGCF-TG	0.0384	0.0198	0.0636	0.0266	0.1015	0.0350	0.2119	0.1165	0.2761	0.1343	0.3743	0.1561
DAM	0.0411	0.0210	0.0690	0.0281	0.1090	0.0372	0.2082	0.1198	0.2890	0.1418	0.3915	0.1658
BGCN	**0.0491**	**0.0258**	**0.0829**	**0.0346**	**0.1304**	**0.0453**	**0.2347**	**0.1345**	**0.3248**	**0.1593**	**0.4355**	**0.1851**
% Improv.	19.67%	22.89%	20.17%	23.18%	19.65%	21.76%	10.77%	12.22%	12.36%	12.33%	11.23%	11.62%

1. **Our proposed BGCN achieves the best performance.**

Experiments

❑ Overal Performance

Table 2: Performance comparisons on two real-world datasets with six baselines



Method	Netease						Youshu					
	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80
MF-BPR	0.0355	0.0181	0.0600	0.0246	0.0948	0.0323	0.1959	0.1117	0.2735	0.1320	0.3710	0.1543
GCN-BG	0.0370	0.0189	0.0617	0.0255	0.1000	0.0342	0.1982	0.1141	0.2661	0.1322	0.3633	0.1541
GCN-TG	0.0402	0.0204	0.0657	0.0272	0.1051	0.0362	0.2032	0.1175	0.2770	0.1371	0.3804	0.1605
NGCF-BG	0.0395	0.0207	0.0646	0.0274	0.1021	0.0359	0.1985	0.1143	0.2658	0.1324	0.3542	0.1524
NGCF-TG	0.0384	0.0198	0.0636	0.0266	0.1015	0.0350	0.2119	0.1165	0.2761	0.1343	0.3743	0.1561
DAM	0.0411	0.0210	0.0690	0.0281	0.1090	0.0372	0.2082	0.1198	0.2890	0.1418	0.3915	0.1658
BGCN	**0.0491**	**0.0258**	**0.0829**	**0.0346**	**0.1304**	**0.0453**	**0.2347**	**0.1345**	**0.3248**	**0.1593**	**0.4355**	**0.1851**
% Improv.	19.67%	22.89%	20.17%	23.18%	19.65%	21.76%	10.77%	12.22%	12.36%	12.33%	11.23%	11.62%

1. **Our proposed BGCN achieves the best performance.**
2. **Graph models have advantages but not enough.**

Experiments

❑ Overal Performance

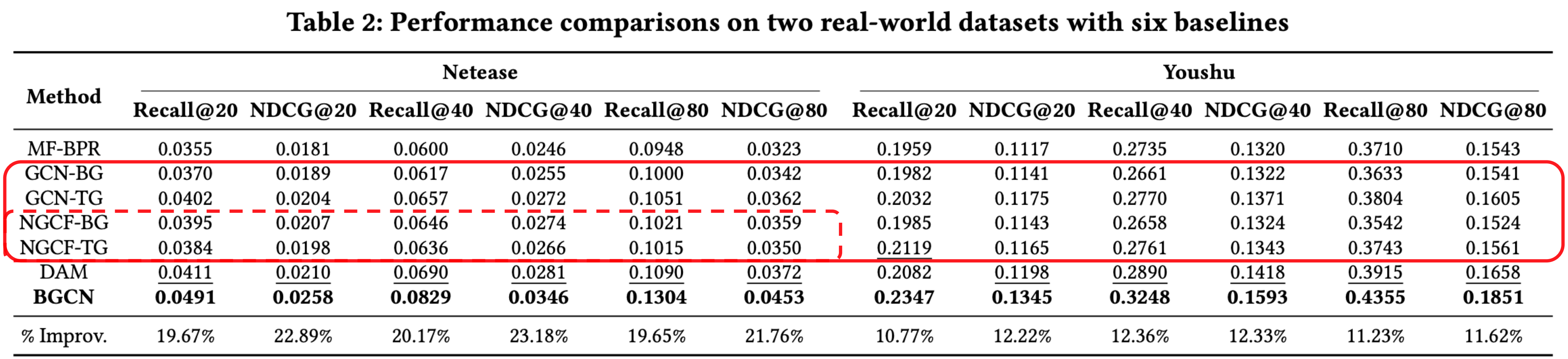


Table 2: Performance comparisons on two real-world datasets with six baselines

Method	Netease						Youshu					
	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80
MF-BPR	0.0355	0.0181	0.0600	0.0246	0.0948	0.0323	0.1959	0.1117	0.2735	0.1320	0.3710	0.1543
GCN-BG	0.0370	0.0189	0.0617	0.0255	0.1000	0.0342	0.1982	0.1141	0.2661	0.1322	0.3633	0.1541
GCN-TG	0.0402	0.0204	0.0657	0.0272	0.1051	0.0362	0.2032	0.1175	0.2770	0.1371	0.3804	0.1605
NGCF-BG	0.0395	0.0207	0.0646	0.0274	0.1021	0.0359	0.1985	0.1143	0.2658	0.1324	0.3542	0.1524
NGCF-TG	0.0384	0.0198	0.0636	0.0266	0.1015	0.0350	0.2119	0.1165	0.2761	0.1343	0.3743	0.1561
DAM	0.0411	0.0210	0.0690	0.0281	0.1090	0.0372	0.2082	0.1198	0.2890	0.1418	0.3915	0.1658
BGCN	**0.0491**	**0.0258**	**0.0829**	**0.0346**	**0.1304**	**0.0453**	**0.2347**	**0.1345**	**0.3248**	**0.1593**	**0.4355**	**0.1851**
% Improv.	19.67%	22.89%	20.17%	23.18%	19.65%	21.76%	10.77%	12.22%	12.36%	12.33%	11.23%	11.62%

1. **Our proposed BGCN achieves the best performance.**
2. **Graph models have advantages but not enough.**
3. **More input does not always mean better performance.**

Experiments

❑ Overal Performance

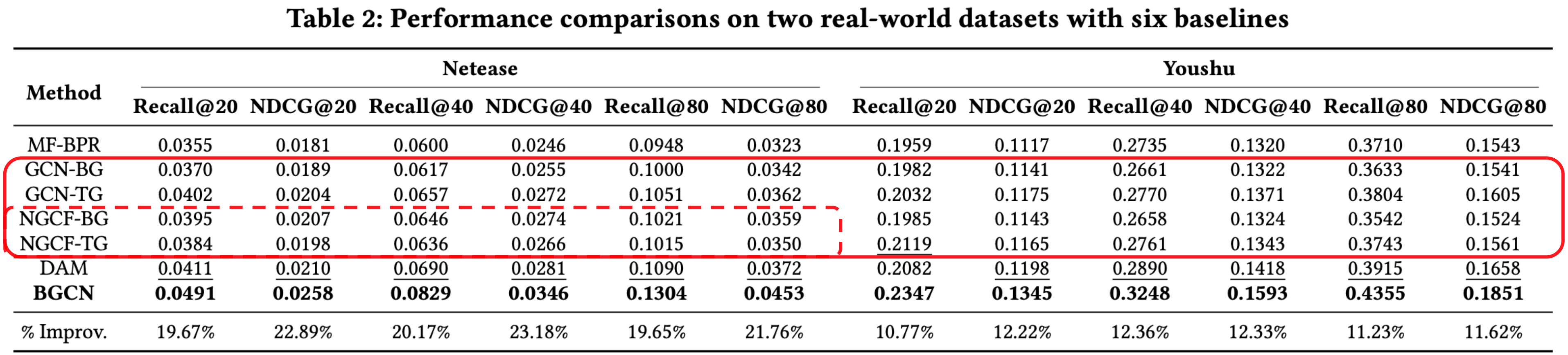
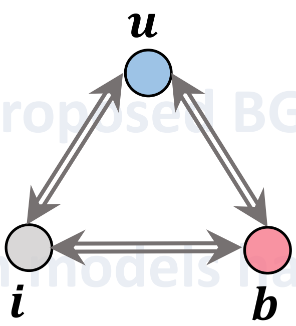


Table 2: Performance comparisons on two real-world datasets with six baselines

Method	Netease						Youshu					
	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80
MF-BPR	0.0355	0.0181	0.0600	0.0246	0.0948	0.0323	0.1959	0.1117	0.2735	0.1320	0.3710	0.1543
GCN-BG	0.0370	0.0189	0.0617	0.0255	0.1000	0.0342	0.1982	0.1141	0.2661	0.1322	0.3633	0.1541
GCN-TG	0.0402	0.0204	0.0657	0.0272	0.1051	0.0362	0.2032	0.1175	0.2770	0.1371	0.3804	0.1605
NGCF-BG	0.0395	0.0207	0.0646	0.0274	0.1021	0.0359	0.1985	0.1143	0.2658	0.1324	0.3542	0.1524
NGCF-TG	0.0384	0.0198	0.0636	0.0266	0.1015	0.0350	0.2119	0.1165	0.2761	0.1343	0.3743	0.1561
DAM	0.0411	0.0210	0.0690	0.0281	0.1090	0.0372	0.2082	0.1198	0.2890	0.1418	0.3915	0.1658
BGCN	**0.0491**	**0.0258**	**0.0829**	**0.0346**	**0.1304**	**0.0453**	**0.2347**	**0.1345**	**0.3248**	**0.1593**	**0.4355**	**0.1851**
% Improv.	19.67%	22.89%	20.17%	23.18%	19.65%	21.76%	10.77%	12.22%	12.36%	12.33%	11.23%	11.62%



1. Our proposed BGCN achieves the best performance.

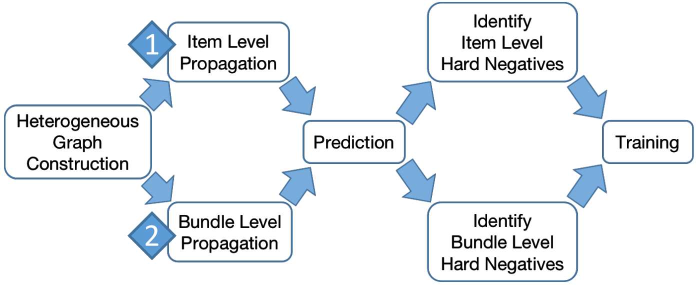
- *interaction or affiliation?*
- ***i** belongs to **b** ?*
- ***b** belongs to **i** ?*

2. Graph models have advantages but not enough.

Our special designs to make graph neural network work in bundle task is necessary.

3. **More input does not always mean better performance.**

Experiments


Heterogeneous Graph Construction
1
Item Level Propagation
2
Bundle Level Propagation
Prediction
Identify Item Level Hard Negatives
Identify Bundle Level Hard Negatives
Training

❑ Ablation Study

- Levels Propagation

1) perform propagation at only the item level;
2) perform propagation at only the bundle level;
3) perform propagation at both levels.

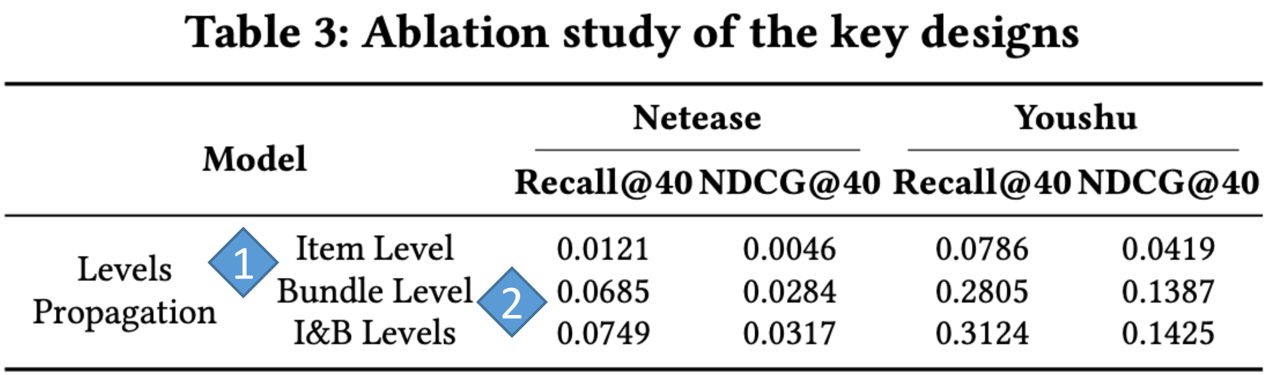
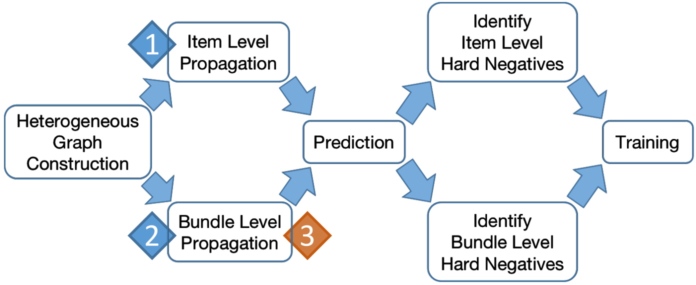


Table 3: Ablation study of the key designs

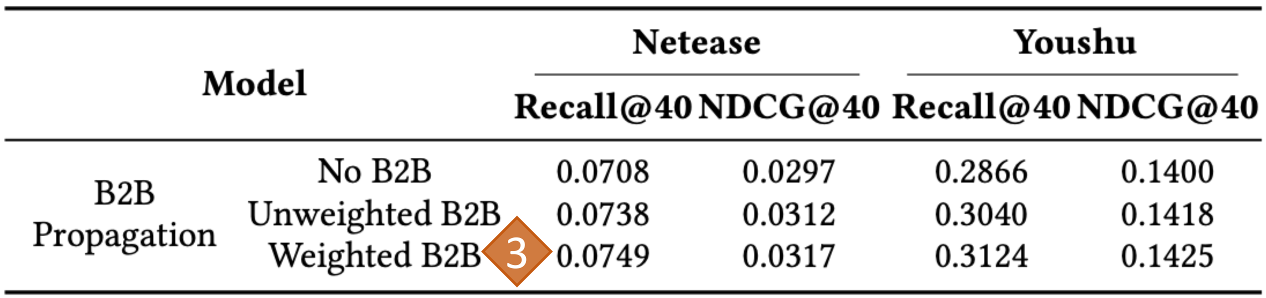
Model		Netease		Youshu	
		Recall@40	NDCG@40	Recall@40	NDCG@40
Levels Propagation	1 Item Level	0.0121	0.0046	0.0786	0.0419
	Bundle Level 2	0.0685	0.0284	0.2805	0.1387
	I&B Levels	0.0749	0.0317	0.3124	0.1425

Experiments


Heterogeneous Graph Construction
1
Item Level Propagation
2
Bundle Level Propagation
3
Prediction
Identify Item Level Hard Negatives
Identify Bundle Level Hard Negatives
Training

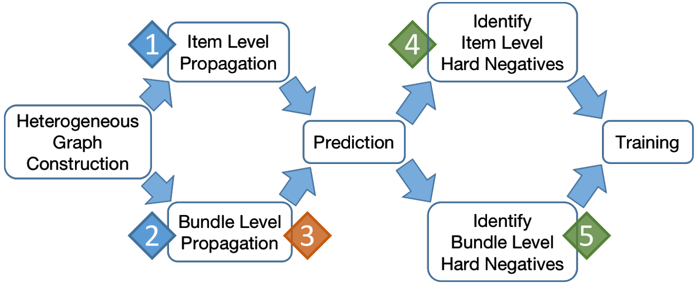
- ❑ Ablation Study
  - Levels Propagation
  - B2B Propagation
    1) bundle level propagation without *b2b*;
    2) bundle level propagation with unweighted *b2b*;
    3) bundle level propagation with weighted *b2b*.

Table 3: Ablation study of the key designs



Model		Netease		Youshu	
		Recall@40	**NDCG@40**	**Recall@40**	**NDCG@40**
B2B Propagation	No B2B	0.0708	0.0297	0.2866	0.1400
	Unweighted B2B	0.0738	0.0312	0.3040	0.1418
	Weighted B2B	0.0749	0.0317	0.3124	0.1425

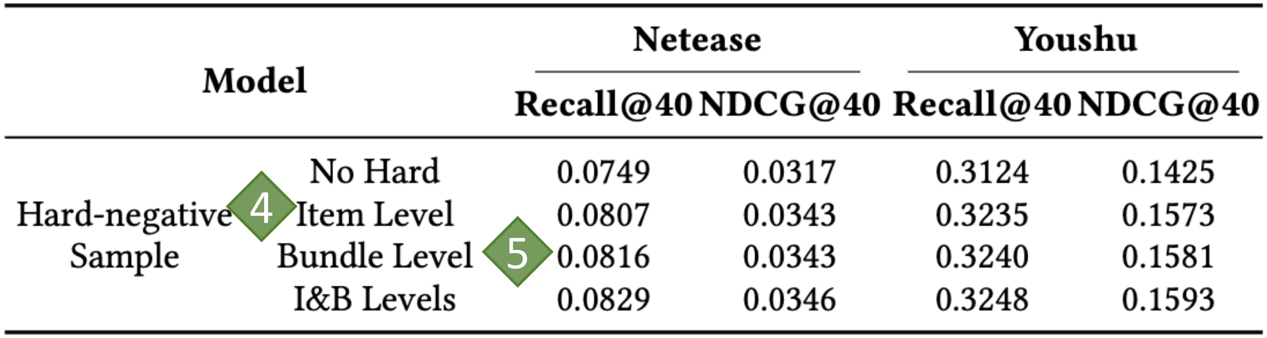
Experiments


1
Item Level Propagation
4
Identify Item Level Hard Negatives
Heterogeneous Graph Construction
Prediction
Training
2
Bundle Level Propagation
3
Identify Bundle Level Hard Negatives
5

❑ Ablation Study

- Levels Propagation
- B2B Propagation
- Hard-negative sample
  1) train without hard samples;
  2) train with hard samples at the item level;
  3) train with hard samples at the bundle level;
  4) train with hard samples at both levels.

