# Graph Neural Network for Recommendations

**Wenqi Fan**

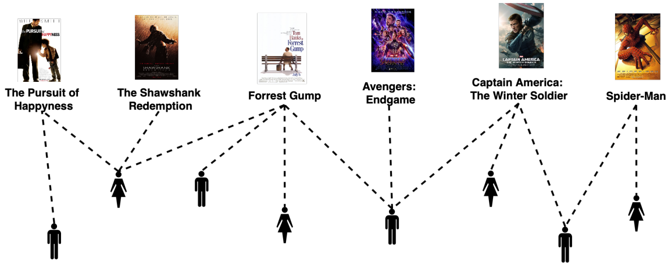
The Hong Kong Polytechnic University

https://wenqifan03.github.io, wenqifan@polyu.edu.hk

Tutorial website: https://deeprs-tutorial.github.io

Data Science and Engineering Lab

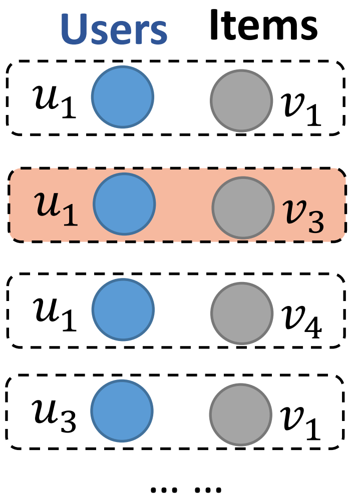
# A General Paradigm




items

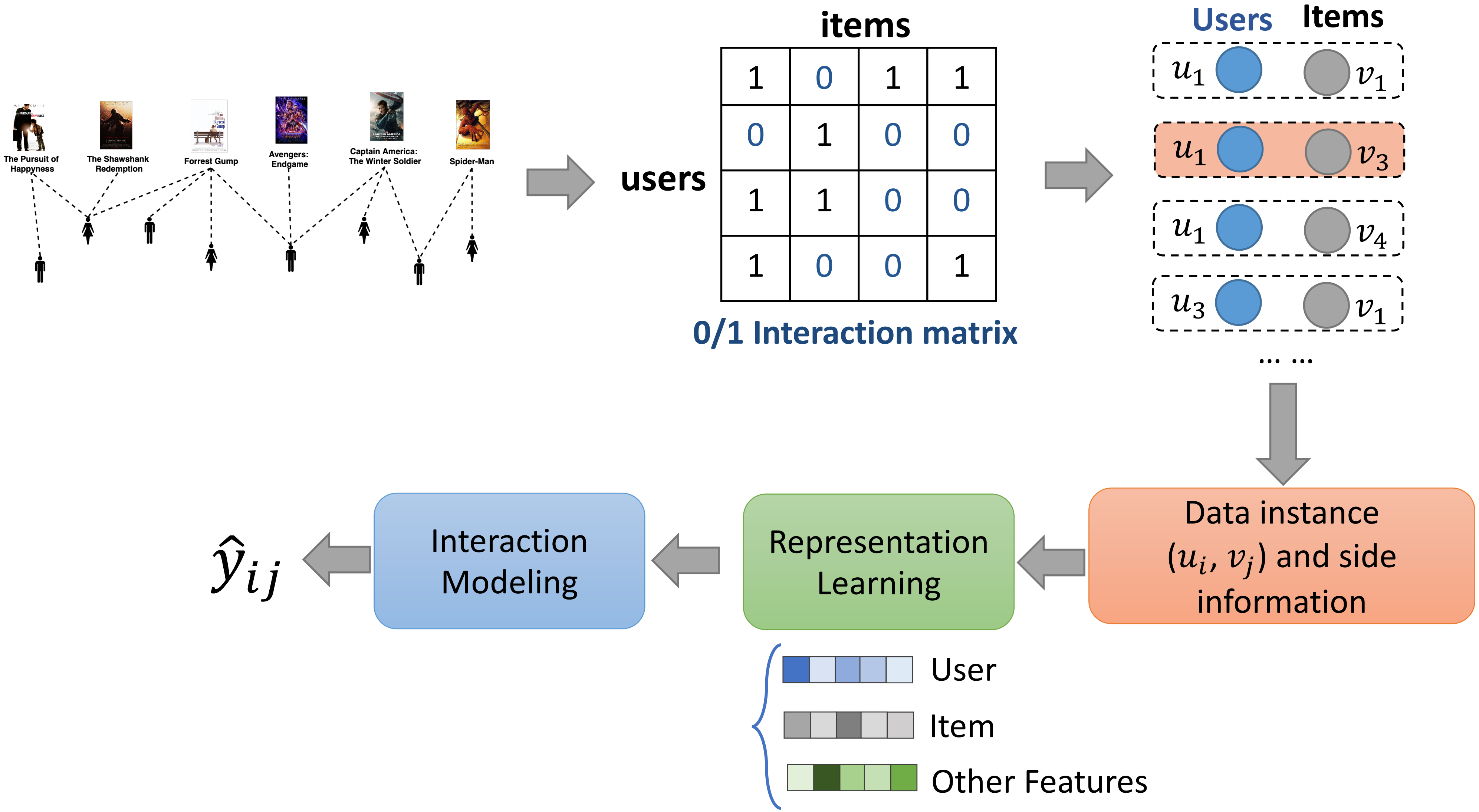
| users | | | | |
|---|---|---|---|---|
| | 1 | 0 | 1 | 1 |
| | 0 | 1 | 0 | 0 |
| | 1 | 1 | 0 | 0 |
| | 1 | 0 | 0 | 1 |