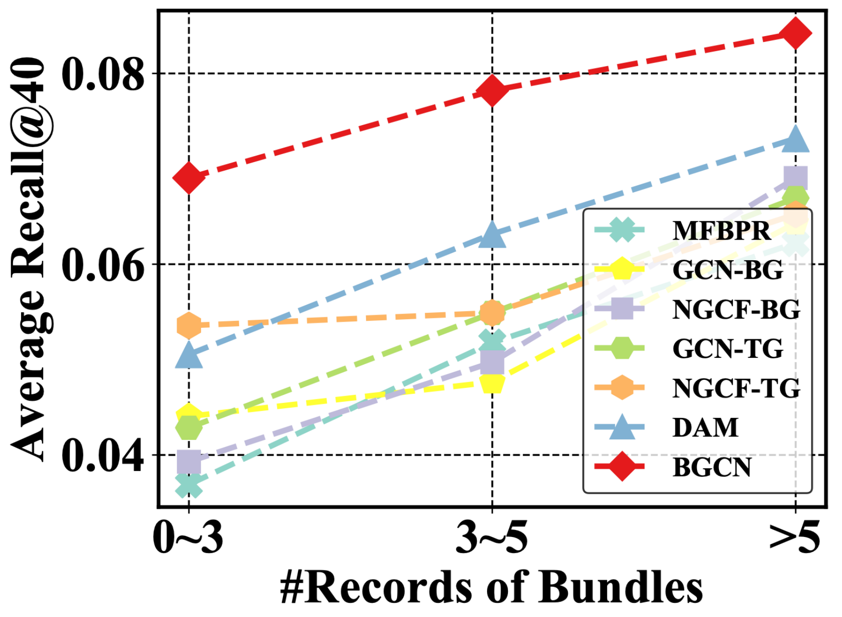
Table 3: Ablation study of the key designs

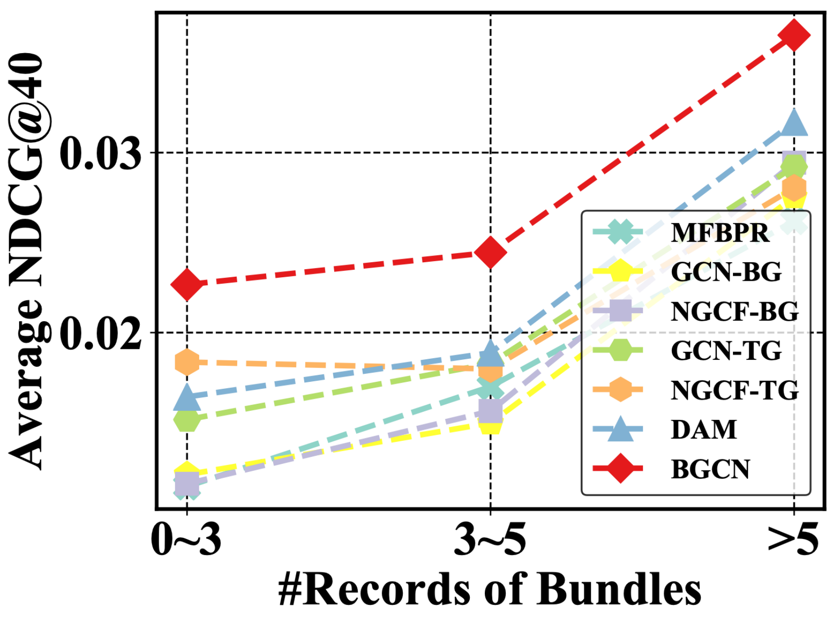


Model		Netease		Youshu	
		Recall@40	NDCG@40	Recall@40	NDCG@40
Hard-negative Sample	No Hard	0.0749	0.0317	0.3124	0.1425
	Item Level	0.0807	0.0343	0.3235	0.1573
	Bundle Level	0.0816	0.0343	0.3240	0.1581
	I&B Levels	0.0829	0.0346	0.3248	0.1593

Experiments

- ❑ Impact of Data Sparsity





Steady performance improvement achieved by BGCN

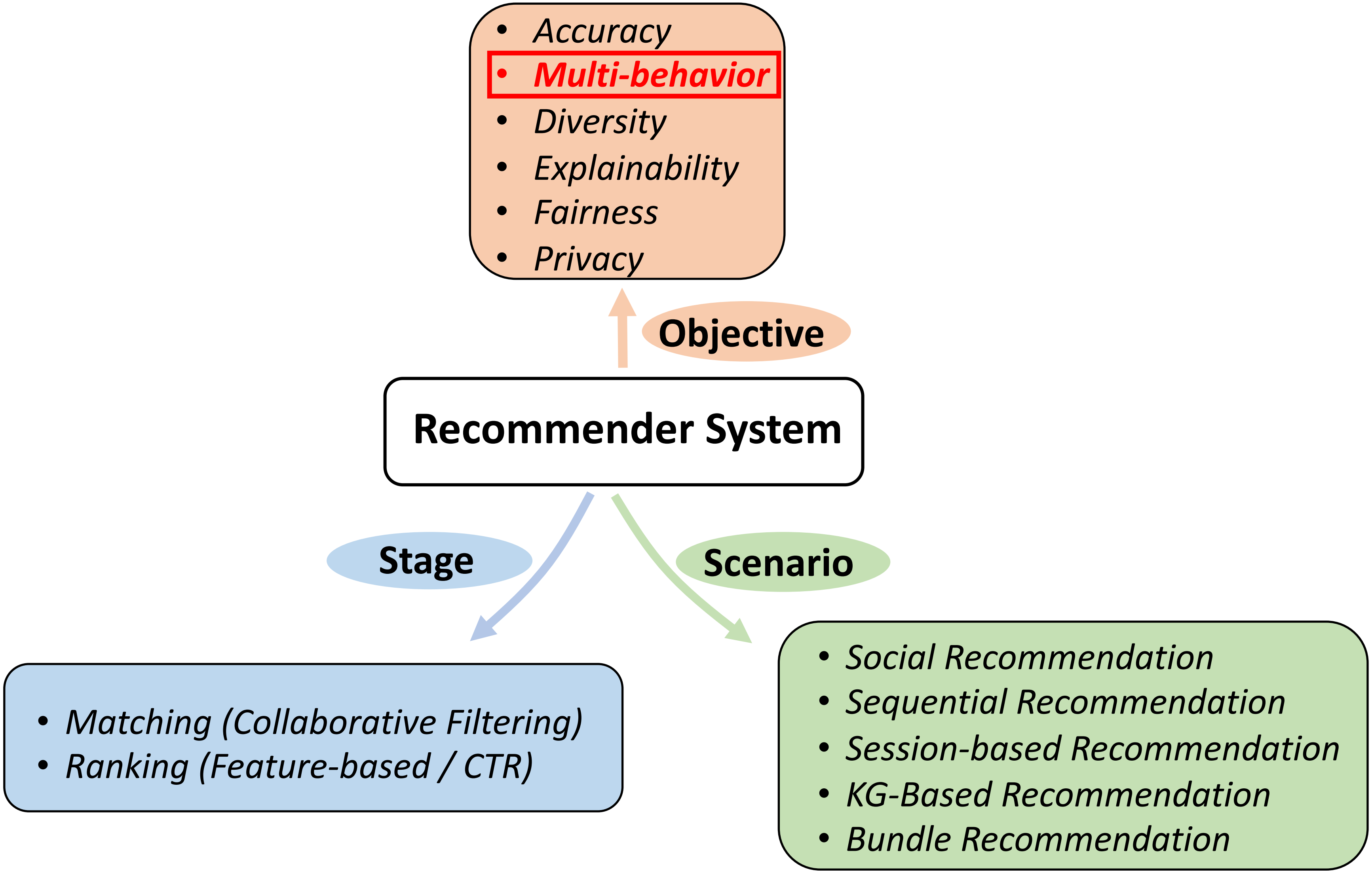
Conclusion

- We propose a **graph-based solution for bundle recommendation** which re-constructs the two kinds of interaction and an affiliation into the graph.

- With item nodes as the bridge, graph convolutional propagation between user and bundle nodes **makes the learned representations capture the item level semantics**.

- We do experiments on two real-world datasets to **demonstrate the superiority** of our model.

Recent advances of GNN-based RecSys



- *Accuracy*
- ***Multi-behavior***
- *Diversity*
- *Explainability*
- *Fairness*
- *Privacy*

Objective

Recommender System

Stage

- *Matching (Collaborative Filtering)*
- *Ranking (Feature-based / CTR)*

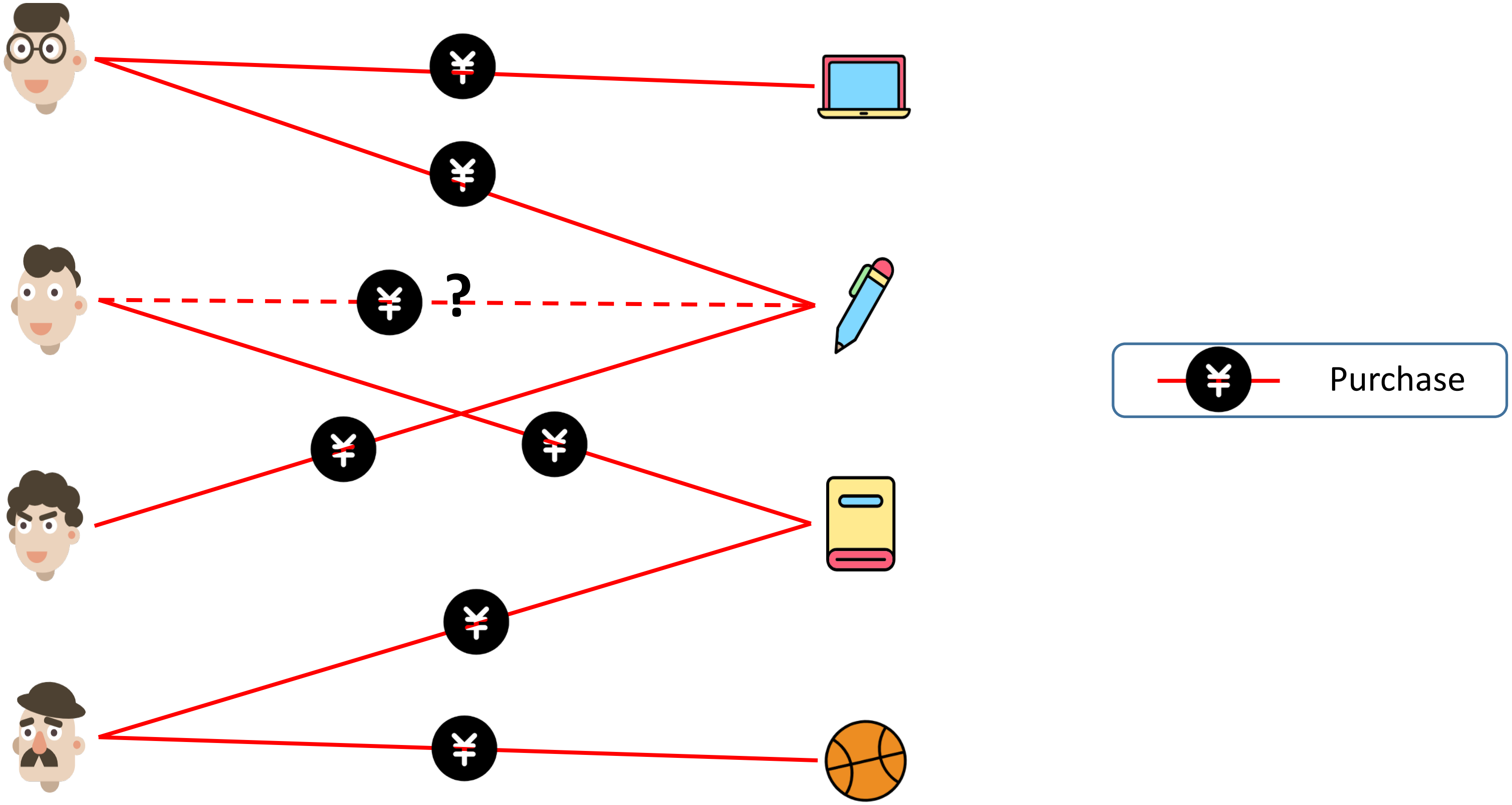
Scenario

- *Social Recommendation*
- *Sequential Recommendation*
- *Session-based Recommendation*
- *KG-Based Recommendation*
- *Bundle Recommendation*

Multi-behavior recommendation with graph convolutional networks.
Jin, B., Gao, C., He, X., Jin, D., & Li, Y. SIGIR 2020

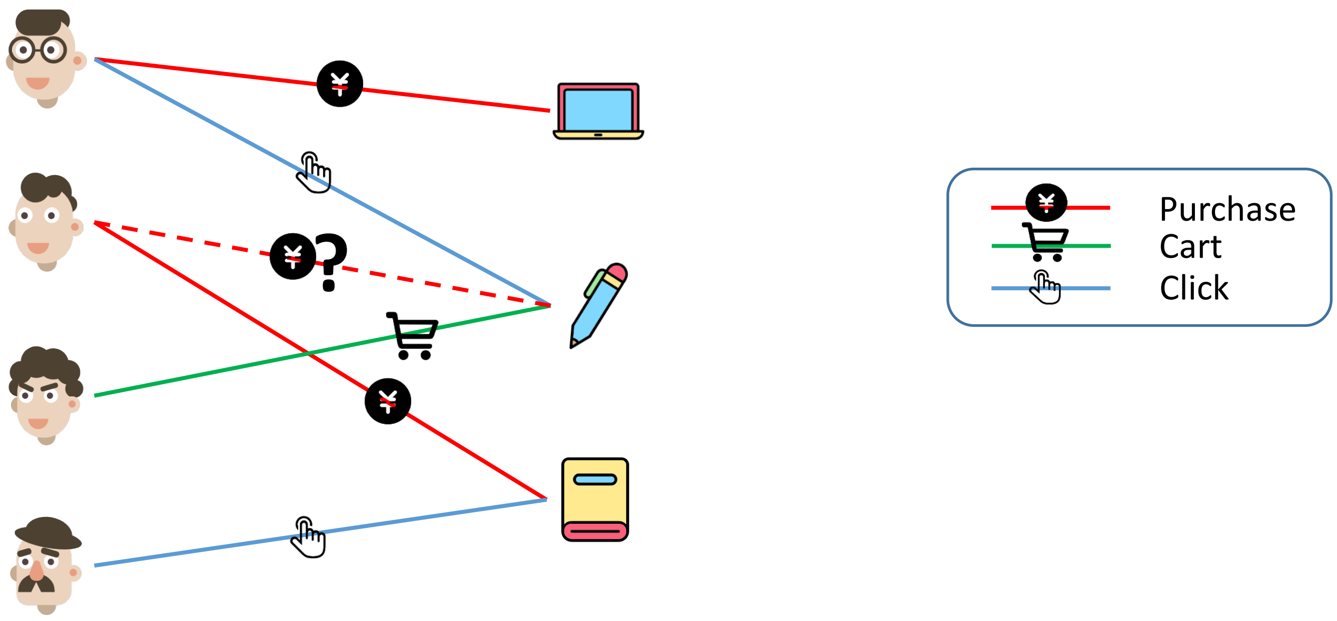
Background

- Traditional recommender system aims to give recommendation for one target behavior



Background

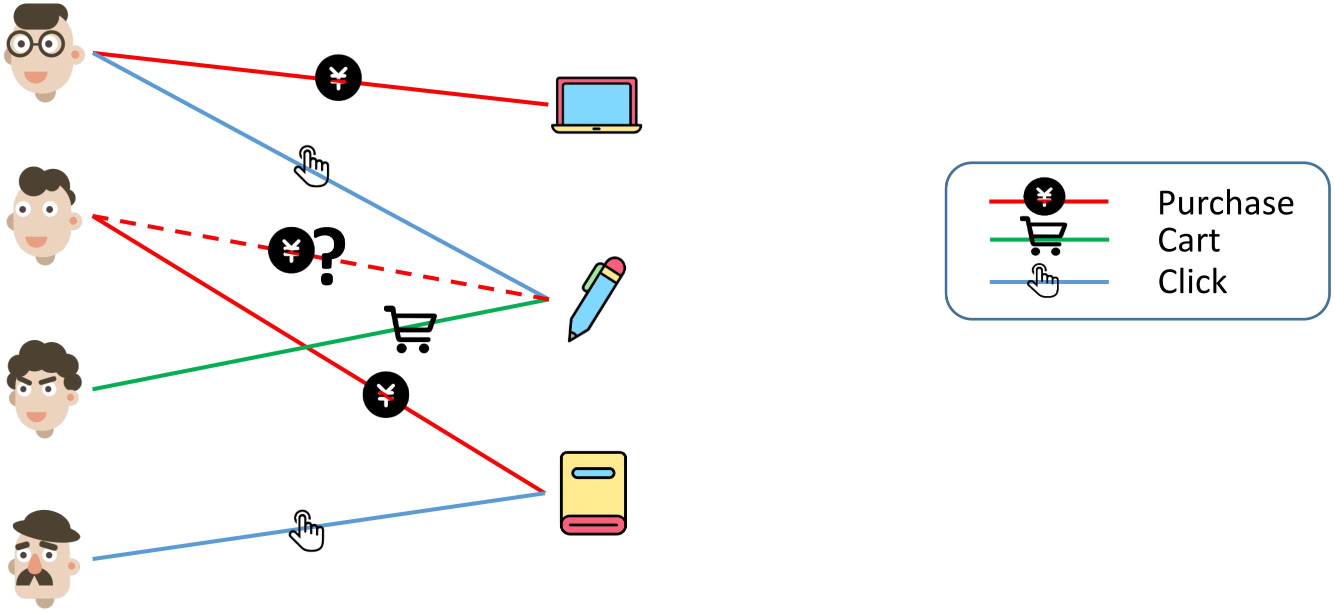
- Platform can collect users’ multi-behavior data



- Recommender systems only utilizing target behavior record suffers from **data sparsity** and **cold-start issue**
  - **The auxiliary multi-behavior data can help alleviate the issue**

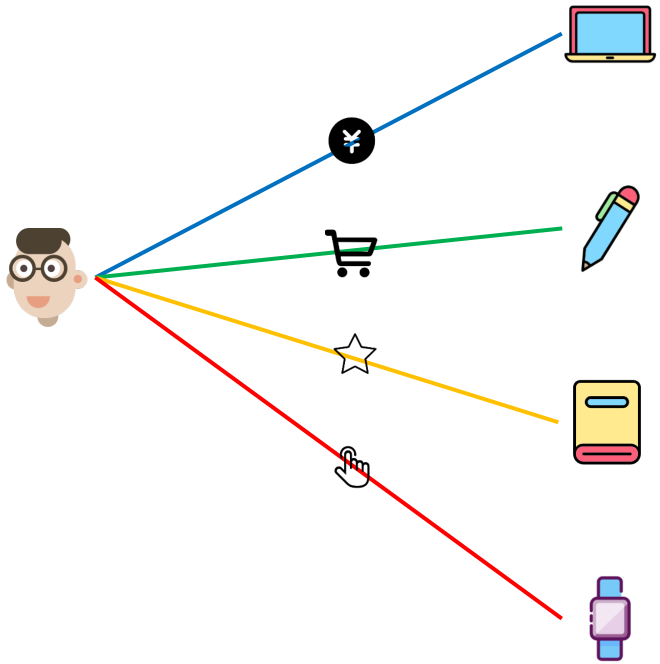
Problem Definition

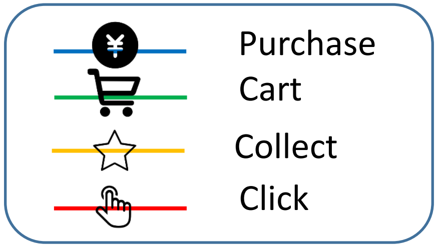
- Input:
  - User-item interaction data of T types of behaviors
- Output:
  - User-item interaction probability under target behavior



Challenge1: Behavior Strength

- Behavior-level
  - There may be intensity difference between behaviors
  - Behavior intensity is vague





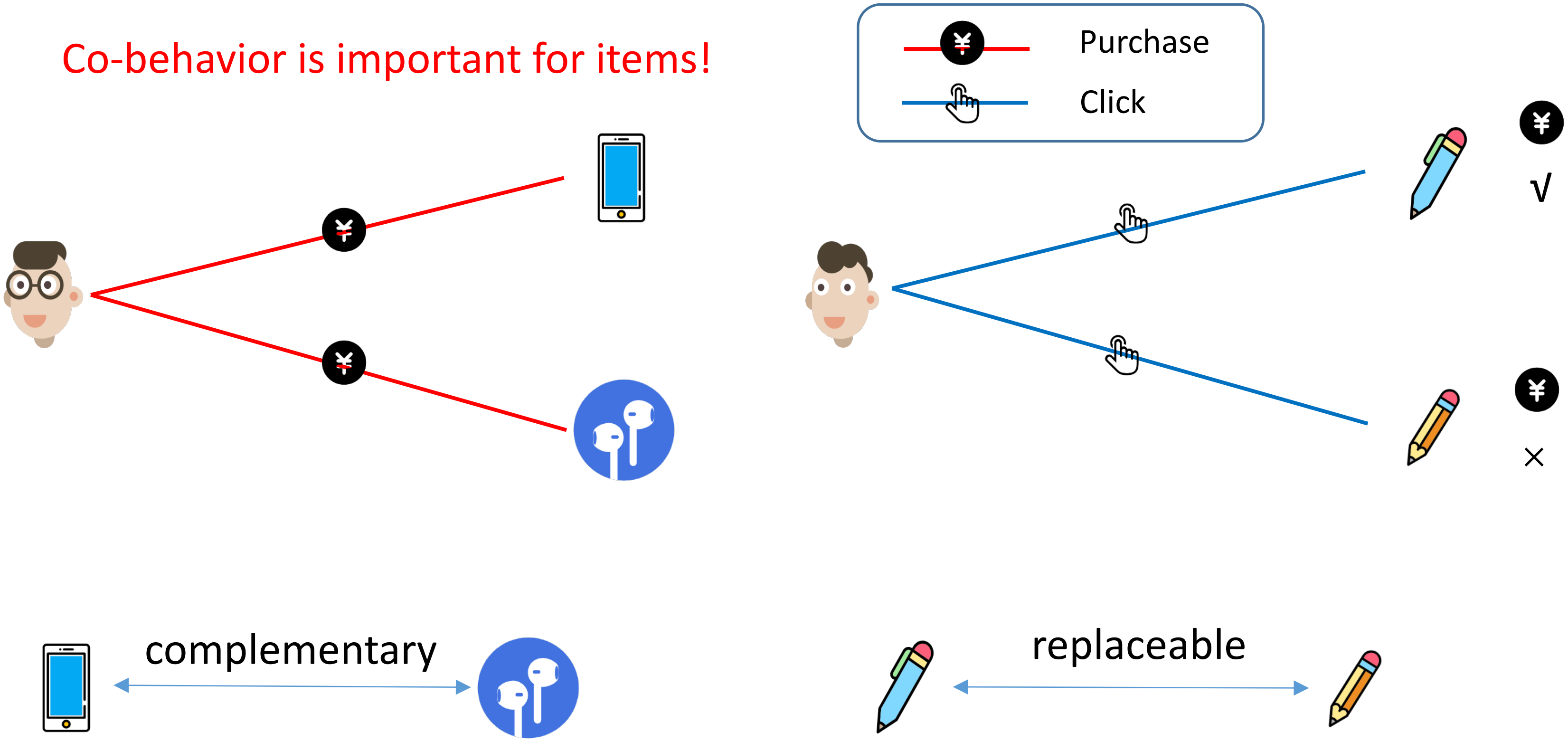
Purchase>Cart>Collect>Click ?

Purchase>Collect>Cart>Click ?

Purchase>Cart=Collect>Click ?

Challenge2: Behavior Semantics

- Item-level
  - Item relation is diverse among various types of behavior
  - Items may be complementary or replaceable or …



Co-behavior is important for items!

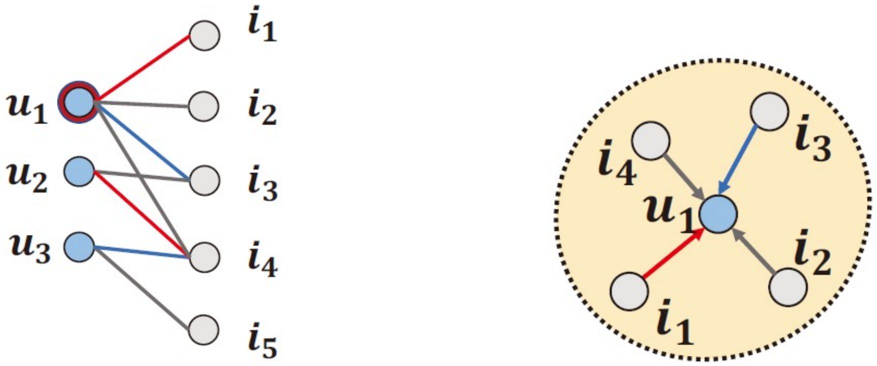
Existing Method

- Methods
  - Sampling based: MCBPR, BPRH,…
  - Multi-task based: CMF, NMTR, …

- Behavior Strength:
  - They must assume an artificial behavior-strength sequence (however, behaviors’ strength may be vague)
- Behavior Semantics:
  - Not considered at all

Methodology: Our MBGCN Model

- Why we use GCN?
  - Capture CF effect
  - Extract High-order information in multi-behavior data

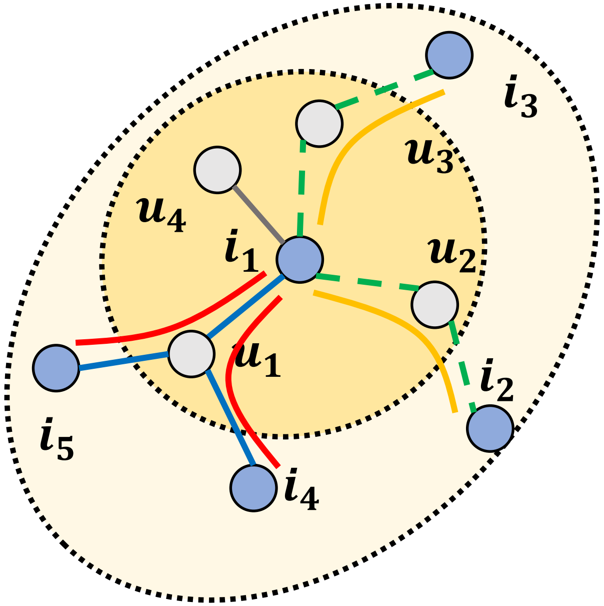


(a) U-I Interaction Graph

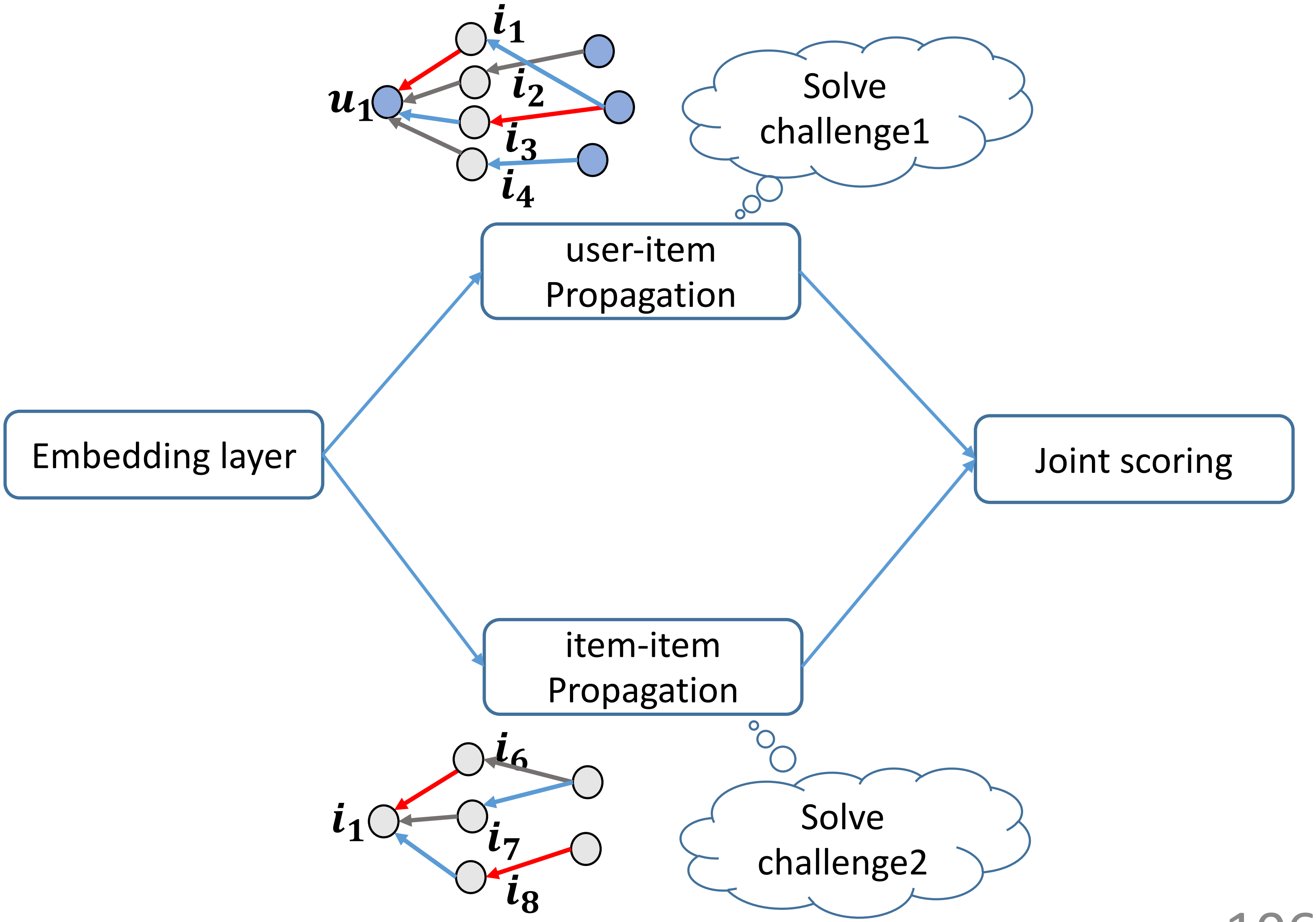
(b) Local Graph of u_1

Methodology: Our MBGCN Model

- Graph Construction
- Nodes: user, item
- Edges:
  - user-t-item (t represents a type of behavior)
- Meta-path:
  - item-t-user-t-item (t represents a type of behavior)



Methodology: Our MBGCN Model



Solve challenge1

user-item Propagation

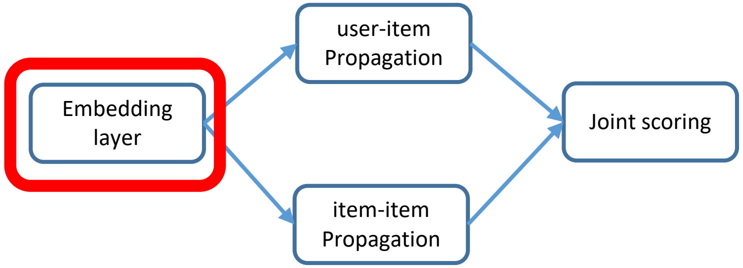
Embedding layer

Joint scoring

item-item Propagation

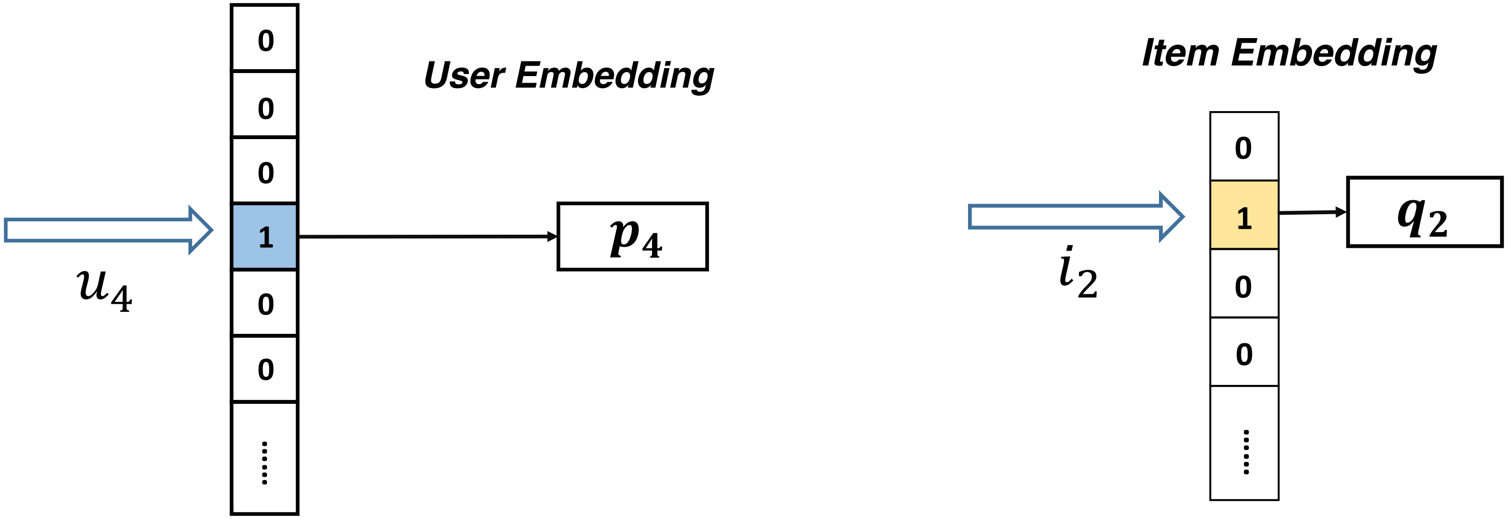
Solve challenge2

Our MBGCN Model

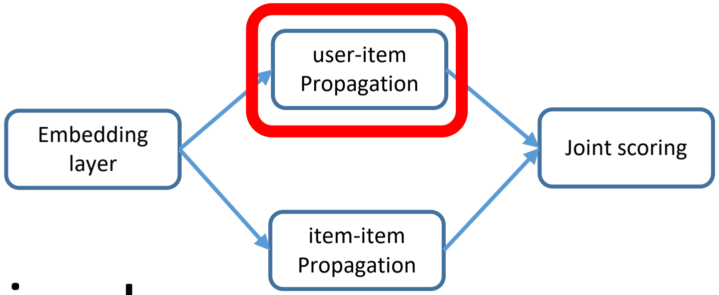


- Embedding layer
  - Convert user/item one-hot vector to user/item embedding

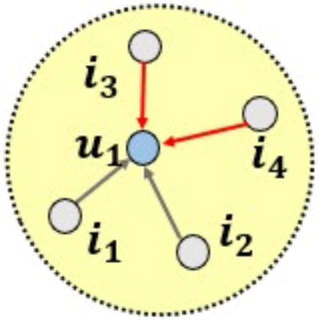
$$p_{u_k} = \boldsymbol{P} \cdot ID_k^U, \quad q_{i_j} = \boldsymbol{Q} \cdot ID_j^V$$

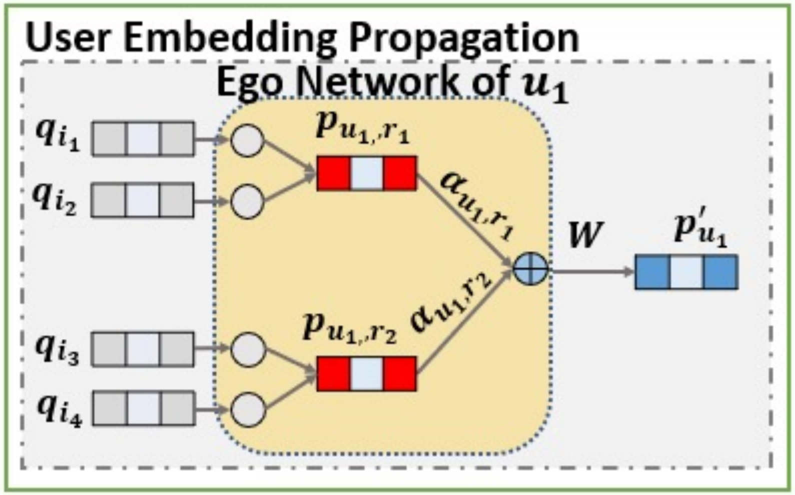


Our MBGCN Model



- Behavior-aware User-Item Propagation Layer
  - Item->User embedding propagation based on behavior types



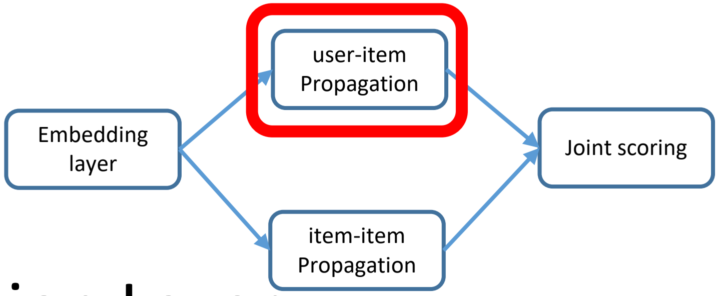


Solve Challenge1

Take behavior strength and user preference into account.

$$\alpha_{ut} = \frac{w_t \cdot n_{ut}}{\sum_{m \in N_r} w_m \cdot n_{um}}$$

Our MBGCN Model



- Behavior-aware User-Item Propagation Layer
  - Behavior importance calculation for each user

$$\alpha_{ut} = \frac{\boxed{w_t} \cdot \boxed{n_{ut}}}{\sum_{n \in N_r} w_m \cdot n_{um}}$$

Behavior count may imply user preference

$\boldsymbol{w_t}$

· behavior-wised importance weight of behavior t

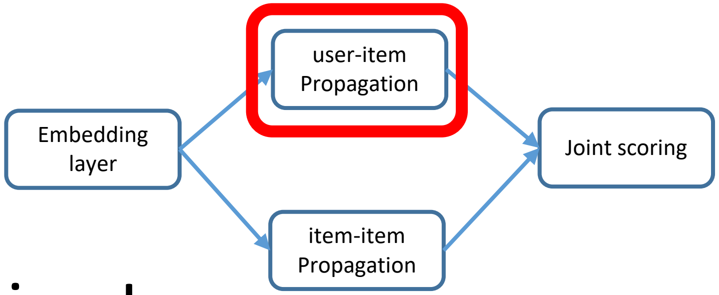
· the same for all users

$\boldsymbol{n_{ut}}$

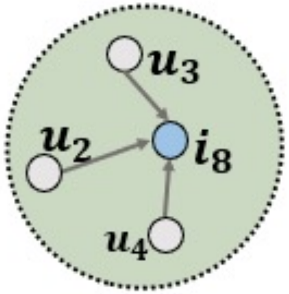
· count of behavior t operated by user u

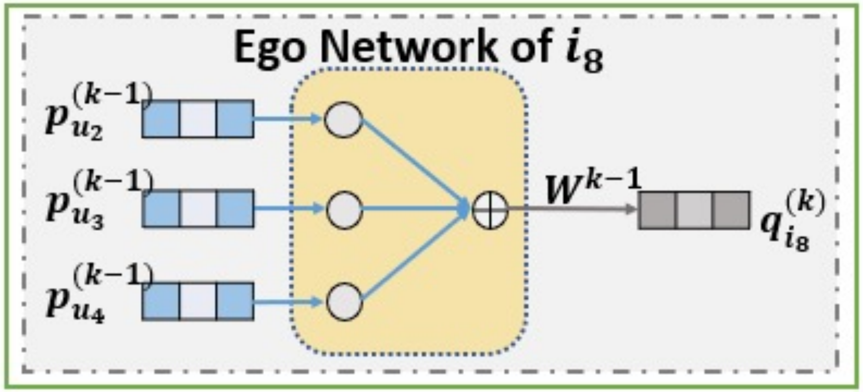
· different depends on user

Our MBGCN Model



- Behavior-aware User-Item Propagation Layer
  - User->Item embedding propagation