**0/1 Interaction matrix**

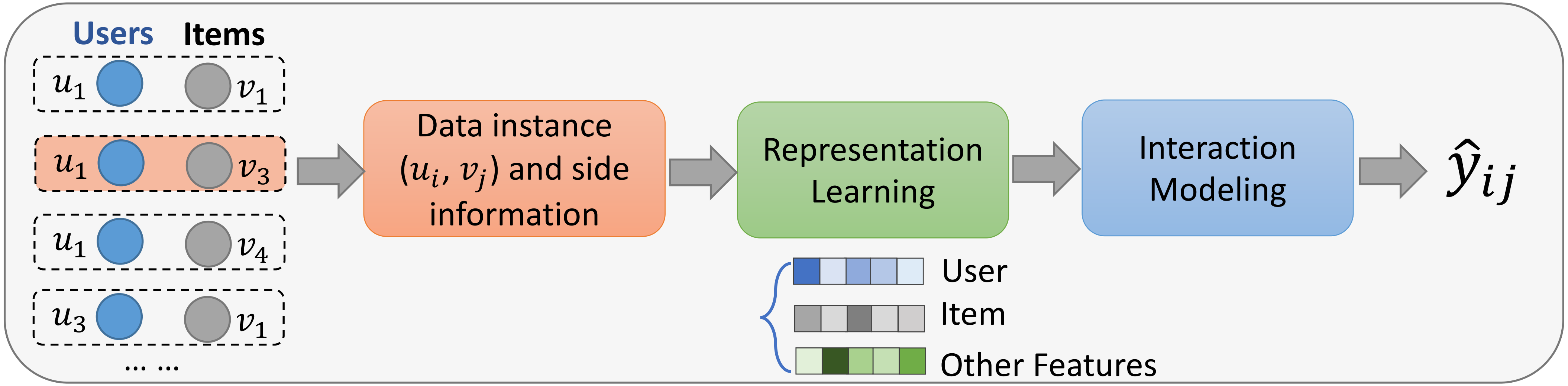



Data instance
$(u_i, v_j)$ and side
information

# A General Paradigm



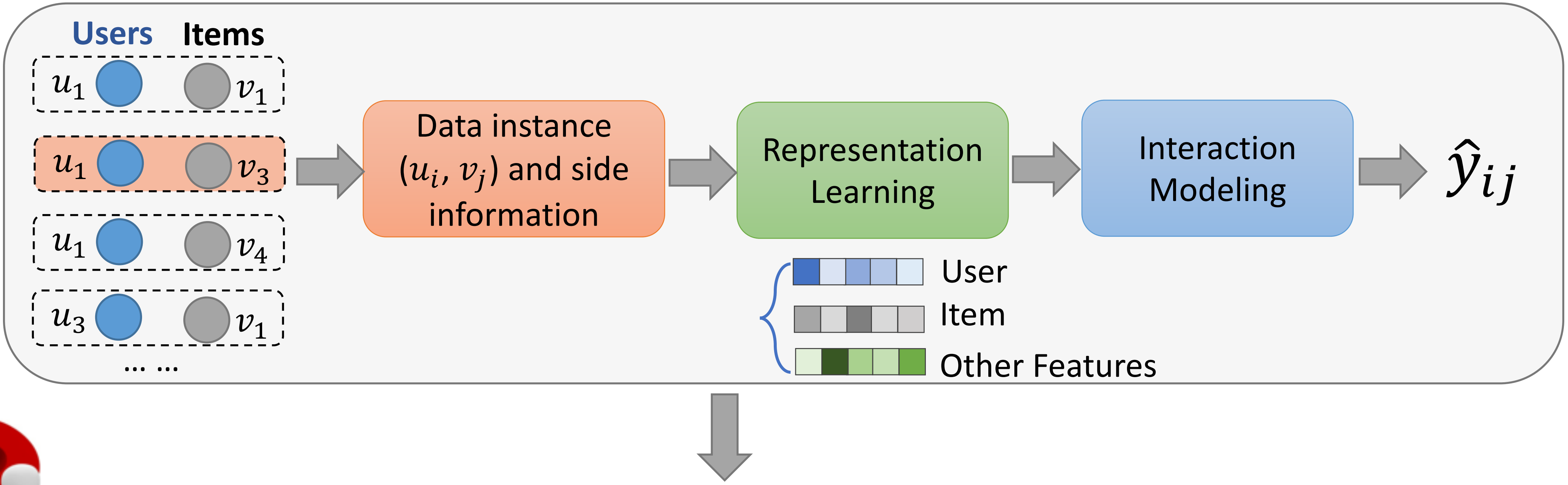
# A General Paradigm




**Information Isolated Island Issue**

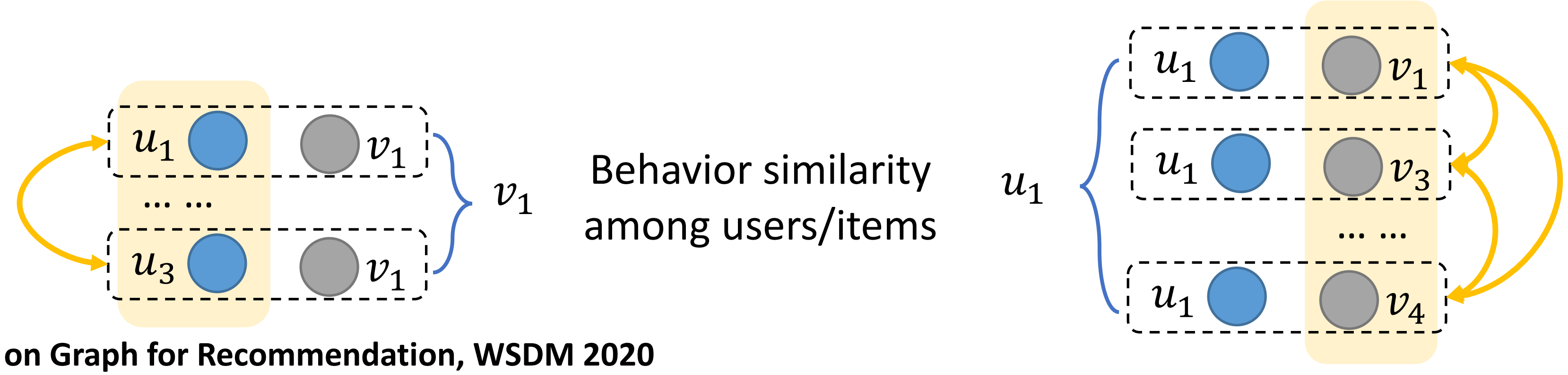
ignore implicit/explicit relationships among instances (**High-order Connectivity** )

# A General Paradigm




**Information Isolated Island Issue**

ignore implicit/explicit relationships among instances (**High-order Connectivity** )



# Data as Graphs

**Most of the data in RS has essentially a graph structure**

- E-commerce, Content Sharing, Social Networking ...

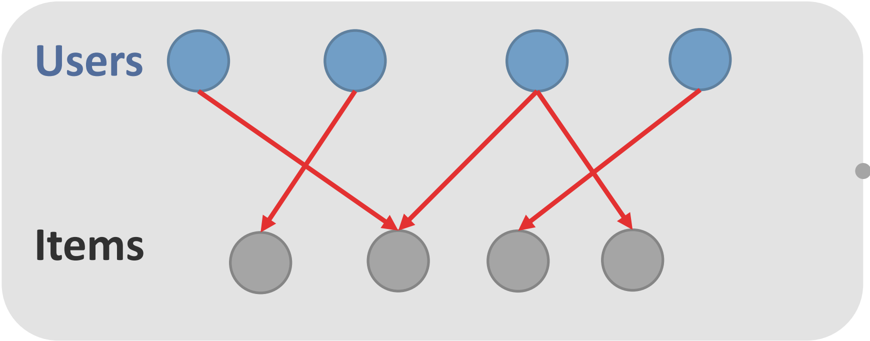
**The world is more closely connected than you might think!**

# Data as Graphs

**Most of the data in RS has essentially a graph structure**

- E-commerce, Content Sharing, Social Networking ...

**The world is more closely connected than you might think!**




User-Item Interactions
(purchase, click, add-to-cart, etc.)

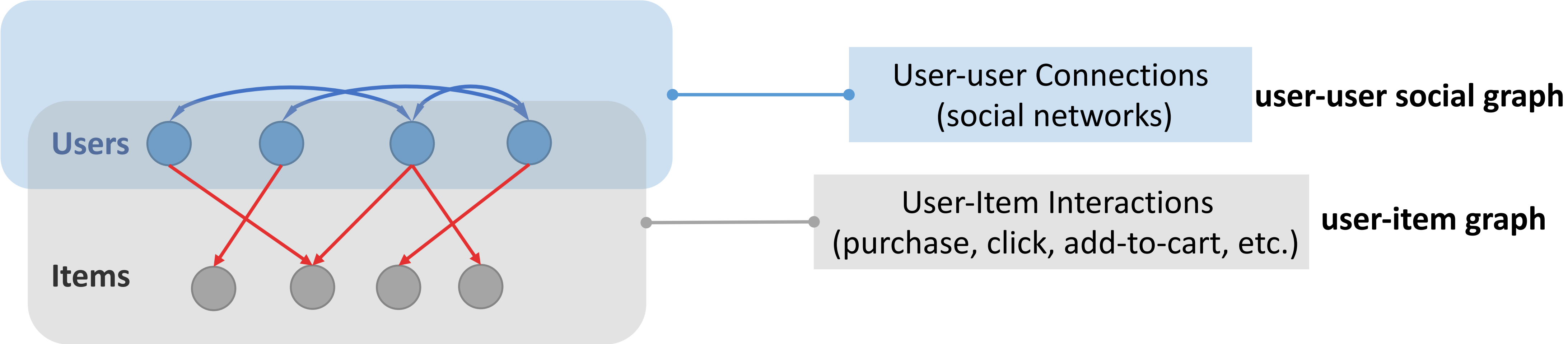
**user-item graph**

# Data as Graphs

**Most of the data in RS has essentially a graph structure**

- E-commerce, Content Sharing, Social Networking ...

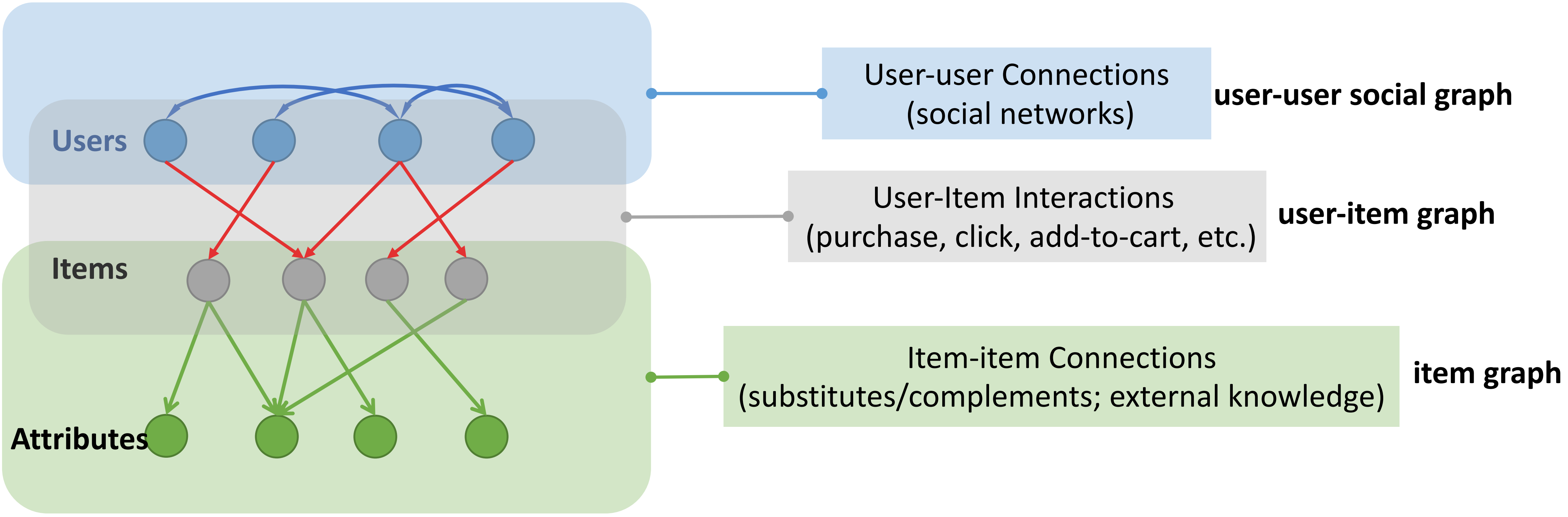
**The world is more closely connected than you might think!**