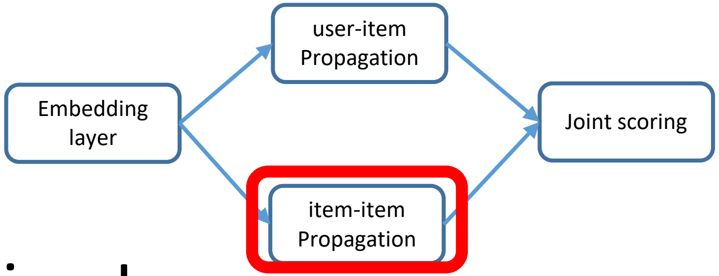


Capture user->item CF signal

$$q_i^{(l+1)} = W^{(l)} \cdot \text{aggregate}(p_j^{(l)} | j \in N_r^U(i))$$

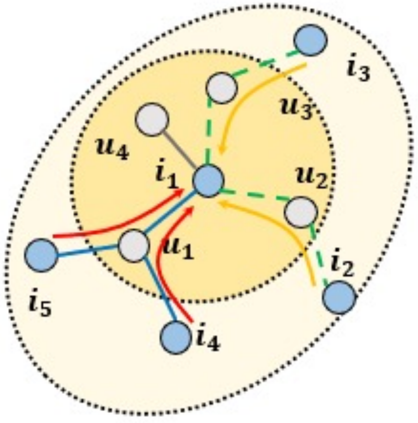
r is the target behavior

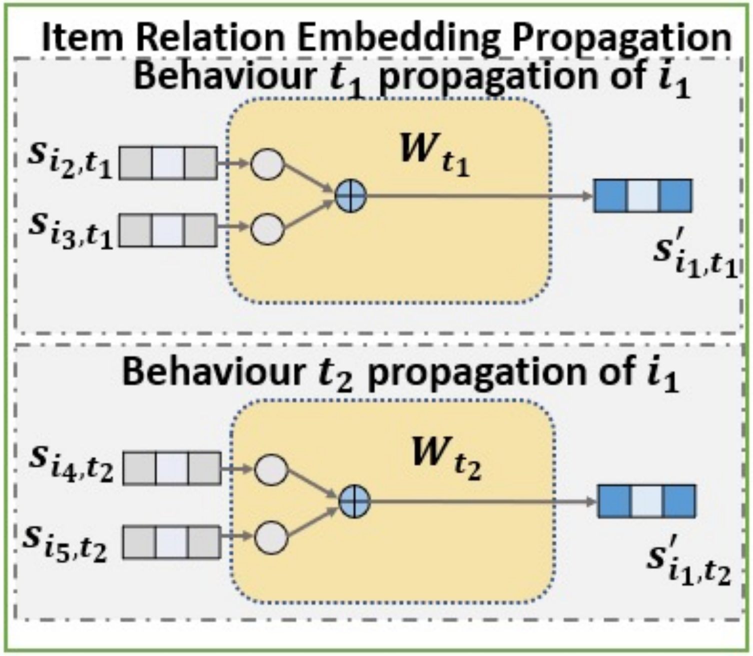
Our MBGCN Model



- Behavior-aware Item-Item Propagation Layer
  - Item->Item embedding propagation based on behavior types

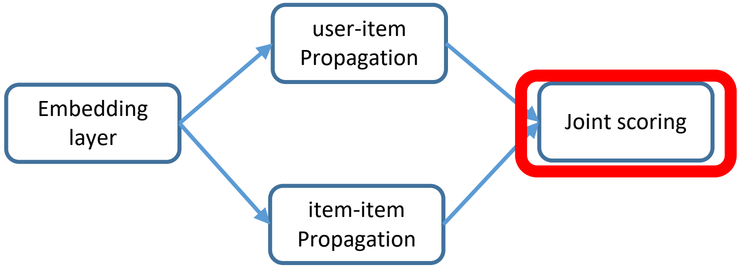
Item node will receive information from it’s neighbor item nodes




Item Relation Embedding Propagation
Behaviour t_1 propagation of i_1
s_{i_2,t_1}
s_{i_3,t_1}
W_{t_1}
s'_{i_1,t_1}
Behaviour t_2 propagation of i_1
s_{i_4,t_2}
s_{i_5,t_2}
W_{t_2}
s'_{i_1,t_2}

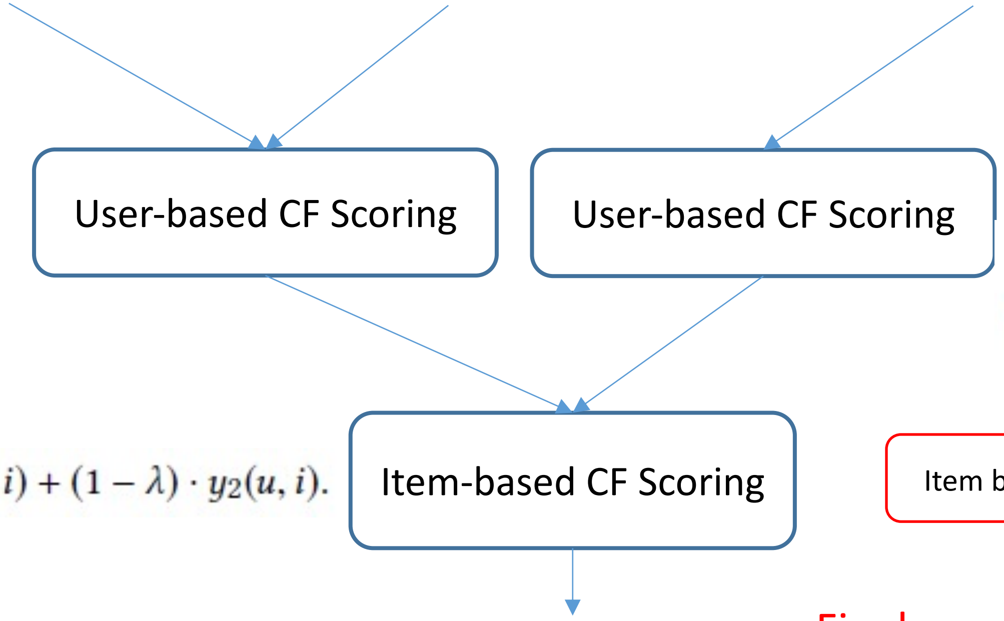
Solve Challenge2 by introducing item-item propagation

Our MBGCN Model



- Joint Prediction

p_u^* User embedding q_i^* Item embedding s_{it}^* Item relation embedding



$$y_1(u,i) = p_u^{*T} \cdot q_i^*$$

$$y_2(u,i) = \sum_{t \in N_r} \sum_{j \in N_t^I(u)} \frac{s_{jt}^{*T} \cdot \boxed{M_t} \cdot s_{it}^*}{|N_t^I(u)|},$$

Item behavior wised relation calculation matrix

$$y(u,i) = \lambda \cdot y_1(u,i) + (1-\lambda) \cdot y_2(u,i).$$

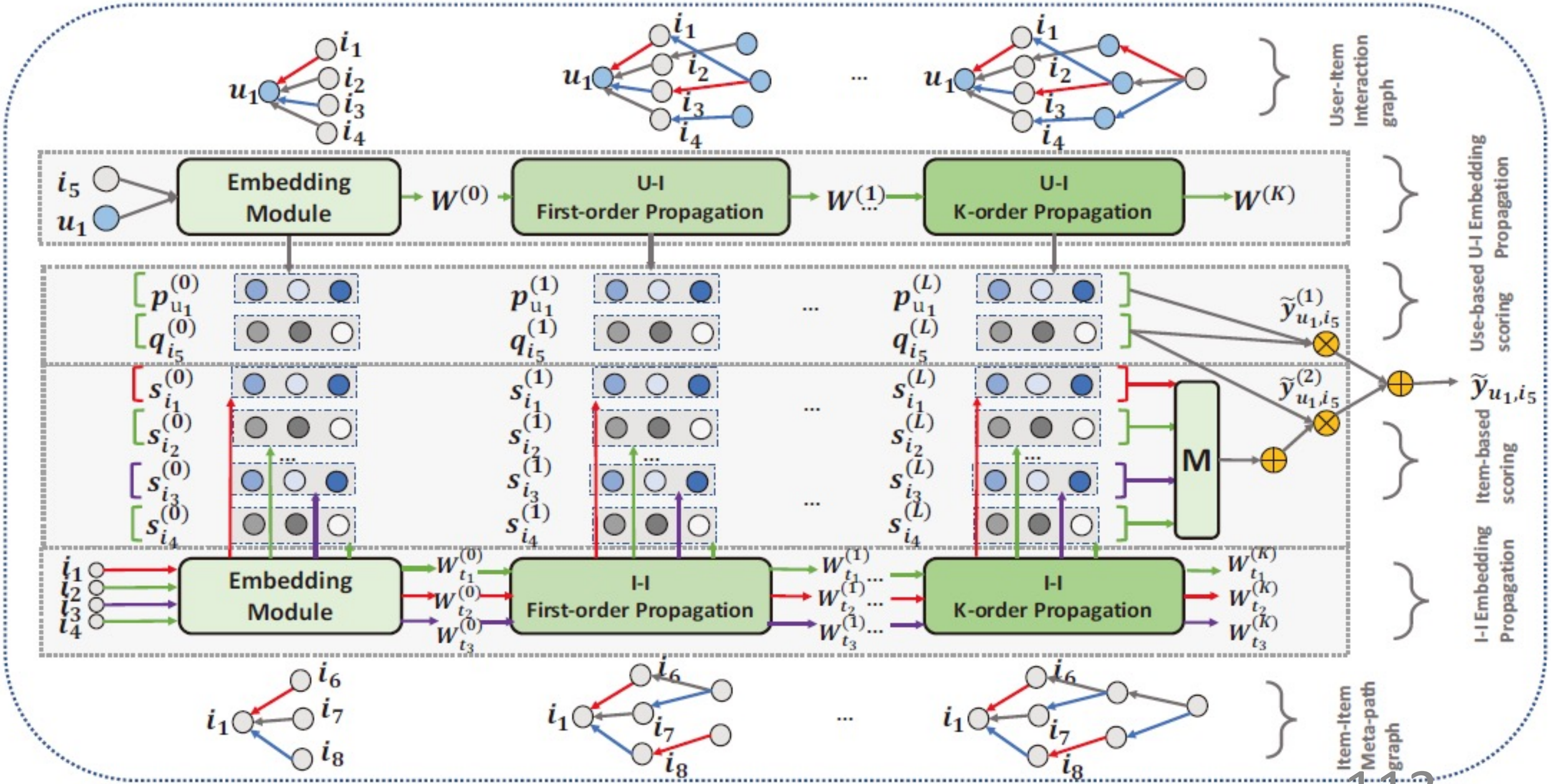
$y(u,i)$

Final score extracts both CF signal and behavior semantics!

- Loss function: BPR $$Loss = \sum_{(u,i,j) \in O} -ln\sigma(y(u,i) - y(u,j)) + \beta \cdot ||\Theta||^2,$$

Methodology: Our MBGCN Model

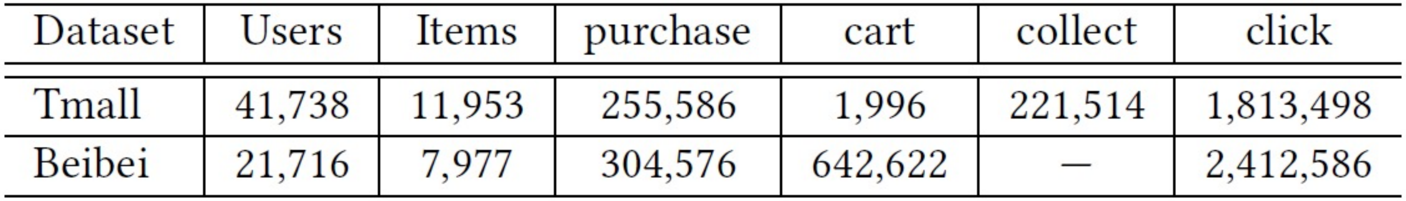
- Whole model



u_1 i_1 i_2 i_3 i_4
...
User-Item Interaction graph
i_5 u_1
Embedding Module
$W^{(0)}$
U-I First-order Propagation
$W^{(1)}$ ...
U-I K-order Propagation
$W^{(K)}$
U-I Embedding Propagation
$p_{u_1}^{(0)}$ $q_{i_5}^{(0)}$
$p_{u_1}^{(1)}$ $q_{i_5}^{(1)}$
$p_{u_1}^{(L)}$ $q_{i_5}^{(L)}$
Use-based scoring
$\tilde{y}_{u_1,i_5}^{(1)}$
$\tilde{y}_{u_1,i_5}^{(2)}$
$\tilde{y}_{u_1,i_5}$
$s_{i_1}^{(0)}$ $s_{i_2}^{(0)}$ $s_{i_3}^{(0)}$ $s_{i_4}^{(0)}$
$s_{i_1}^{(1)}$ $s_{i_2}^{(1)}$ $s_{i_3}^{(1)}$ $s_{i_4}^{(1)}$
$s_{i_1}^{(L)}$ $s_{i_2}^{(L)}$ $s_{i_3}^{(L)}$ $s_{i_4}^{(L)}$
M
Item-based scoring
i_1 i_2 i_3 i_4
Embedding Module
$W_{t_1}^{(0)}$ $W_{t_2}^{(0)}$ $W_{t_3}^{(0)}$
I-I First-order Propagation
$W_{t_1}^{(1)}$... $W_{t_2}^{(1)}$... $W_{t_3}^{(1)}$...
I-I K-order Propagation
$W_{t_1}^{(K)}$ $W_{t_2}^{(K)}$ $W_{t_3}^{(K)}$
I-I Embedding Propagation
i_1 i_6 i_7 i_8
Item-Item Meta-path graph

Experiments

- Dataset
  - Two real-world datasets collected from e-commerce platform



Dataset	Users	Items	purchase	cart	collect	click
Tmall	41,738	11,953	255,586	1,996	221,514	1,813,498
Beibei	21,716	7,977	304,576	642,622	—	2,412,586

- Evaluation protocols
  - Top-K evaluation with two metrics Recall and NDCG

- Baseline
  - Single-behavior models:
    - BPR-MF(UAI09), NeuMF(WWW17), GraphSAGE-OB(NeurIPs17), NGCF-OB(SIGIR19),
  - Multi-behavior models:
    - NMTR(ICDE19), MC-BPR(RecSys16), GraphSAGE-MB(NeurIPs17), NGCF-MB(SIGIR19), RGCN(ESWC2018)

Experiments

- Overall Comparison
  - Tmall

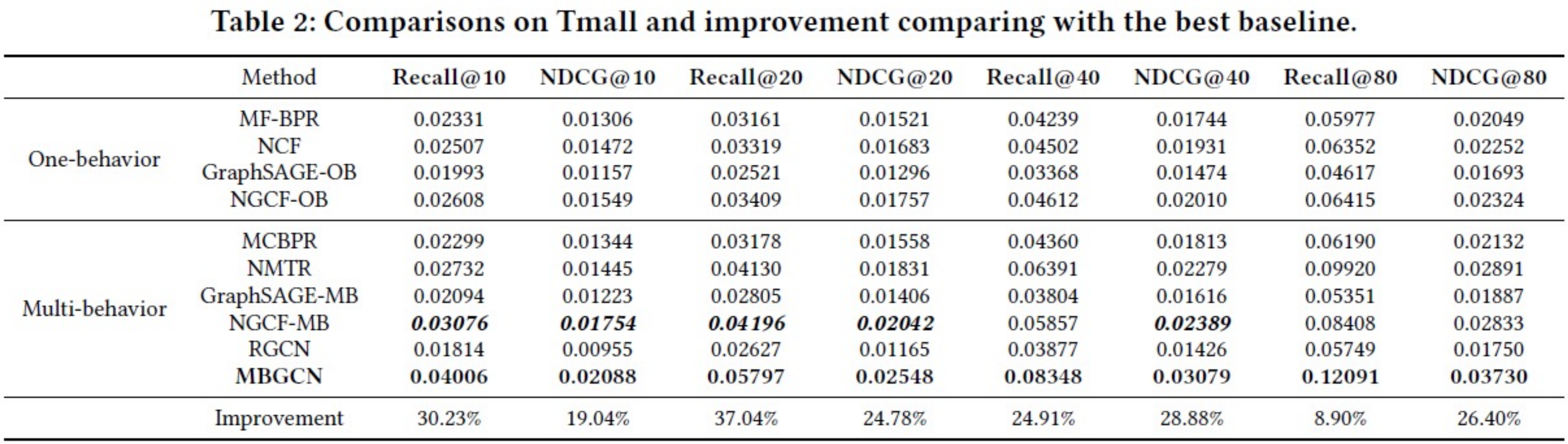


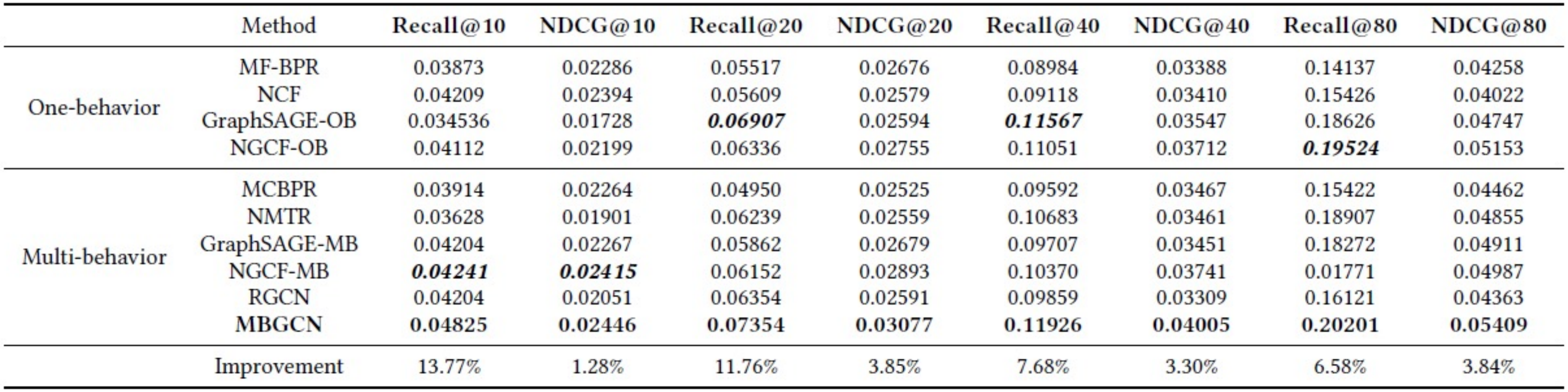
Table 2: Comparisons on Tmall and improvement comparing with the best baseline.

	Method	**Recall@10**	**NDCG@10**	**Recall@20**	**NDCG@20**	**Recall@40**	**NDCG@40**	**Recall@80**	**NDCG@80**
One-behavior	MF-BPR	0.02331	0.01306	0.03161	0.01521	0.04239	0.01744	0.05977	0.02049
	NCF	0.02507	0.01472	0.03319	0.01683	0.04502	0.01931	0.06352	0.02252
	GraphSAGE-OB	0.01993	0.01157	0.02521	0.01296	0.03368	0.01474	0.04617	0.01693
	NGCF-OB	0.02608	0.01549	0.03409	0.01757	0.04612	0.02010	0.06415	0.02324
Multi-behavior	MCBPR	0.02299	0.01344	0.03178	0.01558	0.04360	0.01813	0.06190	0.02132
	NMTR	0.02732	0.01445	0.04130	0.01831	0.06391	0.02279	0.09920	0.02891
	GraphSAGE-MB	0.02094	0.01223	0.02805	0.01406	0.03804	0.01616	0.05351	0.01887
	NGCF-MB	***0.03076***	***0.01754***	***0.04196***	***0.02042***	0.05857	***0.02389***	0.08408	0.02833
	RGCN	0.01814	0.00955	0.02627	0.01165	0.03877	0.01426	0.05749	0.01750
	MBGCN	**0.04006**	**0.02088**	**0.05797**	**0.02548**	**0.08348**	**0.03079**	**0.12091**	**0.03730**
	Improvement	30.23%	19.04%	37.04%	24.78%	24.91%	28.88%	8.90%	26.40%

Experiments

- Overall Comparison
  - Beibei

Table 3: Comparisons on Beibei and improvement comparing with the best baseline.