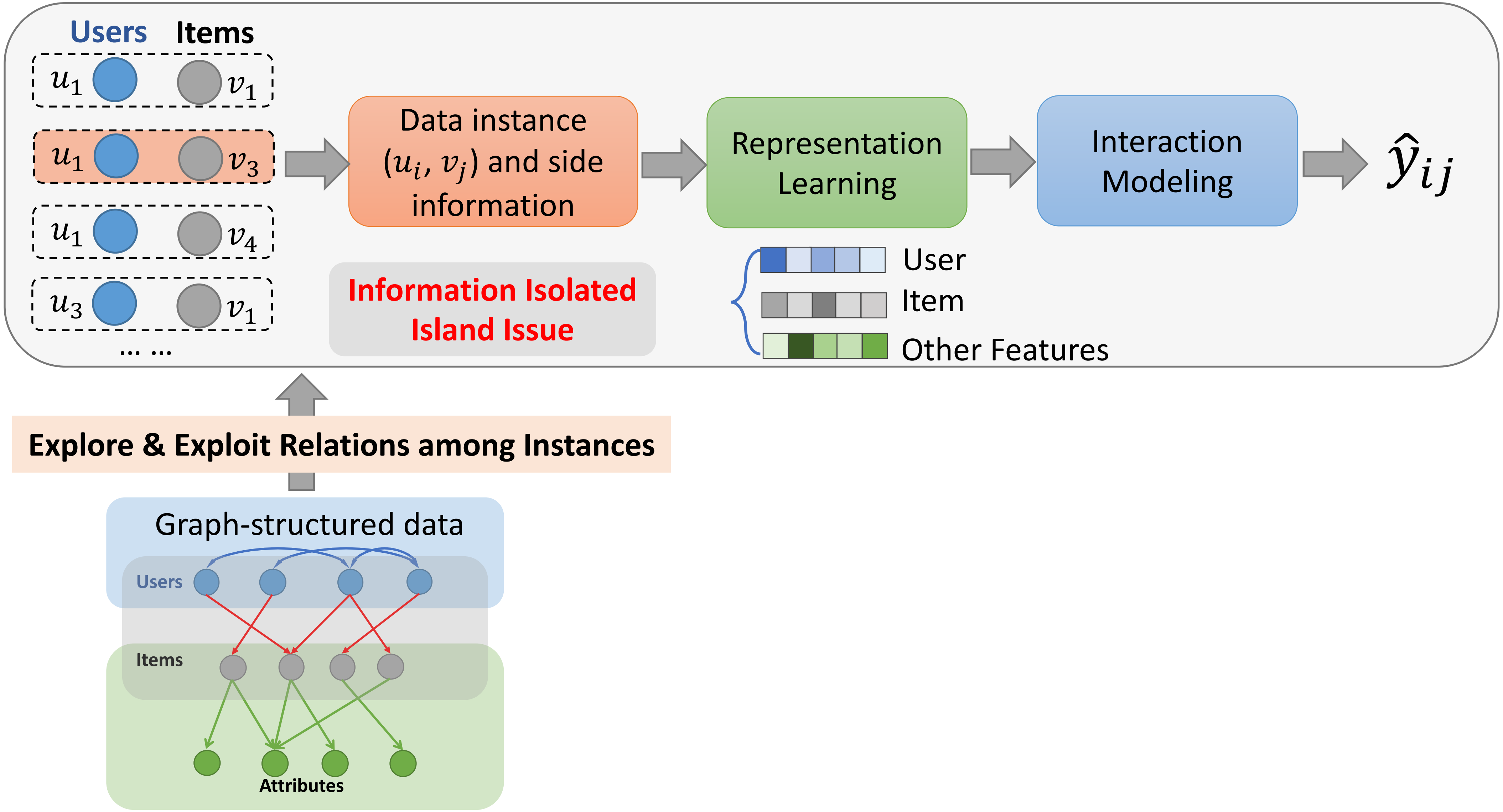
# Data as Graphs

**Most of the data in RS has essentially a graph structure**
- E-commerce, Content Sharing, Social Networking ...

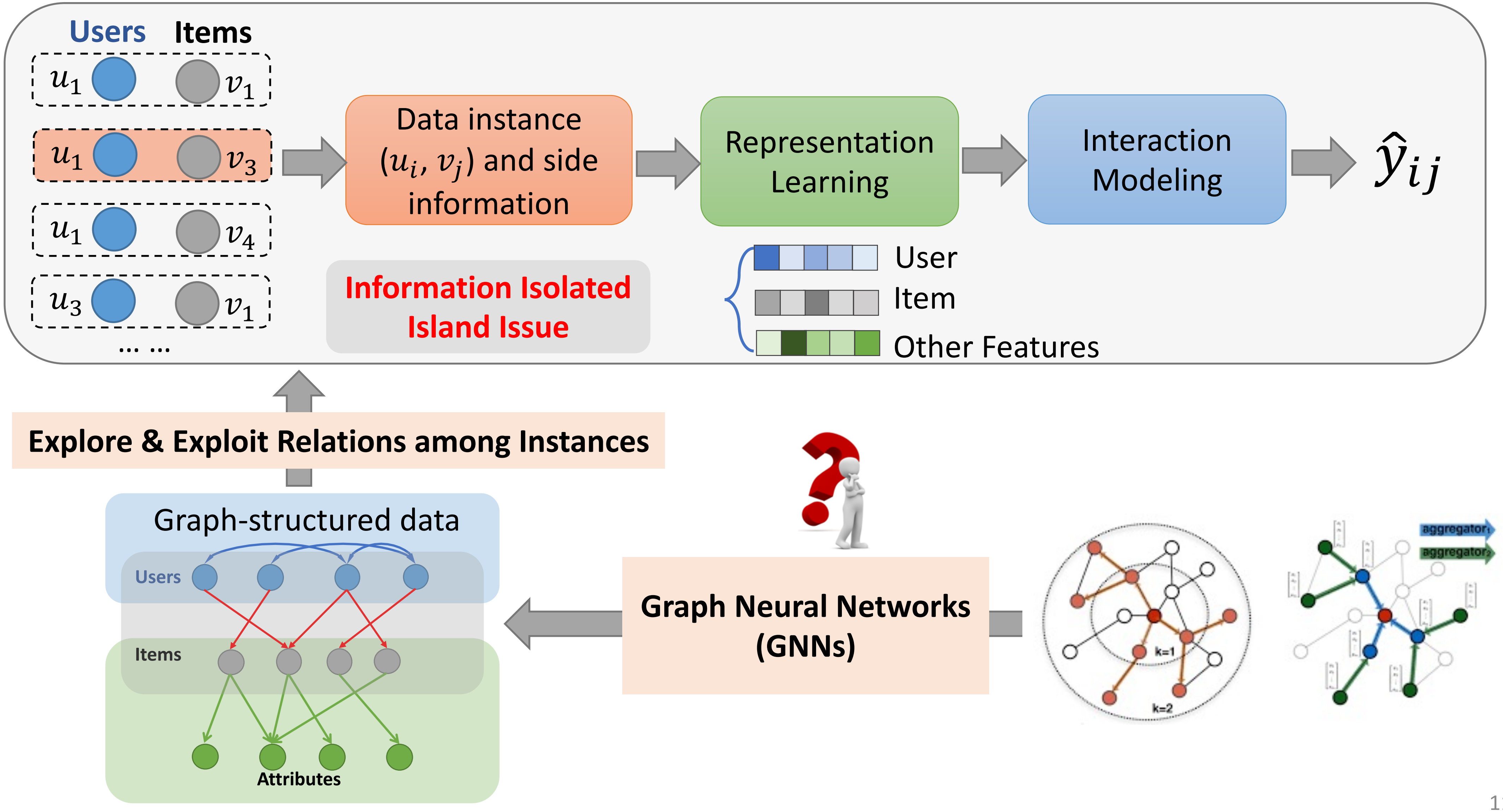
**The world is more closely connected than you might think!**



# How to solve such issue?

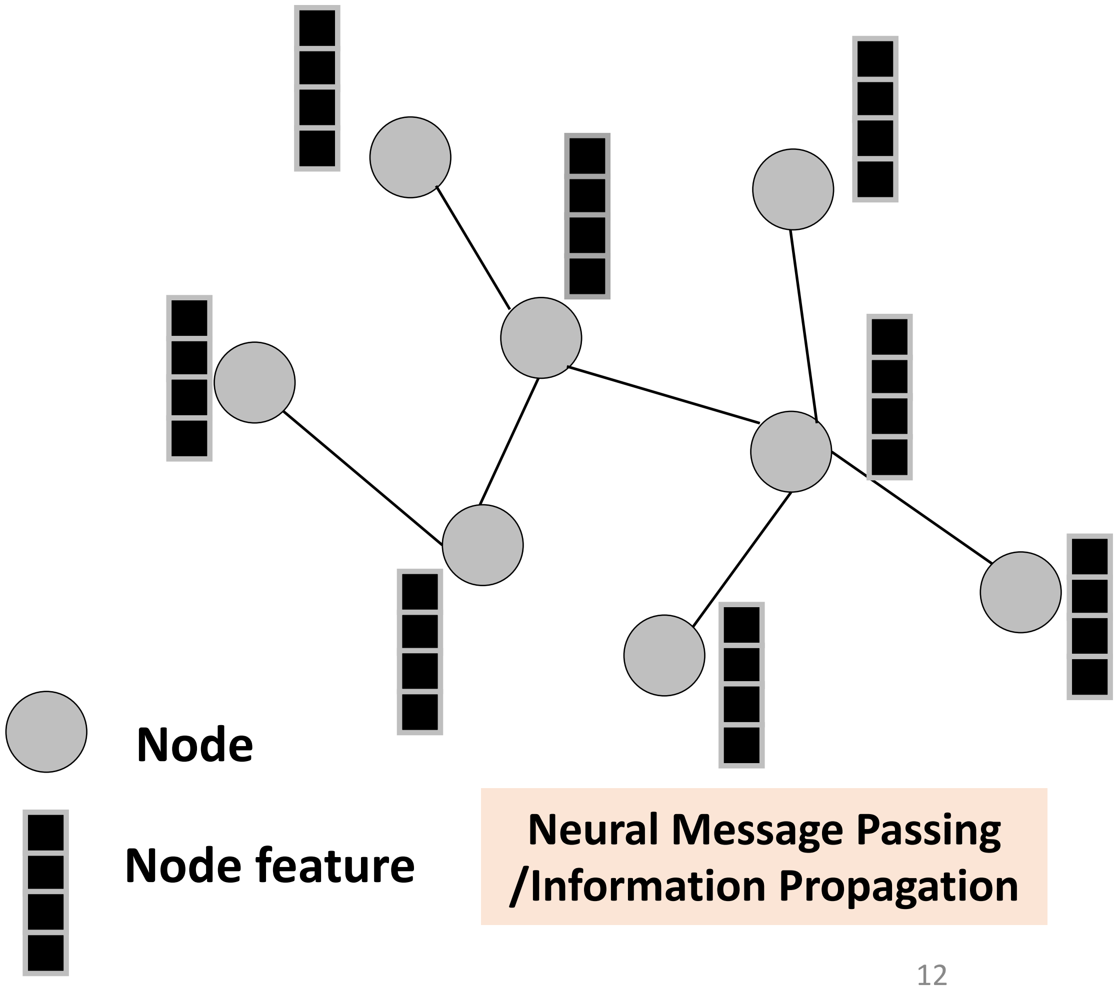


# How to solve such issue?



# Graph Neural Networks (GNNs)

**Key idea**: Generate node embeddings via using neural networks to aggregate information from local neighborhoods.

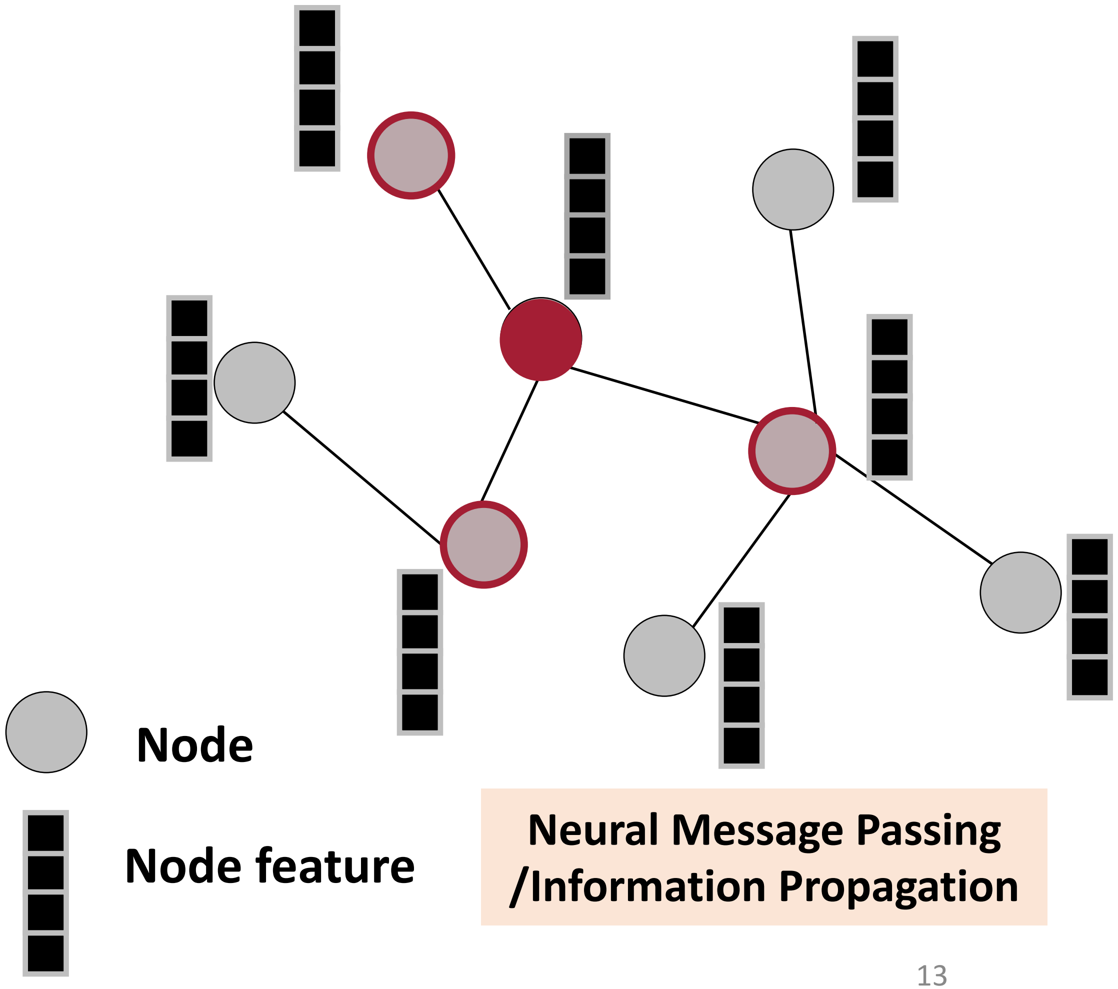



Inductive Representation Learning on Large Graphs, NeuIPS, 2017.

# Graph Neural Networks (GNNs)

**Key idea**: Generate node embeddings via using neural networks to aggregate information from local neighborhoods.

1. Model a local structural information (neighborhood) of a node;



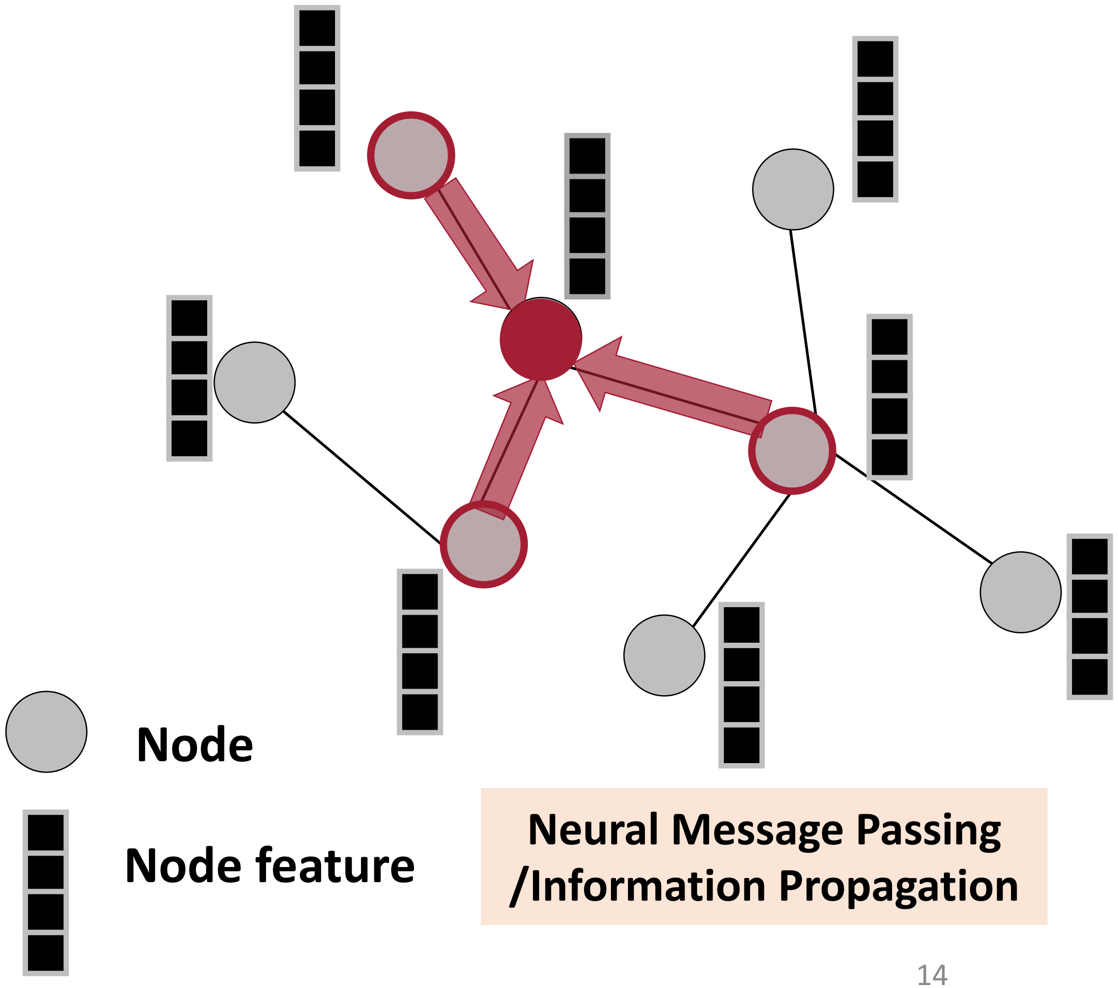

Inductive Representation Learning on Large Graphs, NeuIPS, 2017.

# Graph Neural Networks (GNNs)

**Key idea**: Generate node embeddings via using neural networks to aggregate information from local neighborhoods.

1. Model a local structural information (neighborhood) of a node;
2. Aggregation operation;
3. Representation update.

GNNs can naturally integrate node feature and the topological structure for graph-structured data.




Inductive Representation Learning on Large Graphs, NeuIPS, 2017.

# Graph Neural Networks (GNNs)

**Basic approach:** Average neighbor messages and apply a neural network.

$$\mathbf{h}_v^0 = \mathbf{x}_v$$

Initial 0-th layer embeddings are equal to node $v$'s features

$$\mathbf{h}_v^k = \sigma\left(\mathbf{W}_1^k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{\sqrt{|N(u)|}} + \mathbf{W}_2^k \mathbf{h}_v^{k-1}\right)$$

k-th layer embedding of node $v$

$$\mathbf{z}_v = \mathbf{h}_v^k$$

Embedding after k layers of neighborhood aggregation.



Semi-supervised Classification with Graph Convolutional Network, ICLR, 2017.

# Graph Neural Networks (GNNs)

**Basic approach:** Average neighbor messages and apply a neural network.

$$\mathbf{h}_v^0 = \mathbf{x}_v$$

Initial 0-th layer embeddings are equal to node $v$'s features

Non-linearity (e.g., ReLU or tanh)

trainable matrices (i.e., what we learn)

Previous layer embedding of node $v$

$$\mathbf{h}_v^k = \sigma\left(\mathbf{W}_1^k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{\sqrt{|N(u)|}} + \mathbf{W}_2^k \mathbf{h}_v^{k-1}\right)$$

k-th layer embedding of node $v$

Average of neighbor's previous layer embeddings

$$\mathbf{z}_v = \mathbf{h}_v^k$$

Embedding after k layers of neighborhood aggregation.



Semi-supervised Classification with Graph Convolutional Network, ICLR, 2017.