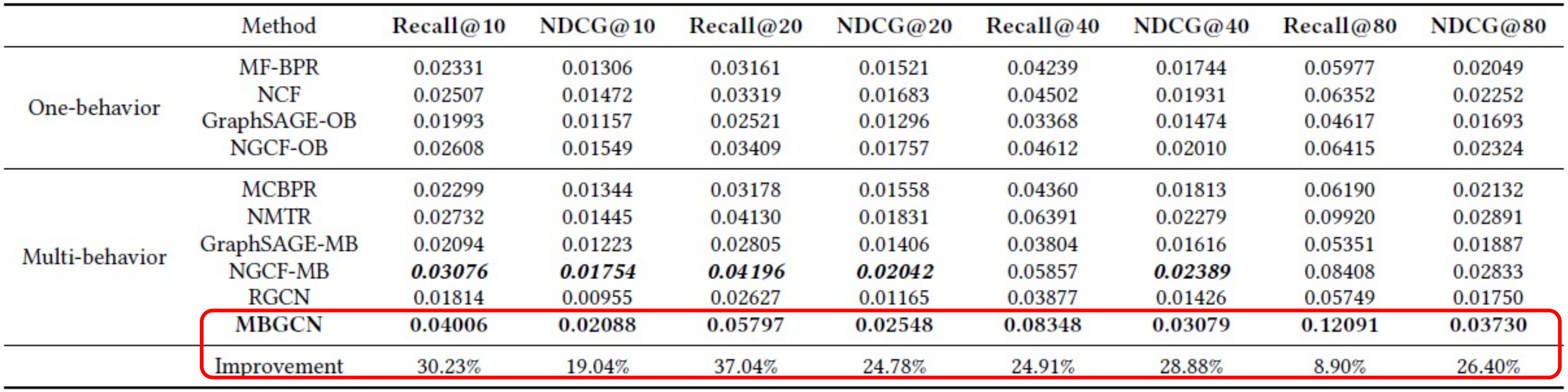


	Method	Recall@10	NDCG@10	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80
One-behavior	MF-BPR	0.03873	0.02286	0.05517	0.02676	0.08984	0.03388	0.14137	0.04258
	NCF	0.04209	0.02394	0.05609	0.02579	0.09118	0.03410	0.15426	0.04022
	GraphSAGE-OB	0.034536	0.01728	***0.06907***	0.02594	***0.11567***	0.03547	0.18626	0.04747
	NGCF-OB	0.04112	0.02199	0.06336	0.02755	0.11051	0.03712	***0.19524***	0.05153
Multi-behavior	MCBPR	0.03914	0.02264	0.04950	0.02525	0.09592	0.03467	0.15422	0.04462
	NMTR	0.03628	0.01901	0.06239	0.02559	0.10683	0.03461	0.18907	0.04855
	GraphSAGE-MB	0.04204	0.02267	0.05862	0.02679	0.09707	0.03451	0.18272	0.04911
	NGCF-MB	***0.04241***	***0.02415***	0.06152	0.02893	0.10370	0.03741	0.01771	0.04987
	RGCN	0.04204	0.02051	0.06354	0.02591	0.09859	0.03309	0.16121	0.04363
	MBGCN	**0.04825**	**0.02446**	**0.07354**	**0.03077**	**0.11926**	**0.04005**	**0.20201**	**0.05409**
	Improvement	13.77%	1.28%	11.76%	3.85%	7.68%	3.30%	6.58%	3.84%

Experiments

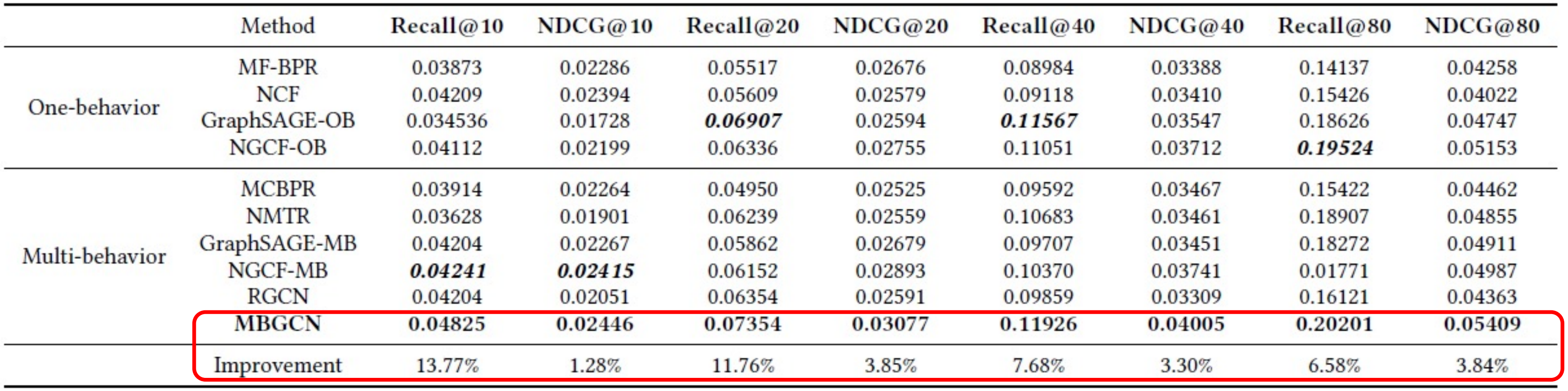
- Overall Comparison

Table 2: Comparisons on Tmall and improvement comparing with the best baseline.



	Method	Recall@10	NDCG@10	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80
One-behavior	MF-BPR	0.02331	0.01306	0.03161	0.01521	0.04239	0.01744	0.05977	0.02049
	NCF	0.02507	0.01472	0.03319	0.01683	0.04502	0.01931	0.06352	0.02252
	GraphSAGE-OB	0.01993	0.01157	0.02521	0.01296	0.03368	0.01474	0.04617	0.01693
	NGCF-OB	0.02608	0.01549	0.03409	0.01757	0.04612	0.02010	0.06415	0.02324
Multi-behavior	MCBPR	0.02299	0.01344	0.03178	0.01558	0.04360	0.01813	0.06190	0.02132
	NMTR	0.02732	0.01445	0.04130	0.01831	0.06391	0.02279	0.09920	0.02891
	GraphSAGE-MB	0.02094	0.01223	0.02805	0.01406	0.03804	0.01616	0.05351	0.01887
	NGCF-MB	***0.03076***	***0.01754***	***0.04196***	***0.02042***	0.05857	***0.02389***	0.08408	0.02833
	RGCN	0.01814	0.00955	0.02627	0.01165	0.03877	0.01426	0.05749	0.01750
	MBGCN	**0.04006**	**0.02088**	**0.05797**	**0.02548**	**0.08348**	**0.03079**	**0.12091**	**0.03730**
	Improvement	30.23%	19.04%	37.04%	24.78%	24.91%	28.88%	8.90%	26.40%

Table 3: Comparisons on Beibei and improvement comparing with the best baseline.



	Method	Recall@10	NDCG@10	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80
One-behavior	MF-BPR	0.03873	0.02286	0.05517	0.02676	0.08984	0.03388	0.14137	0.04258
	NCF	0.04209	0.02394	0.05609	0.02579	0.09118	0.03410	0.15426	0.04022
	GraphSAGE-OB	0.034536	0.01728	***0.06907***	0.02594	***0.11567***	0.03547	0.18626	0.04747
	NGCF-OB	0.04112	0.02199	0.06336	0.02755	0.11051	0.03712	***0.19524***	0.05153
Multi-behavior	MCBPR	0.03914	0.02264	0.04950	0.02525	0.09592	0.03467	0.15422	0.04462
	NMTR	0.03628	0.01901	0.06239	0.02559	0.10683	0.03461	0.18907	0.04855
	GraphSAGE-MB	0.04204	0.02267	0.05862	0.02679	0.09707	0.03451	0.18272	0.04911
	NGCF-MB	***0.04241***	***0.02415***	0.06152	0.02893	0.10370	0.03741	0.01771	0.04987
	RGCN	0.04204	0.02051	0.06354	0.02591	0.09859	0.03309	0.16121	0.04363
	MBGCN	**0.04825**	**0.02446**	**0.07354**	**0.03077**	**0.11926**	**0.04005**	**0.20201**	**0.05409**
	Improvement	13.77%	1.28%	11.76%	3.85%	7.68%	3.30%	6.58%	3.84%

Observation1: Our model performs the best

Experiments

- Overall Comparison

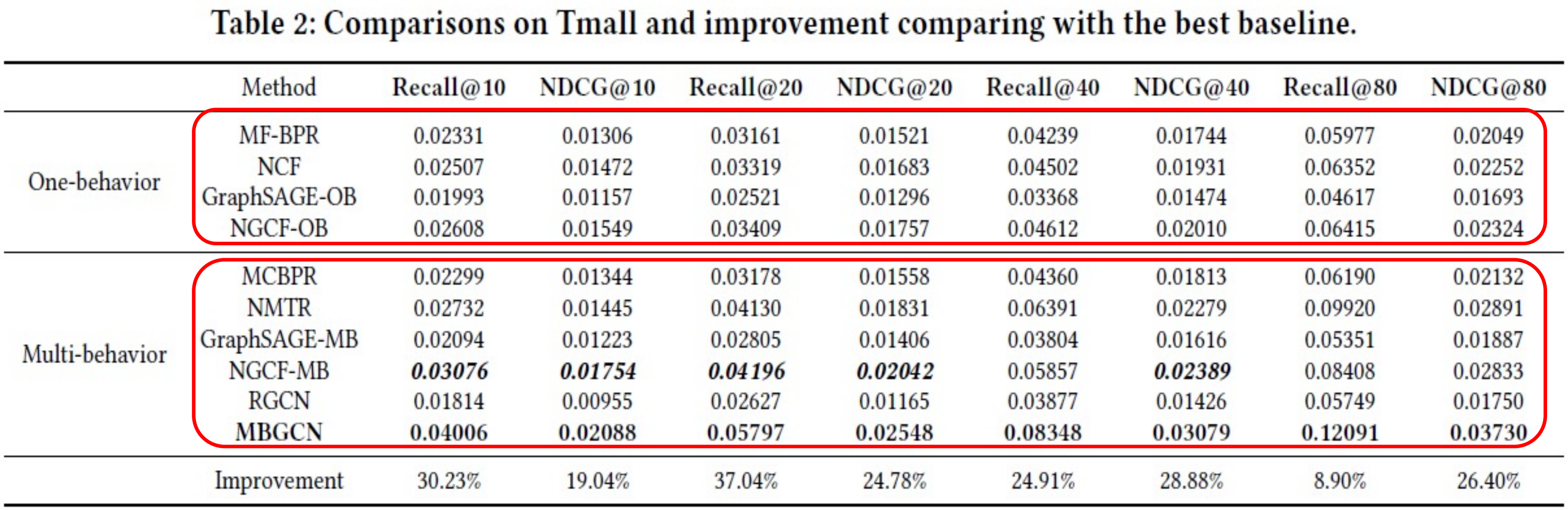


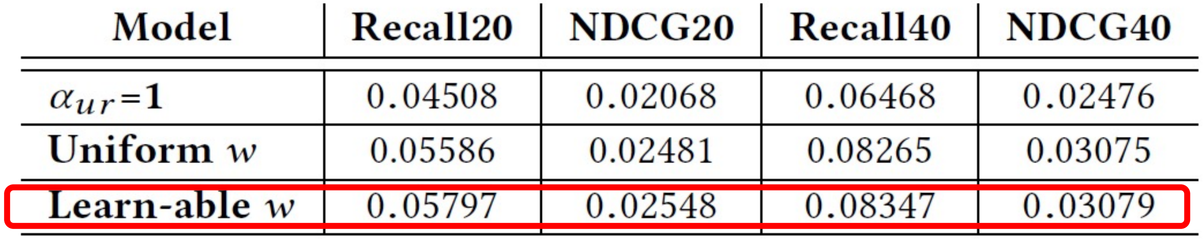
Table 2: Comparisons on Tmall and improvement comparing with the best baseline.

	Method	Recall@10	NDCG@10	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80
One-behavior	MF-BPR	0.02331	0.01306	0.03161	0.01521	0.04239	0.01744	0.05977	0.02049
	NCF	0.02507	0.01472	0.03319	0.01683	0.04502	0.01931	0.06352	0.02252
	GraphSAGE-OB	0.01993	0.01157	0.02521	0.01296	0.03368	0.01474	0.04617	0.01693
	NGCF-OB	0.02608	0.01549	0.03409	0.01757	0.04612	0.02010	0.06415	0.02324
Multi-behavior	MCBPR	0.02299	0.01344	0.03178	0.01558	0.04360	0.01813	0.06190	0.02132
	NMTR	0.02732	0.01445	0.04130	0.01831	0.06391	0.02279	0.09920	0.02891
	GraphSAGE-MB	0.02094	0.01223	0.02805	0.01406	0.03804	0.01616	0.05351	0.01887
	NGCF-MB	***0.03076***	***0.01754***	***0.04196***	***0.02042***	0.05857	***0.02389***	0.08408	0.02833
	RGCN	0.01814	0.00955	0.02627	0.01165	0.03877	0.01426	0.05749	0.01750
	MBGCN	**0.04006**	**0.02088**	**0.05797**	**0.02548**	**0.08348**	**0.03079**	**0.12091**	**0.03730**
	Improvement	30.23%	19.04%	37.04%	24.78%	24.91%	28.88%	8.90%	26.40%

Observation2: Multi-behavior models perform better than single-behavior models

Experiments

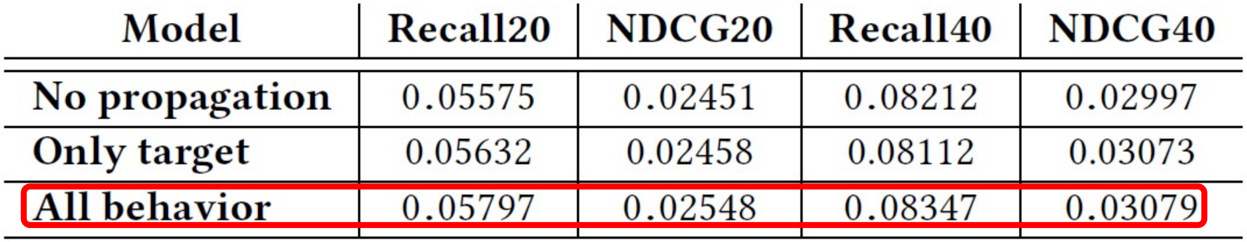
- Ablation Study on Model structure
  - Ablation study of user-item propagation weight



Model	Recall20	NDCG20	Recall40	NDCG40
α_{ur}=1	0.04508	0.02068	0.06468	0.02476
Uniform w	0.05586	0.02481	0.08265	0.03075
Learn-able w	0.05797	0.02548	0.08347	0.03079

Learn-able *w* is the best!

  - Ablation study of item-item propagation method

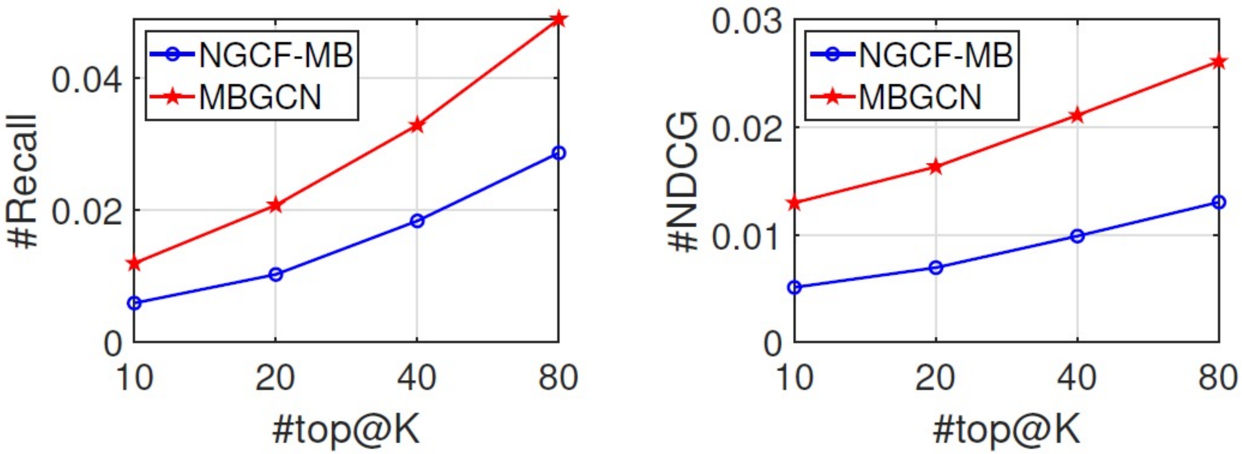


Model	Recall20	NDCG20	Recall40	NDCG40
No propagation	0.05575	0.02451	0.08212	0.02997
Only target	0.05632	0.02458	0.08112	0.03073
All behavior	0.05797	0.02548	0.08347	0.03079

It’s reasonable to have item-item propagation based on all behavior!

Experiments

- Cold-start Problem Study
  - Recommendation for cold-start user
  - Learn users' interest only from auxiliary behaviors



(a) Recall

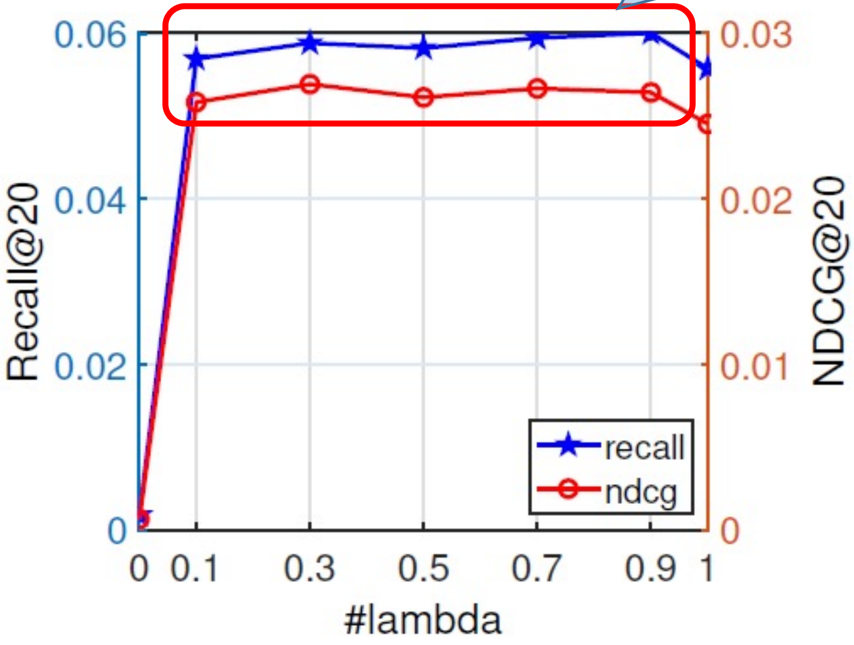
(b) NDCG

Our model can alleviate cold-start problem better!

Experiments

- Hyper-parameter Study

The model is not sensitive to λ.

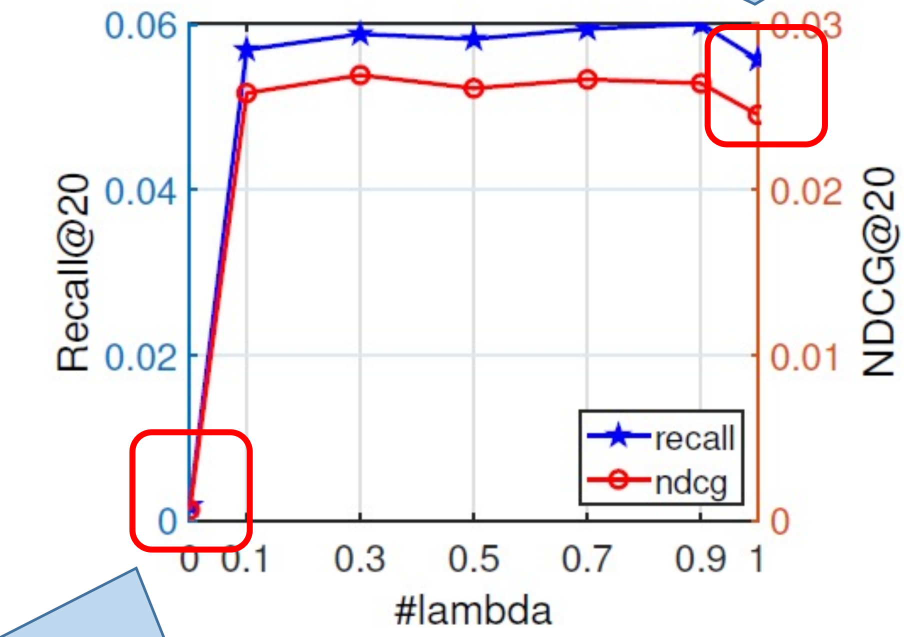


(a) Study of λ

Experiments

- Hyper-parameter Study

Item-Item Propagation is useful!



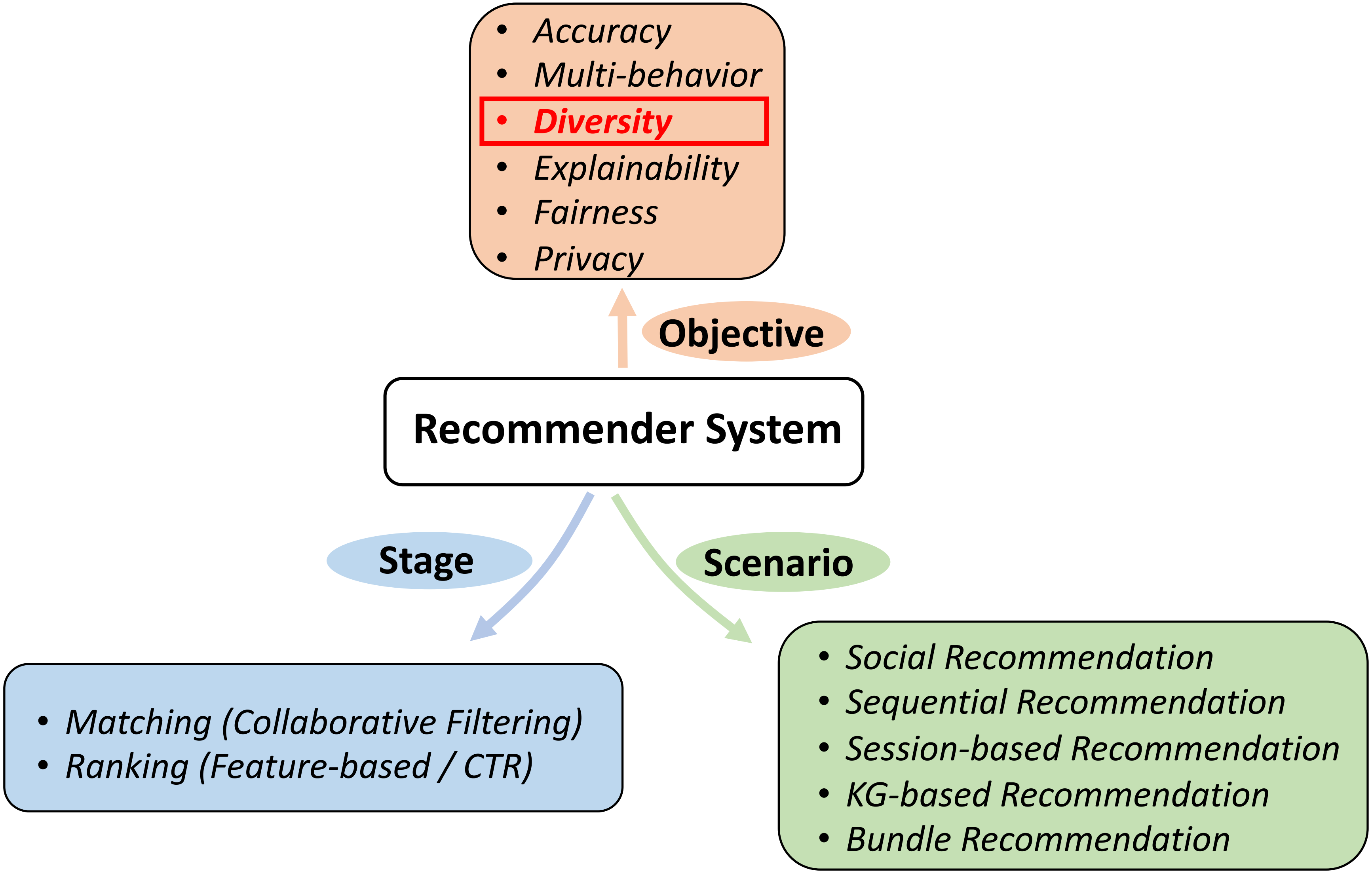
(a) Study of λ

User-Item Propagation is essential!

Conclusion

- We approach the problem of multi-behavior recommendation.
- We develop a MBGCN method with *user-item propagation layer* and *item-item propagation layer* to address two major challenges on modeling *behavior strength* and *behavior semantics*.
- We do experiment on two real-world datasets to demonstrate the superiority of our MBGCN model.

Recent advances of GNN-based RecSys



- *Accuracy*
- *Multi-behavior*
- ***Diversity***
- *Explainability*
- *Fairness*
- *Privacy*

Objective

Recommender System

Stage

Scenario

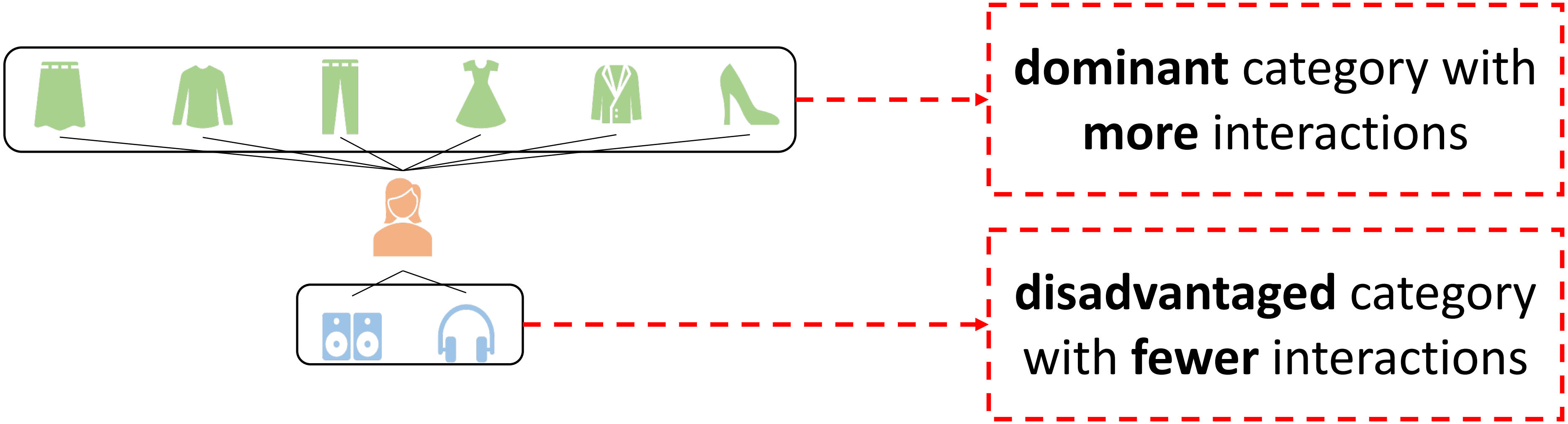
- *Matching (Collaborative Filtering)*
- *Ranking (Feature-based / CTR)*

- *Social Recommendation*
- *Sequential Recommendation*
- *Session-based Recommendation*
- *KG-based Recommendation*
- *Bundle Recommendation*

DGCN: Diversified Recommendation with Graph Convolutional Networks.
Zheng, Y., Gao, C., Chen, L., Jin, D., & Li, Y. WWW 2021

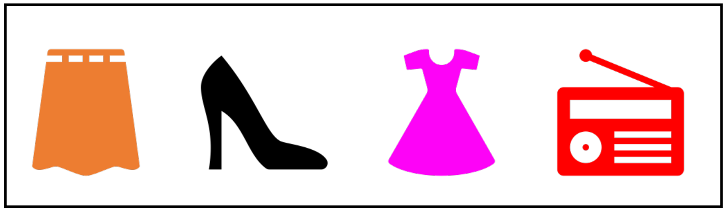
Background

- How to measure a recommender system?
  - accuracy, diversity, freshness, novelty…
- Diversity: dis-similarity among the recommended items



dominant category with **more** interactions

disadvantaged category with **fewer** interactions

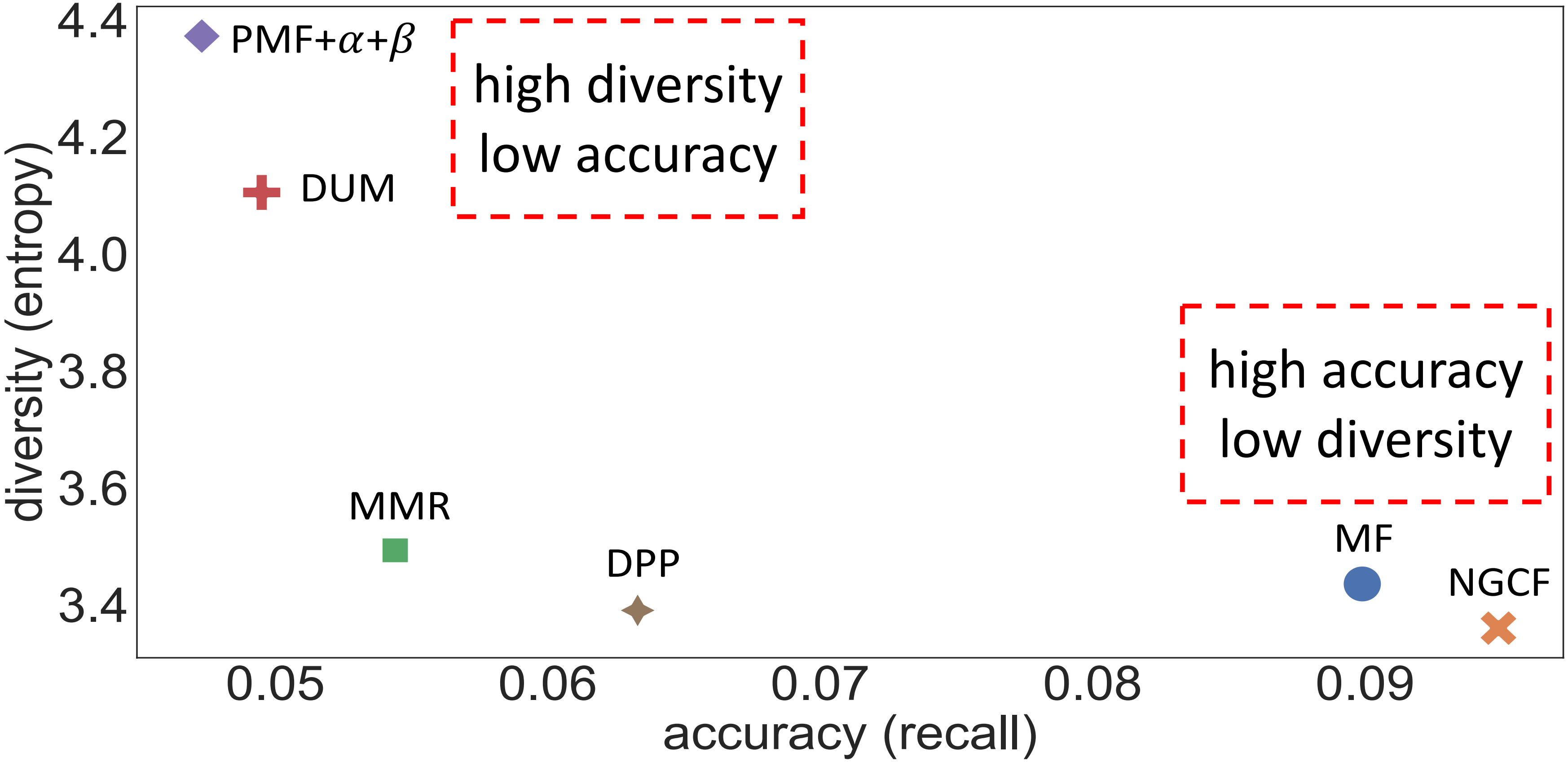


☹ accurate but redundant

☺ accurate and diverse

Background

- Having both accuracy and diversity is challenging
- **Accuracy-Diversity dilemma**



high diversity
low accuracy

high accuracy
low diversity

Goal: better **trade-off** between accuracy and diversity

Existing solutions

- Re-ranking (usually heuristics and greedy), e.g. MMR[1][2]
- First accuracy, then diversity
  - Step 1: Generate candidates (accuracy)
  - Step 2: Re-rank candidates (diversify with some loss on accuracy)

$$MMR \overset{\text{def}}{=} Arg \max_{D_i \in R \setminus S} \left[\lambda(Sim_1(D_i, Q) - (1-\lambda) \max_{D_j \in S} Sim_2(D_i, D_j)) \right]$$

Accuracy and diversity are **decoupled**! ☹

[1] Carbonell, J., & Goldstein, J. (1998, August). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 335-336).

[2] Ziegler, C. N., McNee, S. M., Konstan, J. A., & Lausen, G. (2005, May). Improving recommendation lists through topic diversification. In Proceedings of the 14th international conference on World Wide Web (pp. 22-32).

Challenges

- Insufficient diversity signals in matching models
- Upstream matching models are unaware of diversification

- Sample bias with respect to item category
- Dominant categories have much more samples than disadvantaged categories

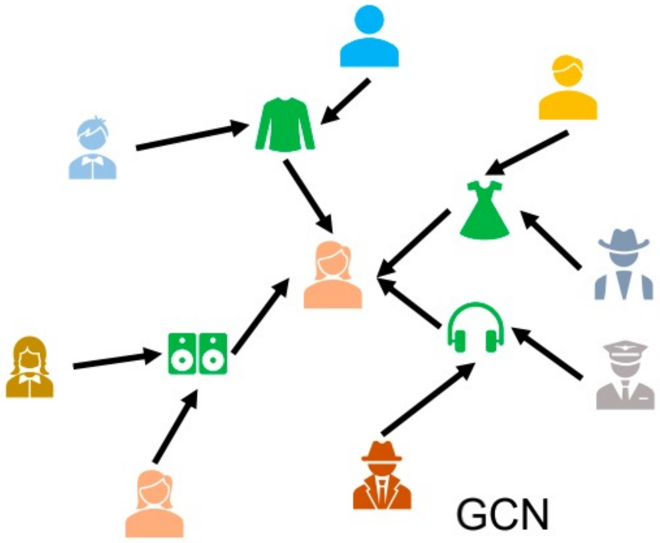
- Accuracy-diversity dilemma
- Higher diversity is often at the cost of lower accuracy

Methodology: Our DGCN Model

- Diversified recommendation with Graph Convolutional Networks (DGCN)

- **Challenge 1**: insufficient diversity signals in matching models
- **Our proposal**: perform diversification with GCN

- **Benefit 1**: diversify during matching instead of diversify after matching (**challenge 1 addressed**)
- **Benefit 2**: higher order neighbors tend to cover more diverse items

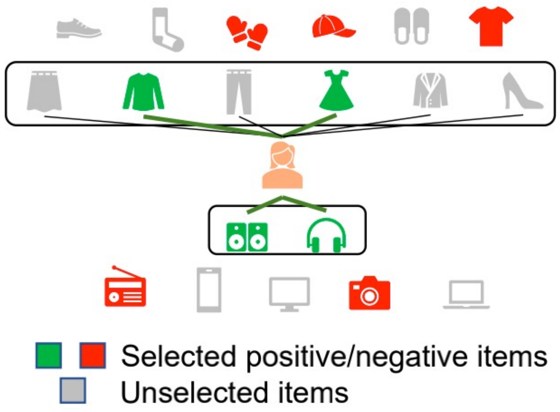


Methodology: Our DGCN Model

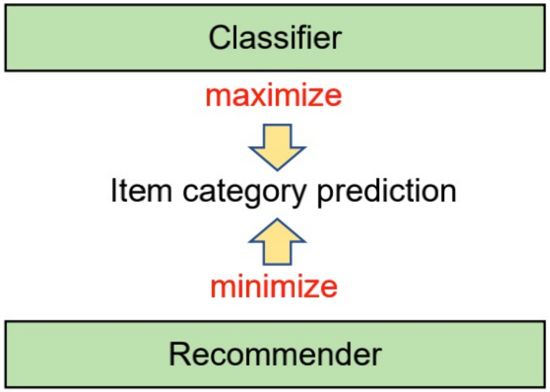
- Diversified recommendation with Graph Convolutional Networks (DGCN)

- **Challenge 2**: sample bias with respect to item category
- **Our proposal**:
  - Diversified neighbor discovering and negative sampling
  - Intuition: balance dominant and disadvantaged category

  - Adversarial learning
  - Intuition: remove category information from item embedding



🟩 🟥 Selected positive/negative items
▨ Unselected items



Classifier

maximize

Item category prediction

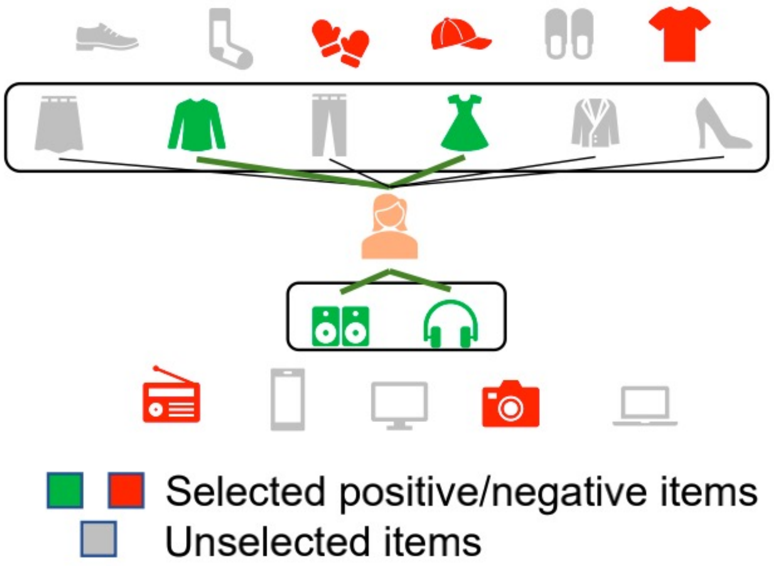
minimize

Recommender

Methodology: Our DGCN Model

- Diversified recommendation with Graph Convolutional Networks (DGCN)

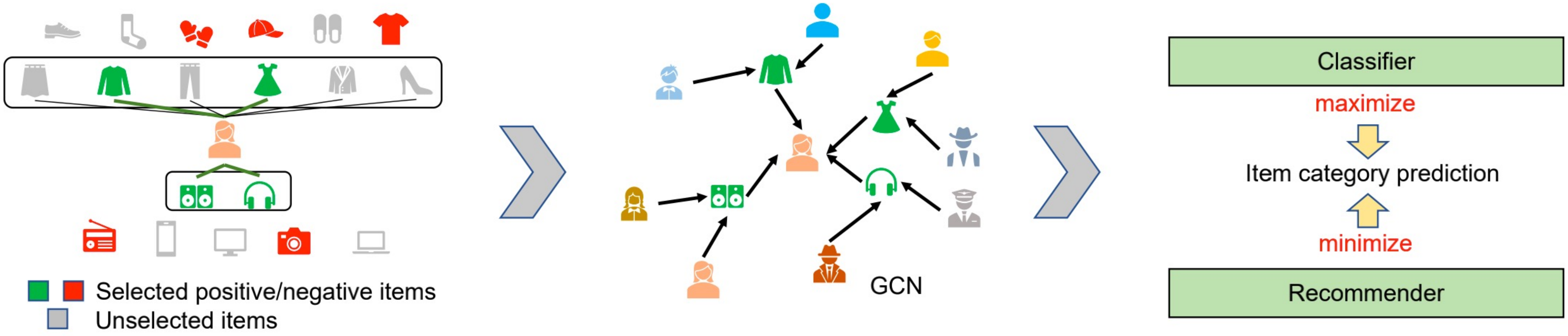
- **Challenge 3**: Accuracy-diversity dilemma
- **Our proposal**:
  - **Tunable** neighbor discovering and negative sampling
  - Two hyper-parameters are introduced to perform **trade-off** between accuracy and diversity



Methodology: Our DGCN Model

- Diversified recommendation with Graph Convolutional Networks (DGCN)

- Diversify during matching with GCN
- Diversified neighbor discovering and negative sampling
- Adversarial learning