# Graph Neural Network (GNN)

- **Simple neighborhood aggregation:**

$$\mathbf{h}_v^k = \sigma\left(\mathbf{W}_1^k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{\sqrt{|N(u)|}} + \mathbf{W}_2^k \mathbf{h}_v^{k-1}\right)$$

- **GraphSAGE:**

- **GAT:**

Inductive Representation Learning on Large Graphs, NeuIPS, 2017.
Graph Attention Networks, ICLR, 2018

# Graph Neural Network (GNN)

- **Simple neighborhood aggregation:**

$$\mathbf{h}_v^k = \sigma\left(\mathbf{W}_1^k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{\sqrt{|N(u)|}} + \mathbf{W}_2^k \mathbf{h}_v^{k-1}\right)$$

- **GraphSAGE:**

$$\mathbf{h}_v^k = \sigma\left(\left[\mathbf{W}_1^k \cdot \mathrm{AGG}\left(\{\mathbf{h}_u^{k-1}, \forall_u \in N(u)\}\right), \mathbf{W}_2^k \cdot \mathbf{h}_v^k\right]\right)$$

**Generalized Aggregation: mean, pooling, LSTM**

- **GAT:**

Inductive Representation Learning on Large Graphs, NeuIPS, 2017.
Graph Attention Networks, ICLR, 2018

# Graph Neural Network (GNN)

- **Simple neighborhood aggregation:**

$$\mathbf{h}_v^k = \sigma\left(\mathbf{W}_1^k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{\sqrt{|N(u)|}} + \mathbf{W}_2^k \mathbf{h}_v^{k-1}\right)$$

- **GraphSAGE:**

$$\mathbf{h}_v^k = \sigma\left([\mathbf{W}_1^k \cdot \mathrm{AGG}\left(\{\mathbf{h}_u^{k-1}, \forall_u \in N(u)\}\right), \mathbf{W}_2^k \cdot \mathbf{h}_v^k]\right)$$

Generalized Aggregation: mean, pooling, LSTM

- **GAT:**

$$\mathbf{h}_v^k = \sigma\left(\sum_{u \in N(v)} \alpha_{v,u} \mathbf{W}^k \, \mathbf{h}_u^{k-1}\right)$$

Learned attention weights

Inductive Representation Learning on Large Graphs, NeuIPS, 2017.
Graph Attention Networks, ICLR, 2018

# Book: Deep Learning on Graphs

https://cse.msu.edu/~mayao4/dlg_book/

Yao Ma and Jiliang Tang, MSU

1. Introduction

**Part One: Foundations**
- 2. Foundations of Graphs
- 3. Foundations of Deep Learning

**Part Two: Methods**
- 4. Graph Embedding
- 5. Graph Neural Networks
- 6. Robust Graph Neural Networks
- 7. Scalable Graph Neural Networks
- 8. Graph Neural Networks for Complex Graphs
- 9. Beyond GNNs: More Deep Models for Graphs

**Part Three: Applications**
- 10. Graph Neural Networks in Natural Language Processing
- 11. Graph Neural Networks in Computer Vision
- 12. Graph Neural Networks in Data Mining
- 13. Graph Neural Networks in Bio-Chemistry and Healthcare

**Part Four: Advances**
- 14. Advanced Methods in Graph Neural Networks
- 15. Advanced Applications in Graph Neural Networks

# GNNs based Recommendation

## Collaborative Filtering

- Graph Convolutional Neural Networks for Web-Scale Recommender Systems (KDD'18)
- Graph Convolutional Matrix Completion (KDD'18 Deep Learning Day )
- Neural Graph Collaborative Filtering (SIGIR'19)
- LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation (SIGIR'20)

## Collaborative Filtering with Side Information (Users/Items)

- **Social Recommendation (Users)**
  - Graph Neural Network for Social Recommendation (WWW'19)
  - A Neural Influence Diffusion Model for Social Recommendation (SIGIR'19)
  - A Graph Neural Network Framework for Social Recommendations (TKDE'20)
- **Knowledge-graph-aware Recommendation (Items)**
  - Knowledge Graph Convolutional Networks for Recommender Systems with Label Smoothness Regularization (KDD'19 and WWW'19)
  - KGAT: Knowledge Graph Attention Network for Recommendation (KDD'19)

# GNNs based Recommendation

## Collaborative Filtering

- Graph Convolutional Neural Networks for Web-Scale Recommender Systems (KDD'18)
- Graph Convolutional Matrix Completion (KDD'18 Deep Learning Day )
- Neural Graph Collaborative Filtering (SIGIR'19)
- LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation (SIGIR'20)

## Collaborative Filtering with Side Information (Users/Items)

- **Social Recommendation (Users)**
  - Graph Neural Network for Social Recommendation (WWW'19)
  - A Neural Influence Diffusion Model for Social Recommendation (SIGIR'19)
  - A Graph Neural Network Framework for Social Recommendations (TKDE'20)
- **Knowledge-graph-aware Recommendation (Items)**
  - Knowledge Graph Convolutional Networks for Recommender Systems with Label Smoothness Regularization (KDD'19 and WWW'19)
  - KGAT: Knowledge Graph Attention Network for Recommendation (KDD'19)

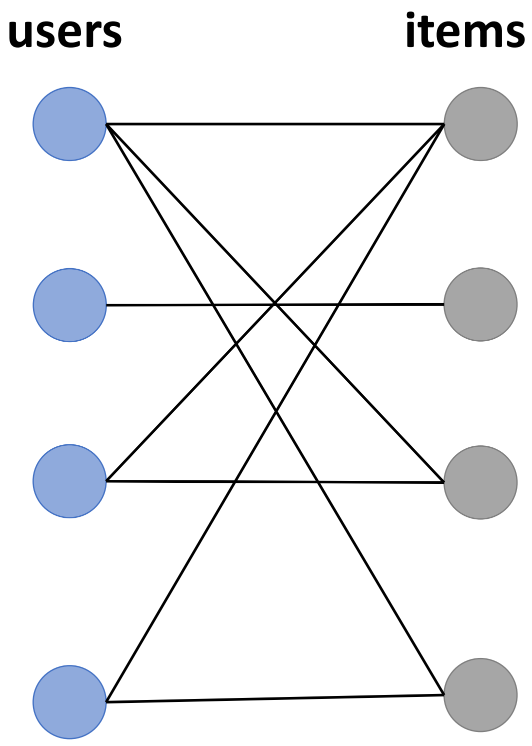
# Interactions as Bipartite Graph

items

users

| | | | |
|---|---|---|---|
| 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 |

**0/1 Interaction matrix**




**Bipartite Graph**

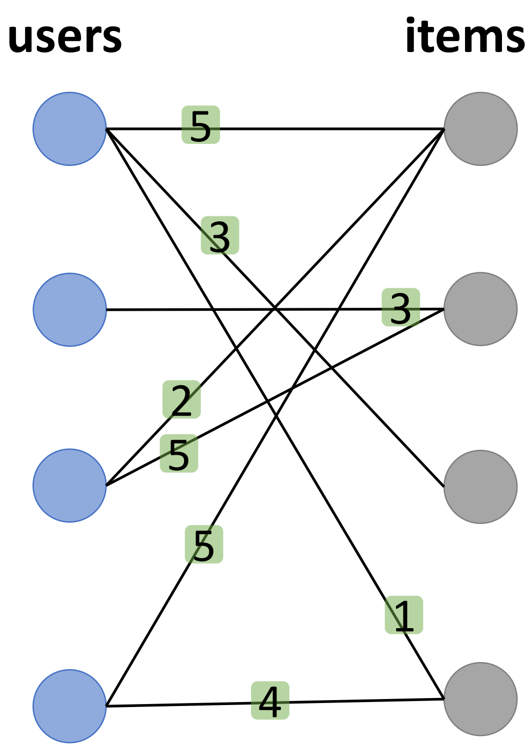
# Interactions as Bipartite Graph

items

users

| | | | |
|---|---|---|---|
| 5 | 0 | 3 | 1 |
| 0 | 3 | 0 | 0 |
| 2 | 5 | 0 | 0 |
| 5 | 0 | 0 | 4 |

**Weighted interaction matrix**

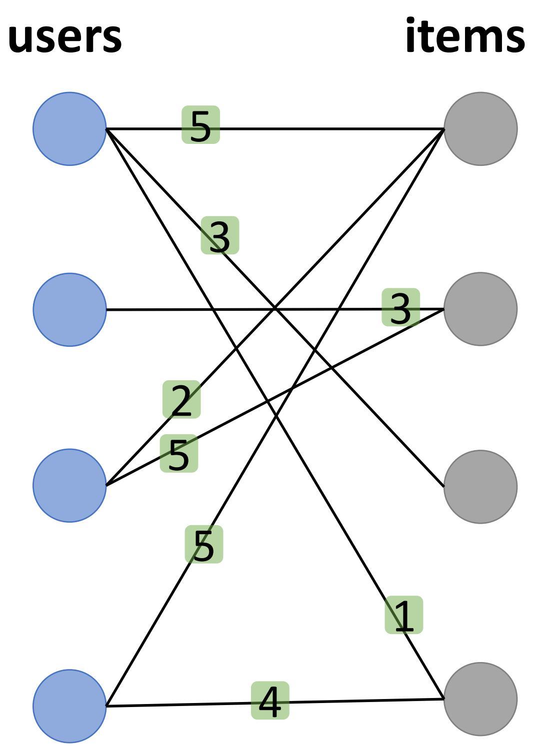



**Bipartite Graph**

# GCMC

## User representation learning

Aggregate for each rating: $\mu_{i,r} = \sum_{j \in \mathcal{N}_{i,r}} \frac{1}{c_{ij}} \boxed{W_r} x_j$




Bipartite Graph

Graph Convolutional Matrix Completion. 2017.

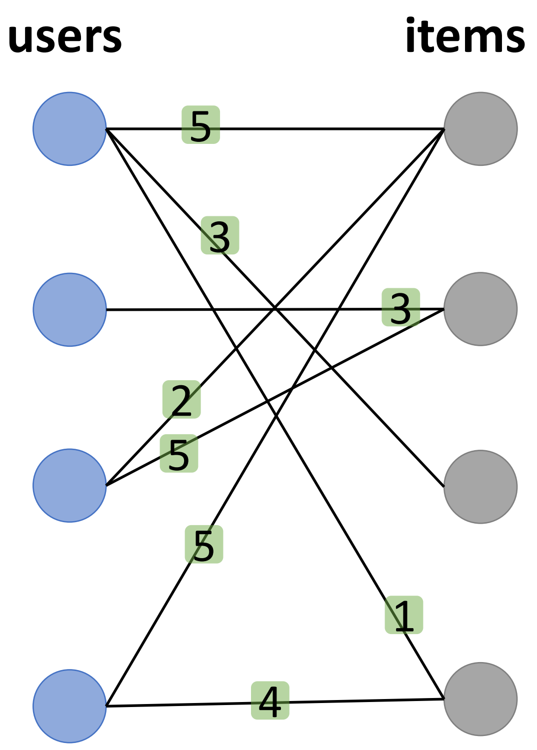
# GCMC

## User representation learning

Aggregate for each rating: $\mu_{i,r} = \sum_{j \in \mathcal{N}_{i,r}} \frac{1}{c_{ij}} \boxed{W_r} x_j$

$$u_i = \mathbf{W} \cdot \sigma(accum(u_{i,1}, \ldots, u_{i,R}))$$

Item representation learning in a similar way

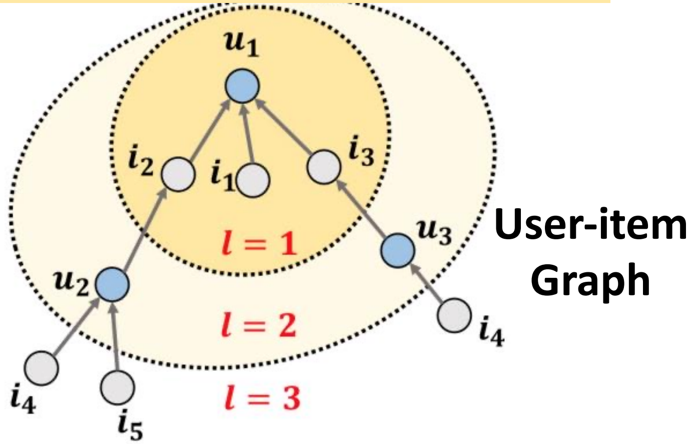



Bipartite Graph

**Graph Convolutional Matrix Completion.** 2017.

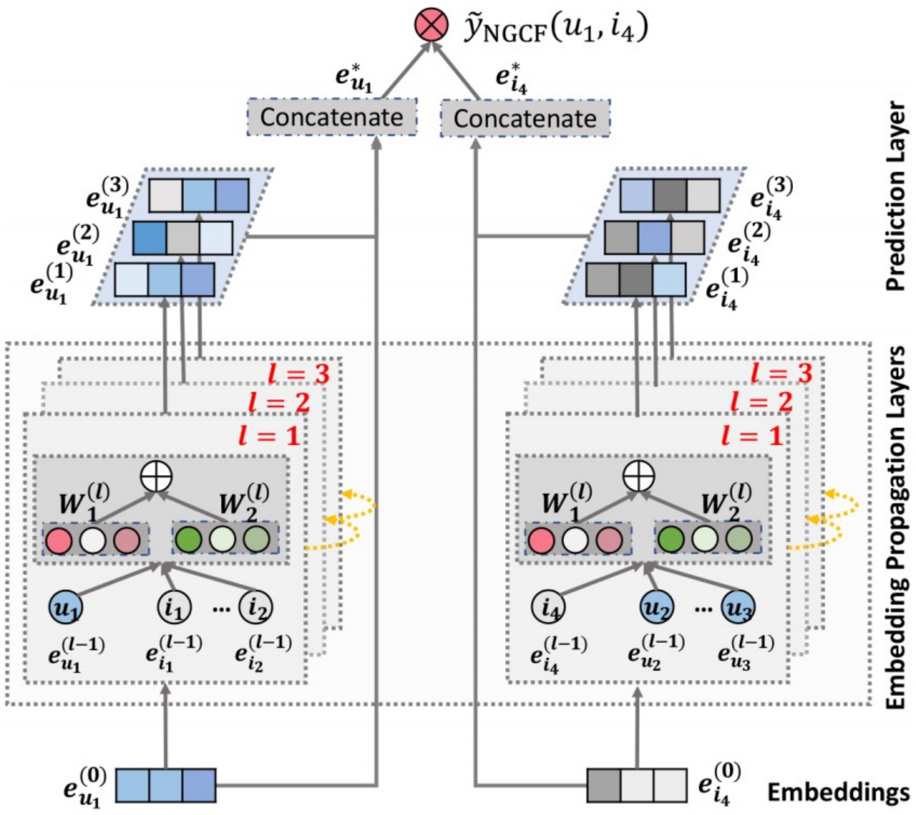
# NGCF

**High-order Connectivity for $u_1$**




**Embedding Propagation, inspired by GNNs**

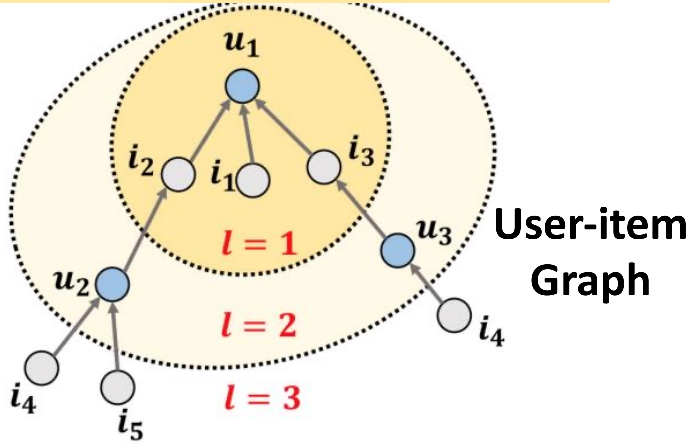
- Propagate embeddings recursively on the user-item graph
- Construct information flows in the embedding space




**Neural Graph Collaborative Filtering.** SIGIR 2019.

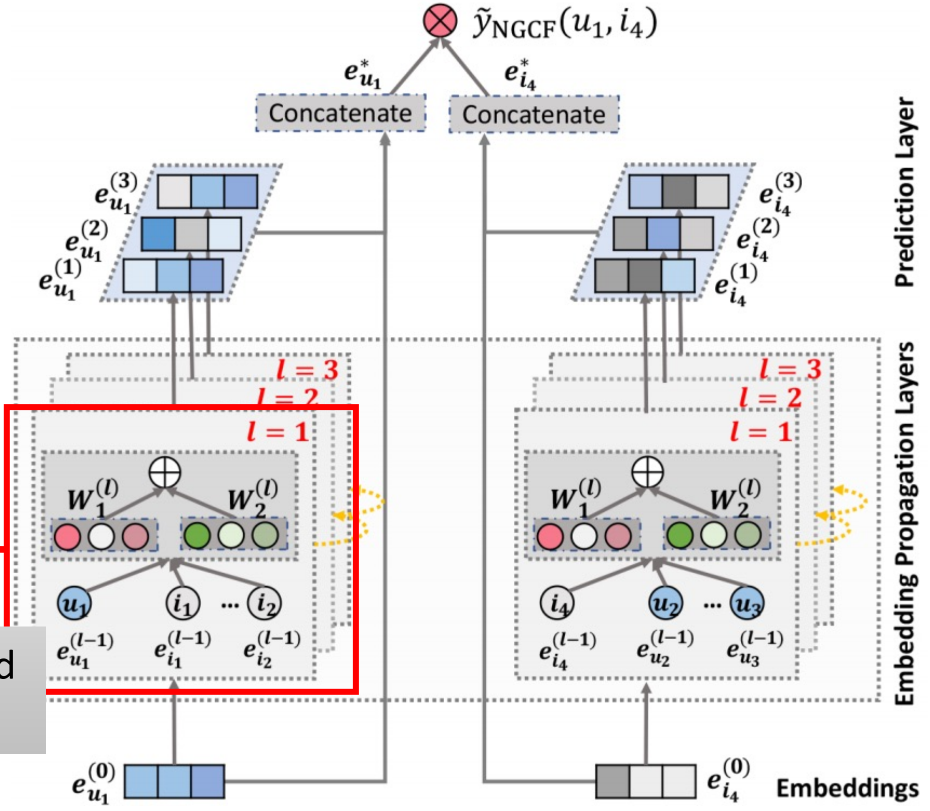
# NGCF

**High-order Connectivity for $u_1$**




**Embedding Propagation, inspired by GNNs**

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




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

$$\begin{cases} \mathbf{m}_{u\leftarrow i}^{(l)} = p_{ui}\Big(\mathbf{W}_1^{(l)}\mathbf{e}_i^{(l-1)} + \mathbf{W}_2^{(l)}(\mathbf{e}_i^{(l-1)} \odot \mathbf{e}_u^{(l-1)})\Big) \\ \mathbf{m}_{u\leftarrow u}^{(l)} = \mathbf{W}_1^{(l)}\mathbf{e}_u^{(l-1)} \end{cases}$$

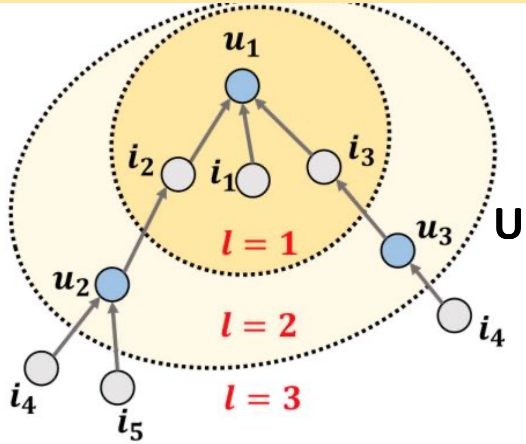
collaborative signal: message passed from interacted items to $u$