🟩 🟥 Selected positive/negative items

▨ Unselected items

GCN

Classifier

maximize

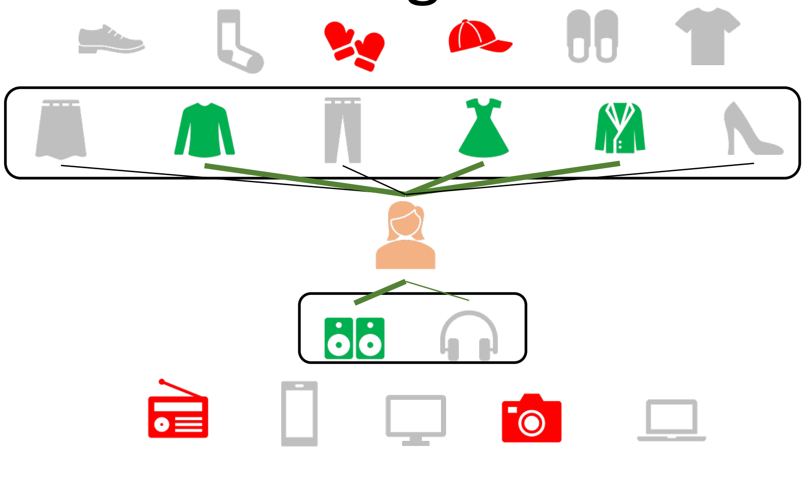
Item category prediction

minimize

Recommender

Methodology: Our DGCN Model

- Diversified neighbor discovering and negative sampling



🟩 🟥 Selected positive/negative items

⬜ Unselected items

select node neighbors and negative items **randomly**

#neighbors (positive samples):

clothes >> electronics

#negative samples:

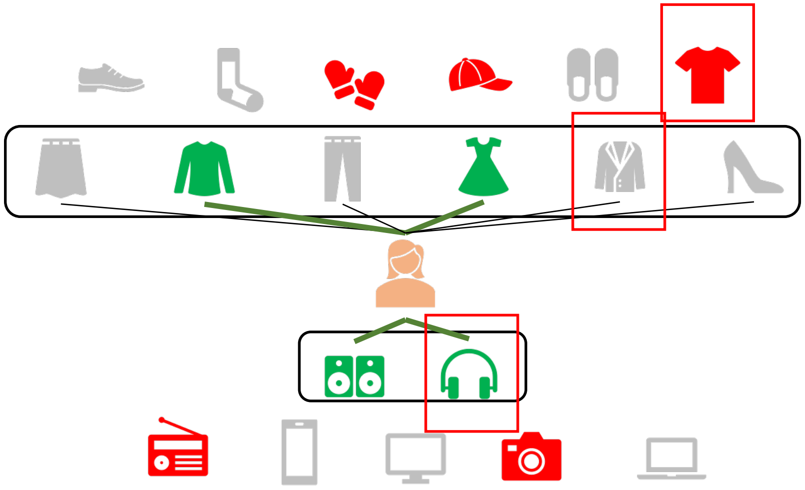
clothes ≈ electonics

#recommended items:

clothes >> electronics

low diversity

idea: select more electronics as neighbors and more clothes as negative items

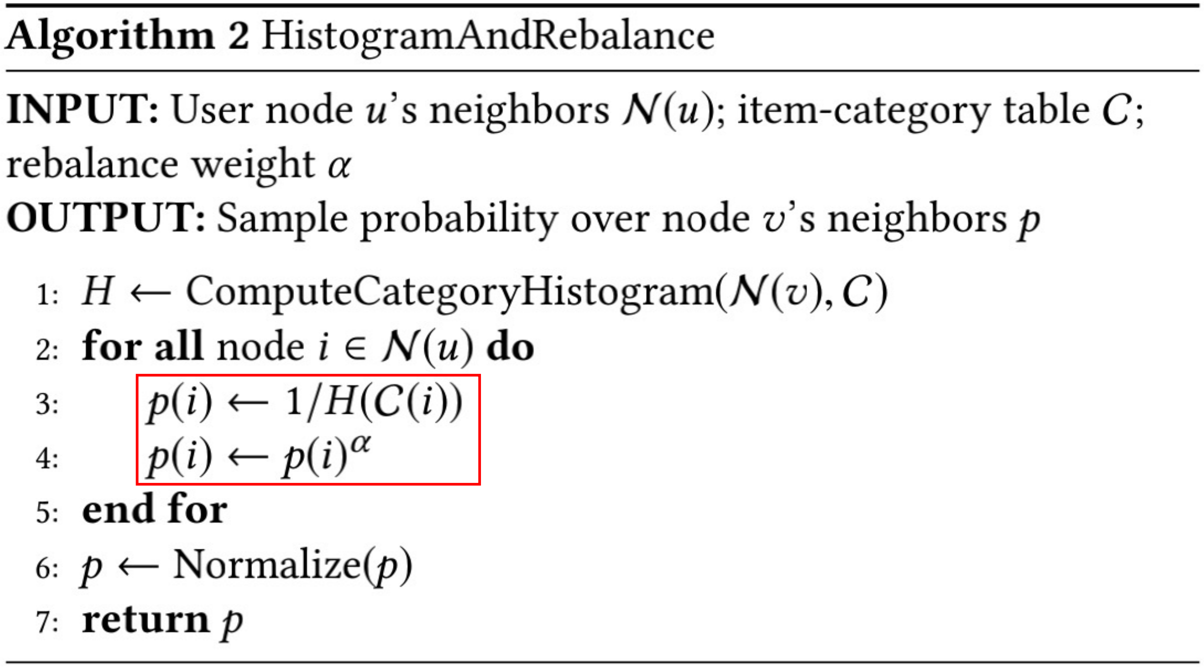


🟩 🟥 Selected positive/negative items

⬜ Unselected items

Methodology: Our DGCN Model

- Diversified **neighbor discovering**



Algorithm 2 HistogramAndRebalance

INPUT: User node u's neighbors $\mathcal{N}(u)$; item-category table C; rebalance weight α

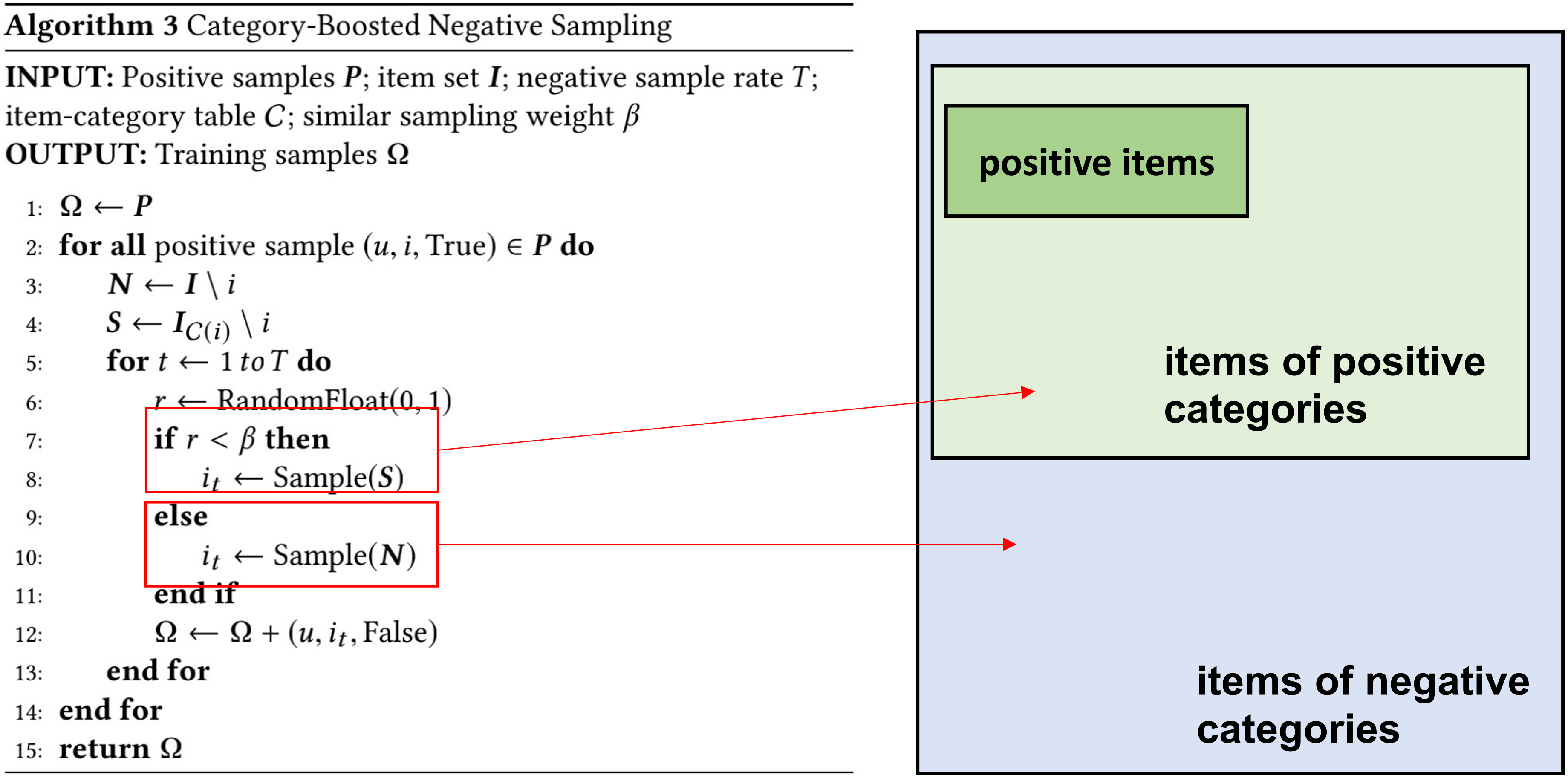
OUTPUT: Sample probability over node v's neighbors p

```
$H \leftarrow$ ComputeCategoryHistogram($\mathcal{N}(v), C$)
for all node $i \in \mathcal{N}(u)$ do
    $p(i) \leftarrow 1/H(C(i))$
    $p(i) \leftarrow p(i)^\alpha$
end for
$p \leftarrow$ Normalize($p$)
return $p$
```

1. Compute category histogram for each user's interacted items
2. Take inverse of the histogram to reweight each interacted item
3. Introduce α to perform trade-off (take exp to smooth)

Methodology: Our DGCN Model

- Diversified **negative sampling**



Algorithm 3 Category-Boosted Negative Sampling

INPUT: Positive samples $\boldsymbol{P}$; item set $\boldsymbol{I}$; negative sample rate T; item-category table C; similar sampling weight β

OUTPUT: Training samples Ω

```
$\Omega \leftarrow \boldsymbol{P}$
for all positive sample $(u, i, \text{True}) \in \boldsymbol{P}$ do
    $\boldsymbol{N} \leftarrow \boldsymbol{I} \setminus i$
    $\boldsymbol{S} \leftarrow \boldsymbol{I}_{C(i)} \setminus i$
    for $t \leftarrow 1$ to $T$ do
        $r \leftarrow \text{RandomFloat}(0, 1)$
        if $r < \beta$ then
            $i_t \leftarrow \text{Sample}(\boldsymbol{S})$
        else
            $i_t \leftarrow \text{Sample}(\boldsymbol{N})$
        end if
        $\Omega \leftarrow \Omega + (u, i_t, \text{False})$
    end for
end for
return $\Omega$
```

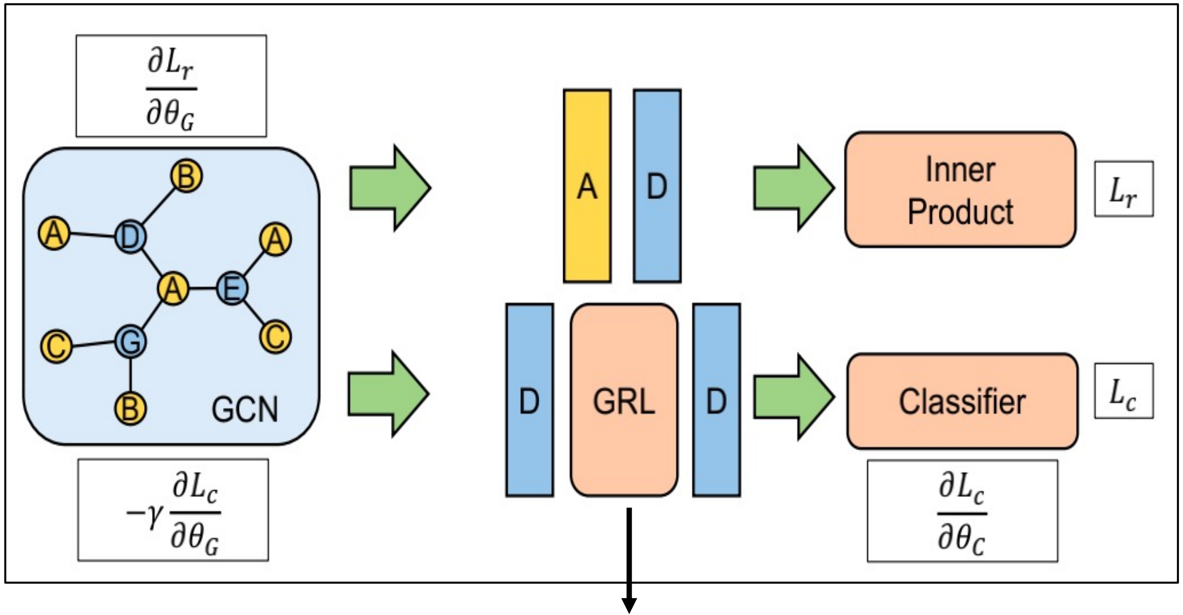
1. Select more negative items from positive categories
2. Introduce β to perform trade-off (sample probability)

Methodology: Our DGCN Model

- Adversarial learning
  - Capture only item-level preference
  - Remove category-level preference

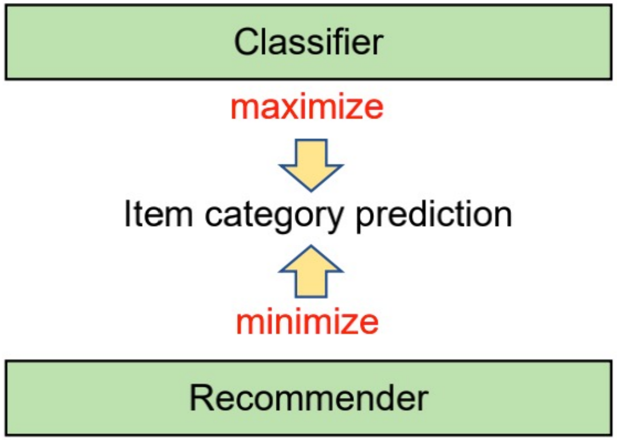
Remove category information from item embeddings!

We can not predict item category from the item embedding!



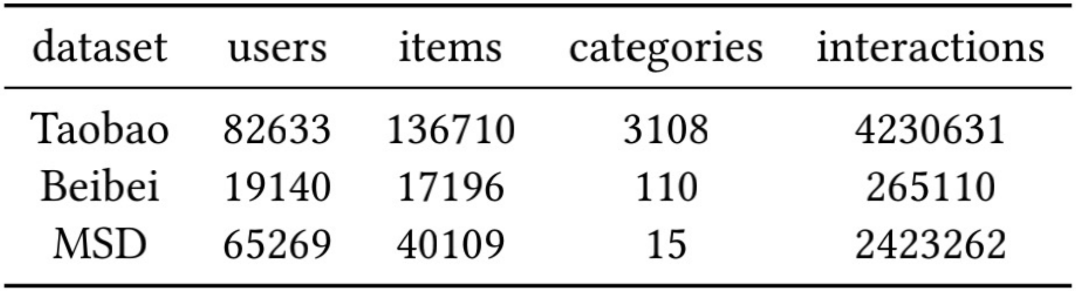
$\frac{\partial L_r}{\partial \theta_G}$
B
A
D
A
A
E
C
G
C
B
GCN
$-\gamma \frac{\partial L_c}{\partial \theta_G}$
A
D
Inner Product
L_r
D
GRL
D
Classifier
L_c
$\frac{\partial L_c}{\partial \theta_C}$

GRL: gradient reversal layer



Experiments

- Datasets
  - Taobao
  - Beibei
  - MSD



dataset	users	items	categories	interactions
Taobao	82633	136710	3108	4230631
Beibei	19140	17196	110	265110
MSD	65269	40109	15	2423262

- Baselines
  - MMR[1][2]
  - DUM[3]
  - PMF + α + β[4]
  - DPP[5]
- Metrics
  - Accuracy: recall, hit ratio
  - Diversity: coverage, entropy, gini index (lower is better)

coverage: #recommended categories
entropy & gini index: equality/fairness of different categories

[1] Carbonell, J., & Goldstein, J. (1998, August). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 335-336).

[2] Ziegler, C. N., McNee, S. M., Konstan, J. A., & Lausen, G. (2005, May). Improving recommendation lists through topic diversification. In Proceedings of the 14th international conference on World Wide Web (pp. 22-32).

[3] Ashkan, A., Kveton, B., Berkovsky, S., & Wen, Z. (2015, January). Optimal Greedy Diversity for Recommendation. In IJCAI (Vol. 15, pp. 1742-1748).

[4] Sha, C., Wu, X., & Niu, J. (2016, January). A Framework for Recommending Relevant and Diverse Items. In IJCAI (Vol. 16, pp. 3868-3874).

[5] Chen, L., Zhang, G., & Zhou, H. (2017). Fast greedy map inference for determinantal point process to improve recommendation diversity. arXiv preprint arXiv:1709.05135.

Experiments

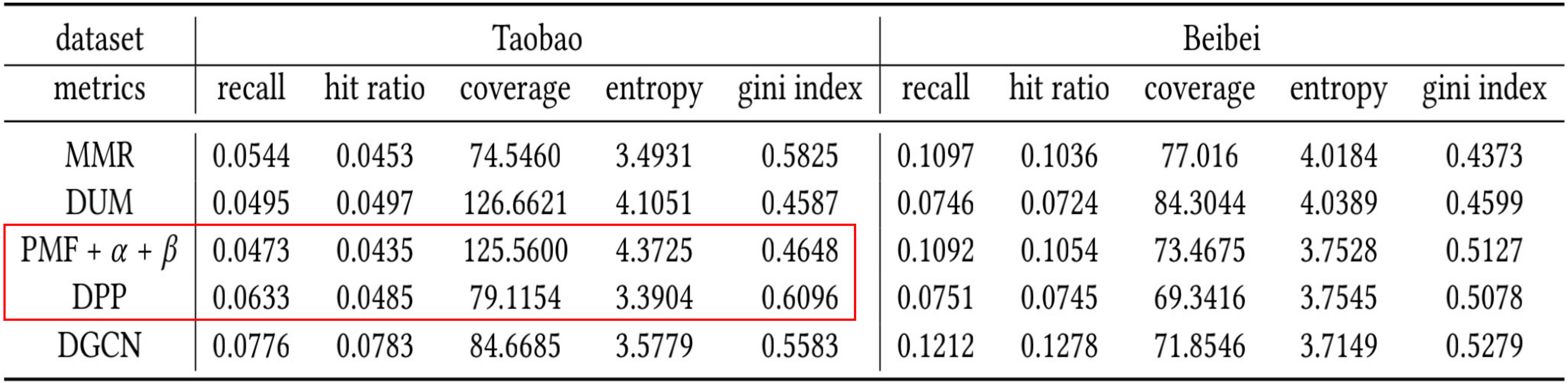
- **RQ1:** How does the proposed method perform compared with other diversified recommendation algorithms?

- **RQ2:** What is the effect of each proposed component in DGCN?

- **RQ3:** How to perform trade-off between accuracy and diversity using DGCN?