Self-connections

**Neural Graph Collaborative Filtering.** SIGIR 2019.

# NGCF

**High-order Connectivity for $u_1$**




**User-item Graph**

**Embedding Propagation, inspired by GNNs**

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

Different layers

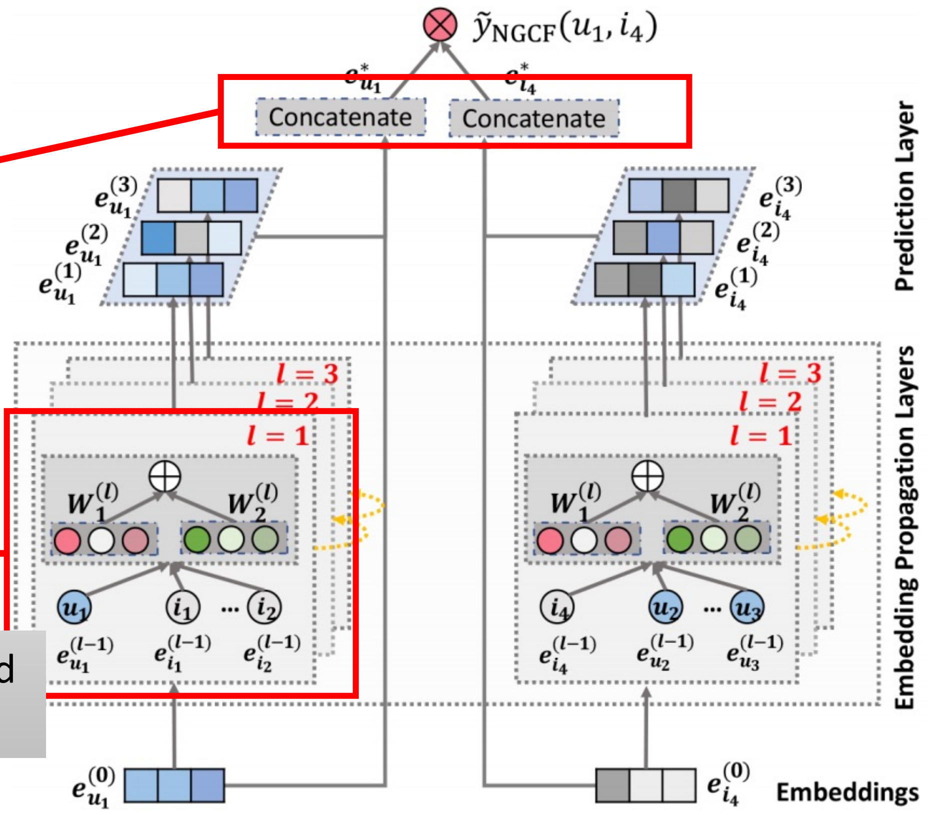
$$\mathbf{e}_u^* = \mathbf{e}_u^{(0)} \| \cdots \| \mathbf{e}_u^{(L)}, \quad \mathbf{e}_i^* = \mathbf{e}_i^{(0)} \| \cdots \| \mathbf{e}_i^{(L)},$$

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

$$\begin{cases} \mathbf{m}_{u\leftarrow i}^{(l)} = p_{ui}\Big(\mathbf{W}_1^{(l)}\mathbf{e}_i^{(l-1)} + \mathbf{W}_2^{(l)}(\mathbf{e}_i^{(l-1)} \odot \mathbf{e}_u^{(l-1)})\Big) \\ \mathbf{m}_{u\leftarrow u}^{(l)} = \mathbf{W}_1^{(l)}\mathbf{e}_u^{(l-1)} \end{cases}$$

collaborative signal: message passed from interacted items to $u$

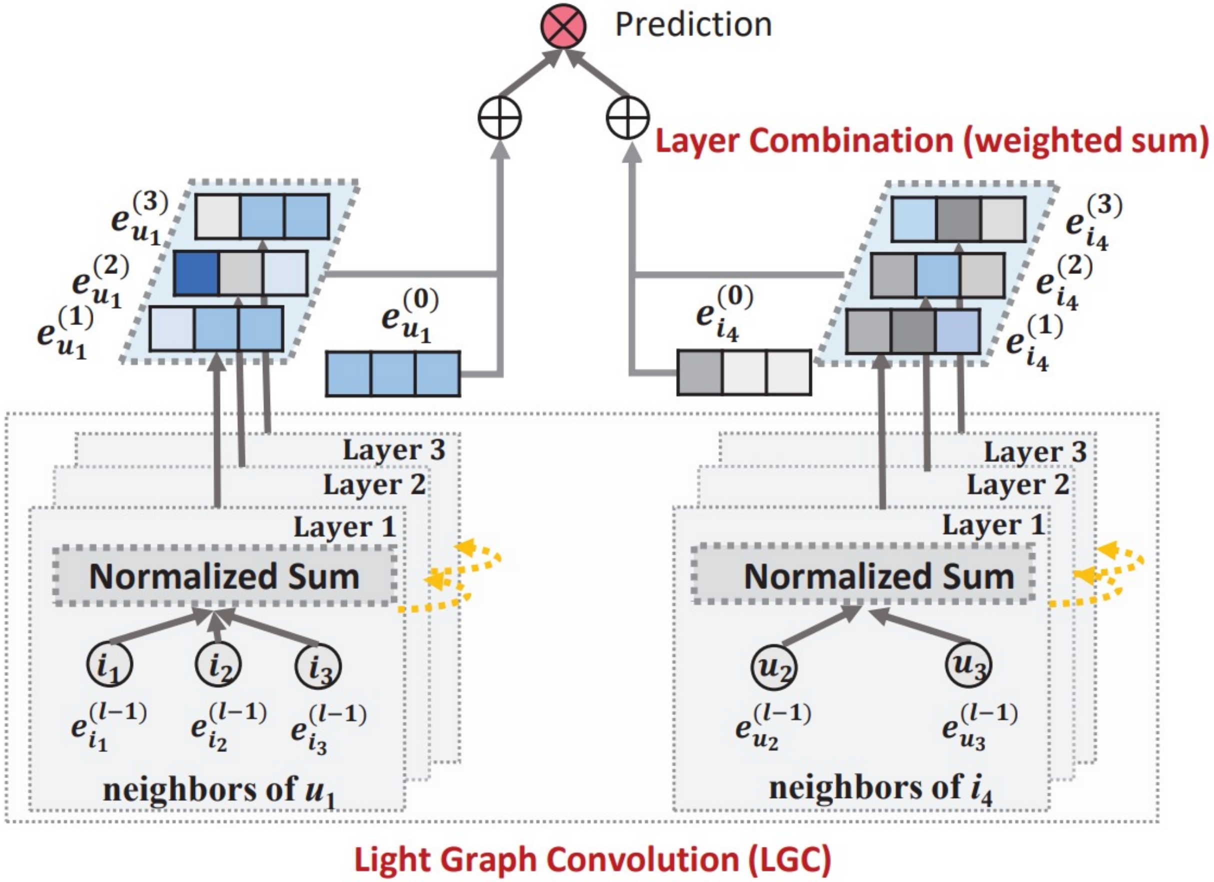
Self-connections




**Neural Graph Collaborative Filtering.** SIGIR 2019.

# LightGCN

## Simplifying GCN for recommendation




**discard feature transformation and nonlinear activation**

**LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation.** SIGIR 2020.

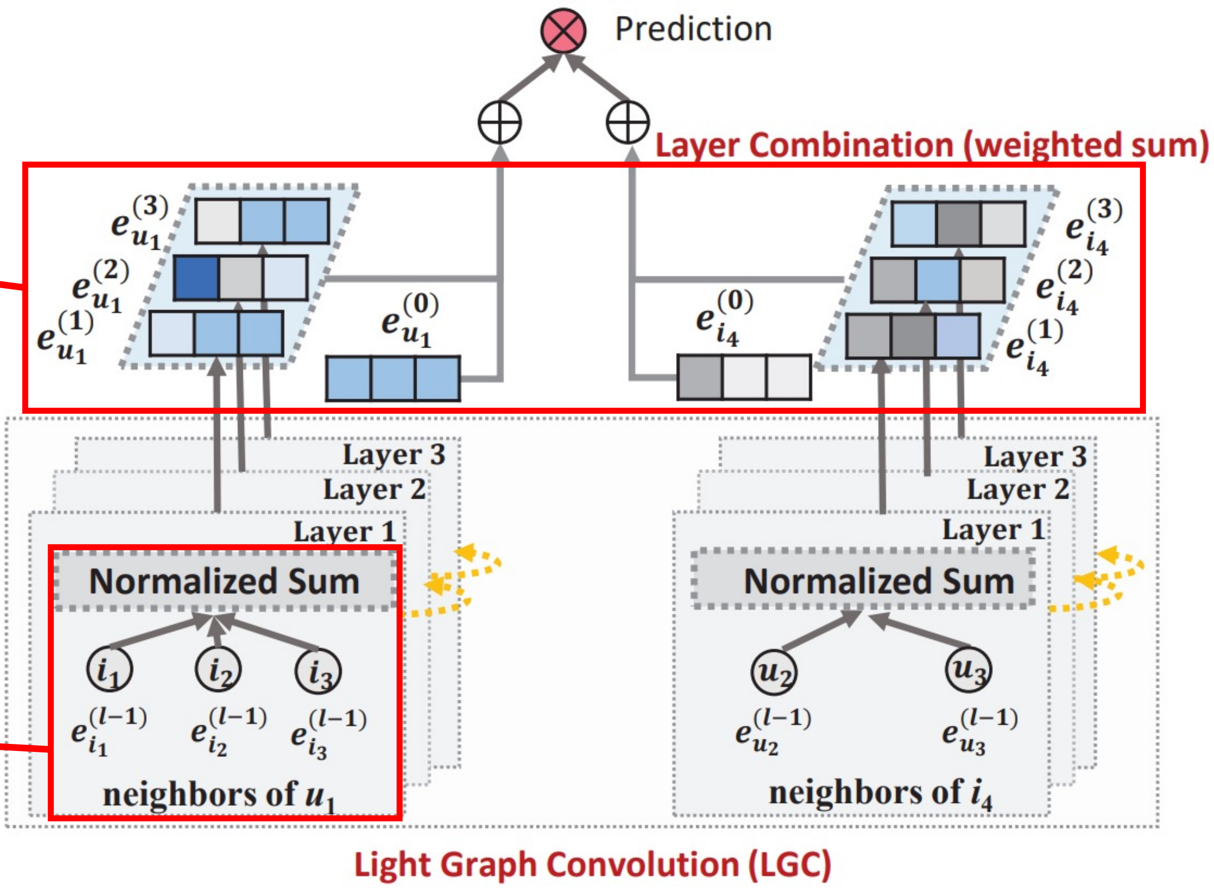
# LightGCN

## Simplifying GCN for recommendation

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

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

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




discard feature transformation and nonlinear activation

# GNN based Recommendation

## Collaborative Filtering

- Graph Convolutional Neural Networks for Web-Scale Recommender Systems (KDD’18)
- Graph Convolutional Matrix Completion (KDD’18 Deep Learning Day )
- Neural Graph Collaborative Filtering (SIGIR’19)
- LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation (SIGIR’20)