Experiments

- Overall Comparison

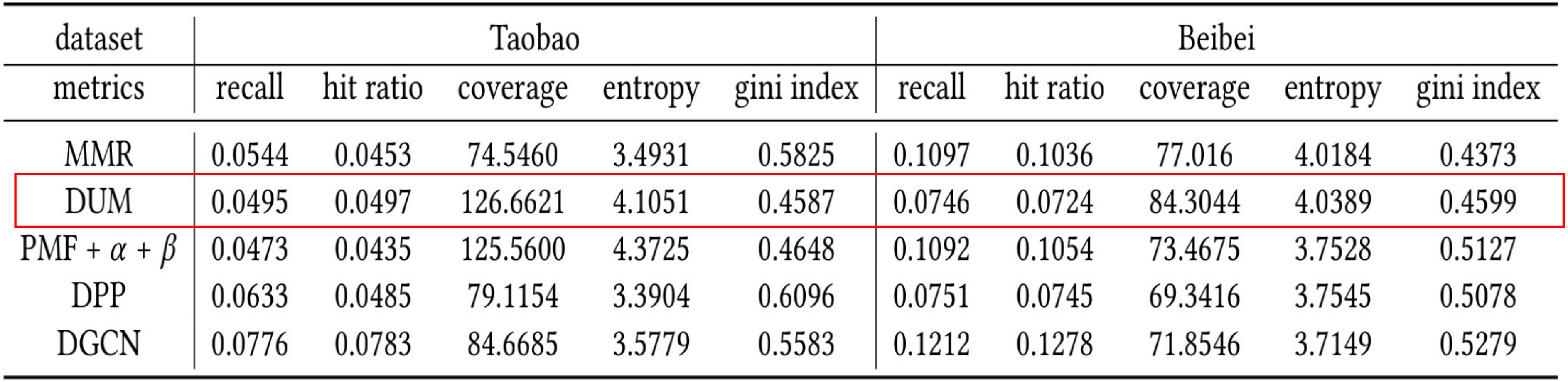


dataset	Taobao					Beibei				
metrics	recall	hit ratio	coverage	entropy	gini index	recall	hit ratio	coverage	entropy	gini index
MMR	0.0544	0.0453	74.5460	3.4931	0.5825	0.1097	0.1036	77.016	4.0184	0.4373
DUM	0.0495	0.0497	126.6621	4.1051	0.4587	0.0746	0.0724	84.3044	4.0389	0.4599
PMF + α + β	0.0473	0.0435	125.5600	4.3725	0.4648	0.1092	0.1054	73.4675	3.7528	0.5127
DPP	0.0633	0.0485	79.1154	3.3904	0.6096	0.0751	0.0745	69.3416	3.7545	0.5078
DGCN	0.0776	0.0783	84.6685	3.5779	0.5583	0.1212	0.1278	71.8546	3.7149	0.5279

- The accuracy-diversity tradeoff exists widely

Experiments

- Overall Comparison



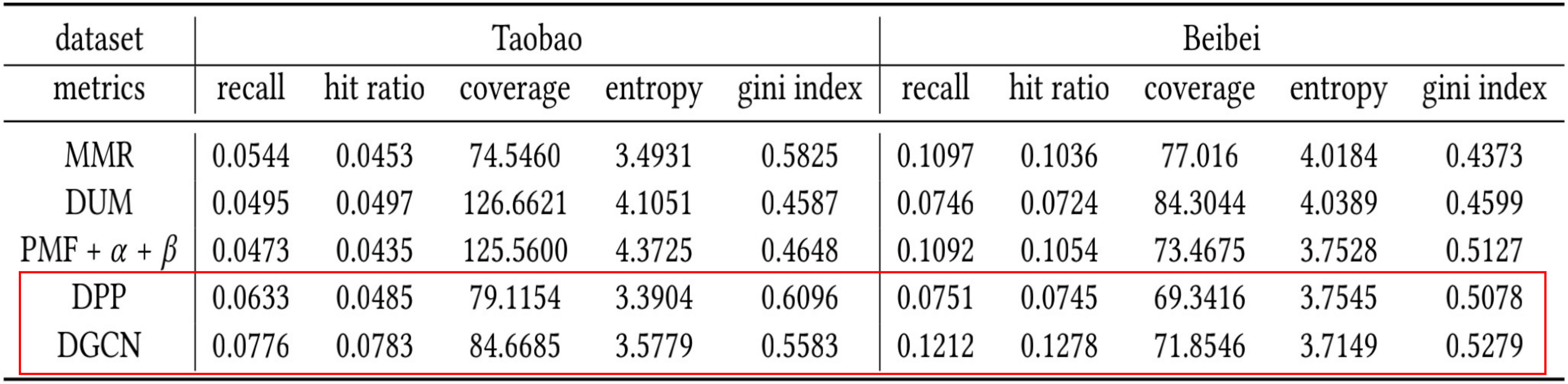
dataset	Taobao					Beibei				
metrics	recall	hit ratio	coverage	entropy	gini index	recall	hit ratio	coverage	entropy	gini index
MMR	0.0544	0.0453	74.5460	3.4931	0.5825	0.1097	0.1036	77.016	4.0184	0.4373
DUM	0.0495	0.0497	126.6621	4.1051	0.4587	0.0746	0.0724	84.3044	4.0389	0.4599
PMF + α + β	0.0473	0.0435	125.5600	4.3725	0.4648	0.1092	0.1054	73.4675	3.7528	0.5127
DPP	0.0633	0.0485	79.1154	3.3904	0.6096	0.0751	0.0745	69.3416	3.7545	0.5078
DGCN	0.0776	0.0783	84.6685	3.5779	0.5583	0.1212	0.1278	71.8546	3.7149	0.5279

- The accuracy-diversity tradeoff exists widely

- It is more difficult to balance the two aspects for greedy algorithms

Experiments

- Overall Comparison



dataset	Taobao					Beibei				
metrics	recall	hit ratio	coverage	entropy	gini index	recall	hit ratio	coverage	entropy	gini index
MMR	0.0544	0.0453	74.5460	3.4931	0.5825	0.1097	0.1036	77.016	4.0184	0.4373
DUM	0.0495	0.0497	126.6621	4.1051	0.4587	0.0746	0.0724	84.3044	4.0389	0.4599
PMF + α + β	0.0473	0.0435	125.5600	4.3725	0.4648	0.1092	0.1054	73.4675	3.7528	0.5127
DPP	0.0633	0.0485	79.1154	3.3904	0.6096	0.0751	0.0745	69.3416	3.7545	0.5078
DGCN	0.0776	0.0783	84.6685	3.5779	0.5583	0.1212	0.1278	71.8546	3.7149	0.5279

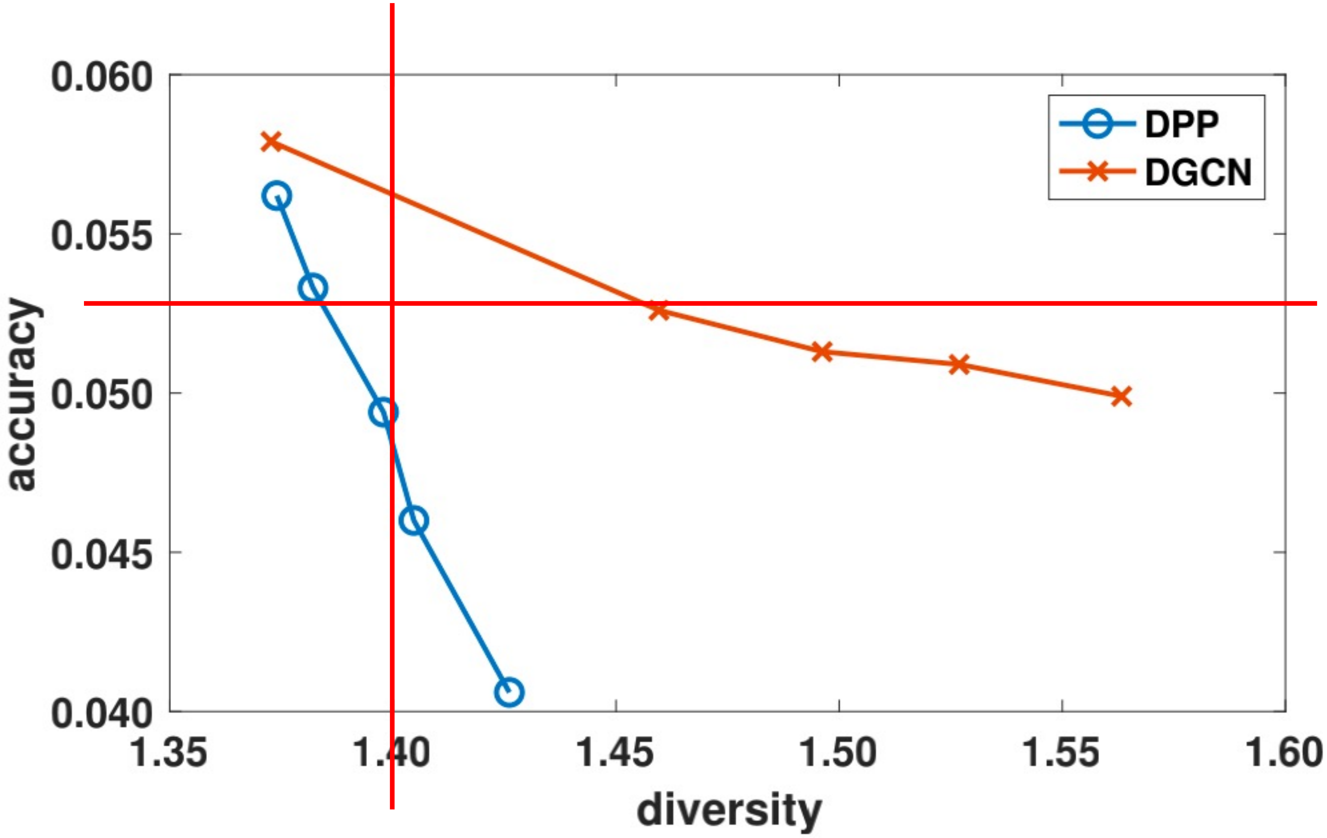
- The accuracy-diversity tradeoff exists widely

- It is more difficult to balance the two aspects for greedy algorithms

- Our proposed DGCN achieves a better overall performance

Experiments

- Overall Comparison



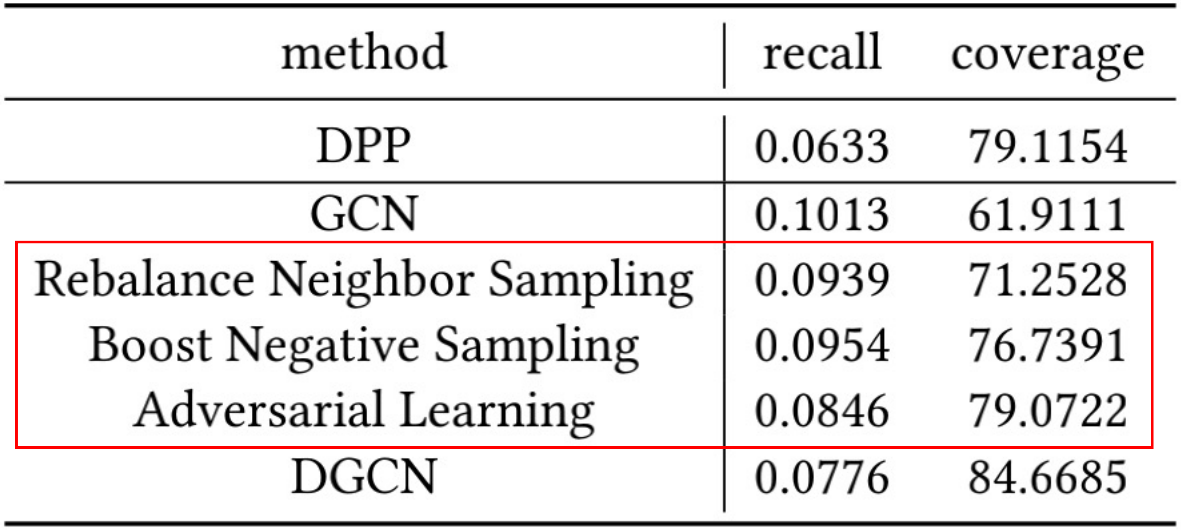
with same diversity, DGCN achieves better accuracy

with same accuracy, DGCN achieves better diversity

- DGCN attains a better overall performance **considering both accuracy and diversity** against state-of-the-art DPP method

Experiments

- Study on DGCN

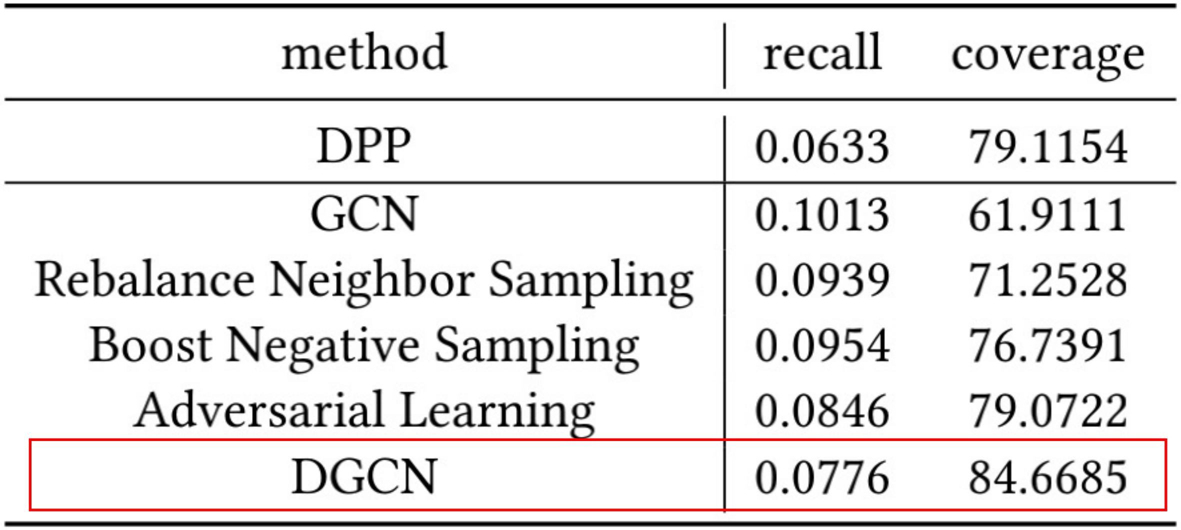


method	recall	coverage
DPP	0.0633	79.1154
GCN	0.1013	61.9111
Rebalance Neighbor Sampling	0.0939	71.2528
Boost Negative Sampling	0.0954	76.7391
Adversarial Learning	0.0846	79.0722
DGCN	0.0776	84.6685

- Each component alone contributes to improve diversity

Experiments

- Study on DGCN

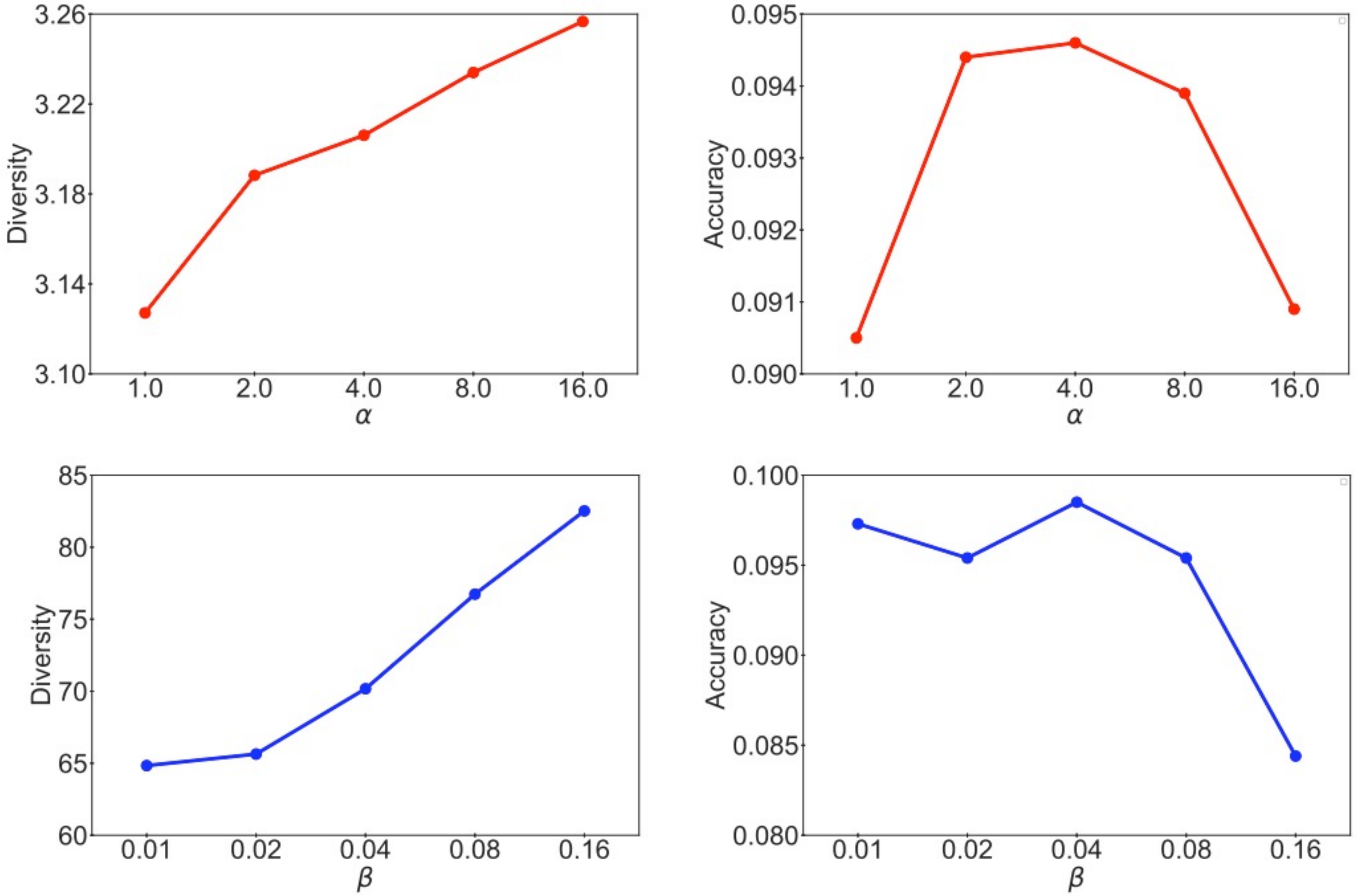


method	recall	coverage
DPP	0.0633	79.1154
GCN	0.1013	61.9111
Rebalance Neighbor Sampling	0.0939	71.2528
Boost Negative Sampling	0.0954	76.7391
Adversarial Learning	0.0846	79.0722
DGCN	0.0776	84.6685

- Each component alone contributes to improve diversity
- Combining the three special designs achieves the most diverse recommendation

Experiments

- Trade-off between accuracy and diversity



- Trade-off is successfully achieved by tuning the two introduced hyper-parameters, α and β

Conclusion and Future Work

- We propose **diversification during matching** based on GCN, which attains better overall performance compared with existing **diversification after matching** approaches. **Better trade-off** between accuracy and diversity can be effectively achieved by the proposed DGCN model.

- **Future work**
- Automate the process of neighbor discovering and negative sampling in DGCN and replace it with a learnable module.

Outline

- Background
- Motivations and Challenges of GNN-based RecSys
- Recent Advances of GNN-based RecSys
  - Part I – Collaborative Filtering, Knowledge Graph-based RecSys
  - Part II – Feature-based Sequential/Bundle/Multi-behavior/Diversified RecSys
- Open Problems and Future Directions

Open discussions

- Go Deeper
  - Requiring more efforts and explorations
- Efficiency on large-scale graphs
  - A concern in industrial deployment
- Hyper-graph
- Dynamic Graph

Thanks!

https://sites.google.com/view/gnn-recsys

Gao. et al. "Graph neural networks for recommender systems: Challenges, methods, and directions." *arXiv preprint arXiv:2109.12843* (2021).