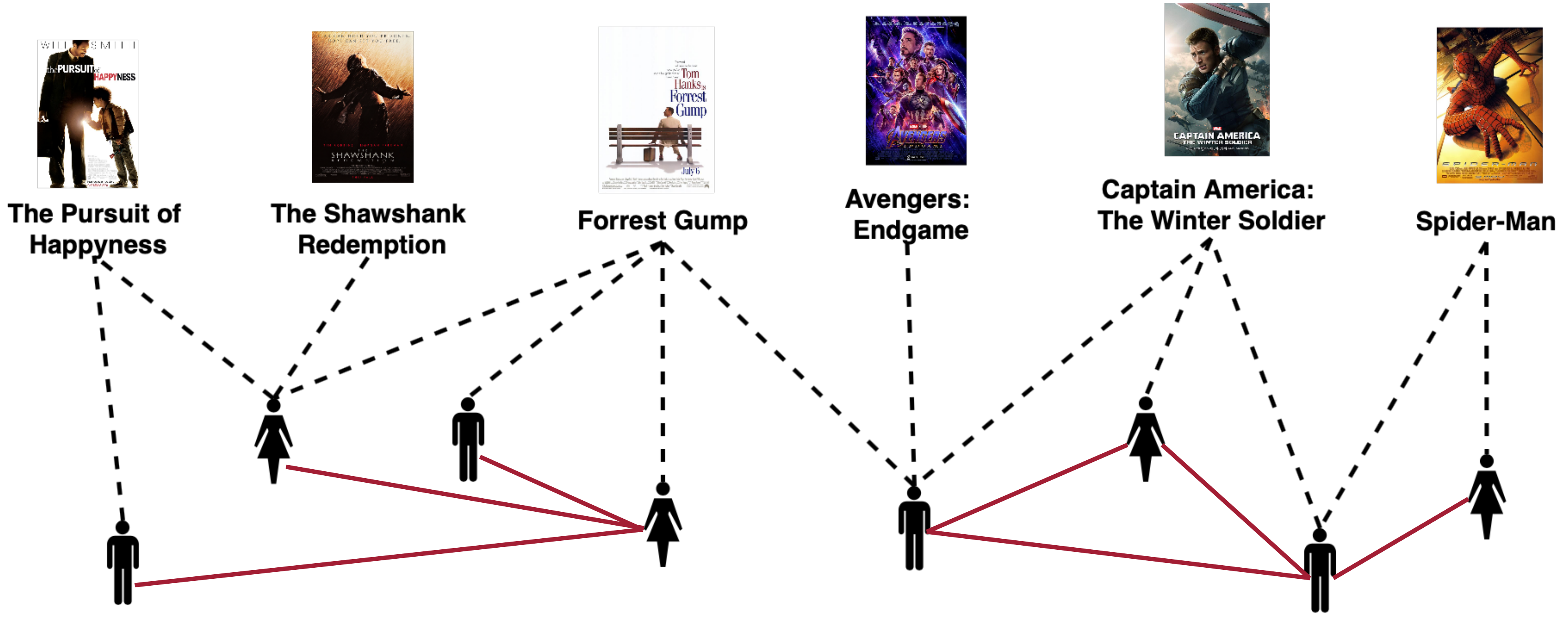
## Collaborative Filtering with Side Information (Users/Items)

- **Social Recommendation (Users)**
  - Graph Neural Network for Social Recommendation (WWW’19)
  - A Neural Influence Diffusion Model for Social Recommendation (SIGIR’19)
  - A Graph Neural Network Framework for Social Recommendations (TKDE’20)
- **Knowledge-graph-aware Recommendation (Items)**
  - Knowledge Graph Convolutional Networks for Recommender Systems with Label Smoothness Regularization (KDD’19 and WWW’19)
  - KGAT: Knowledge Graph Attention Network for Recommendation (KDD’19)

# Social Recommendation

## Side information about users: social networks

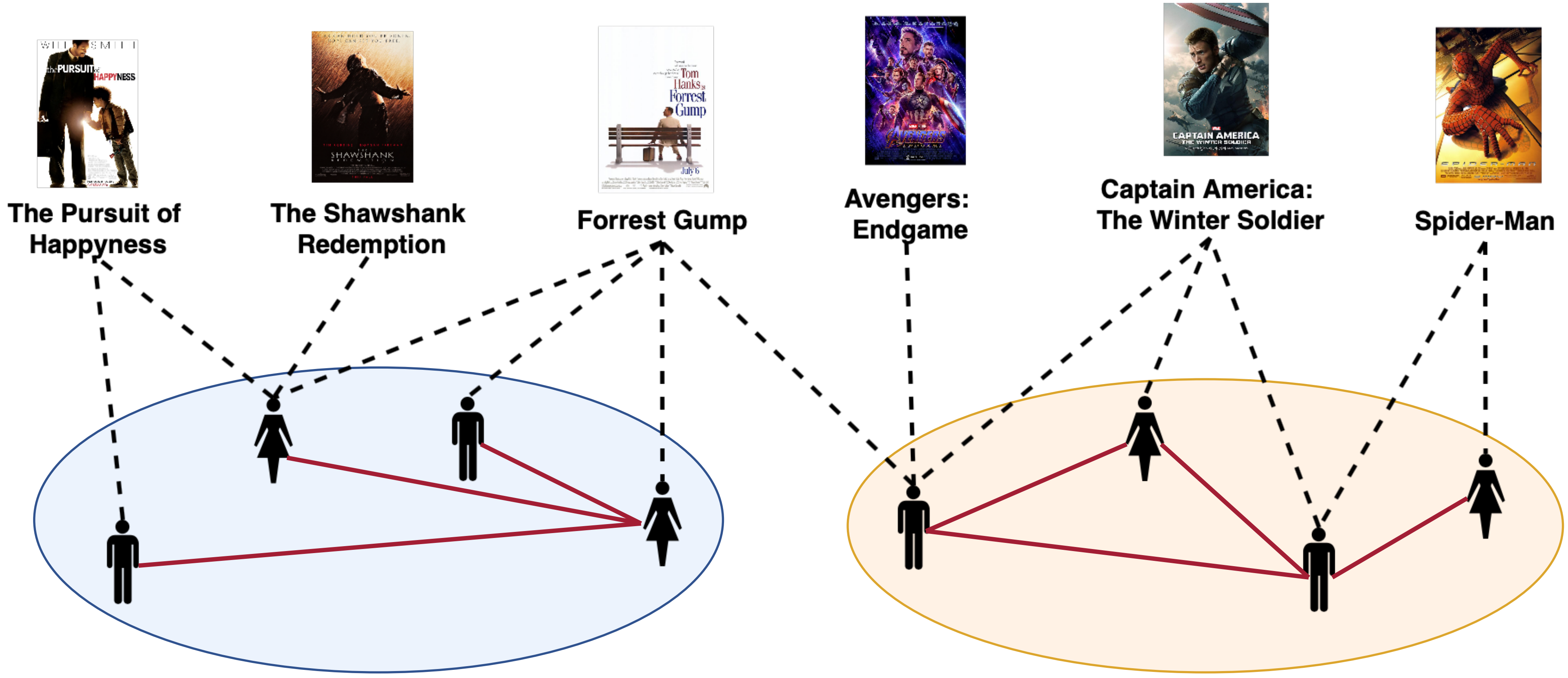
- **Users' preferences are similar to or influenced by the people around them (nearer neighbours)** [Tang et. al, 2013]



# Social Recommendation

## Side information about users: social networks

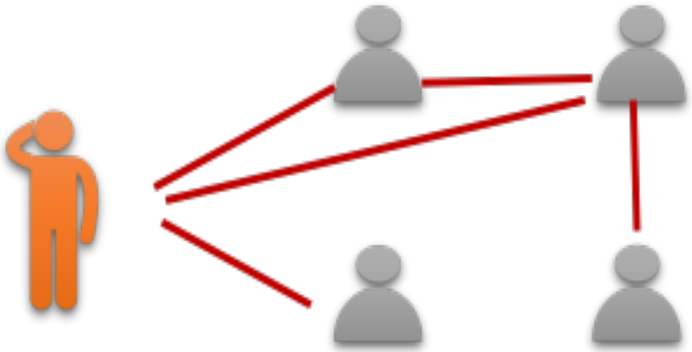
- Users' preferences **are similar to or influenced by** the people around them (nearer neighbours) [Tang et. al, 2013]



# GraphRec

## Graph Data in Social Recommendation


User-Item Graph

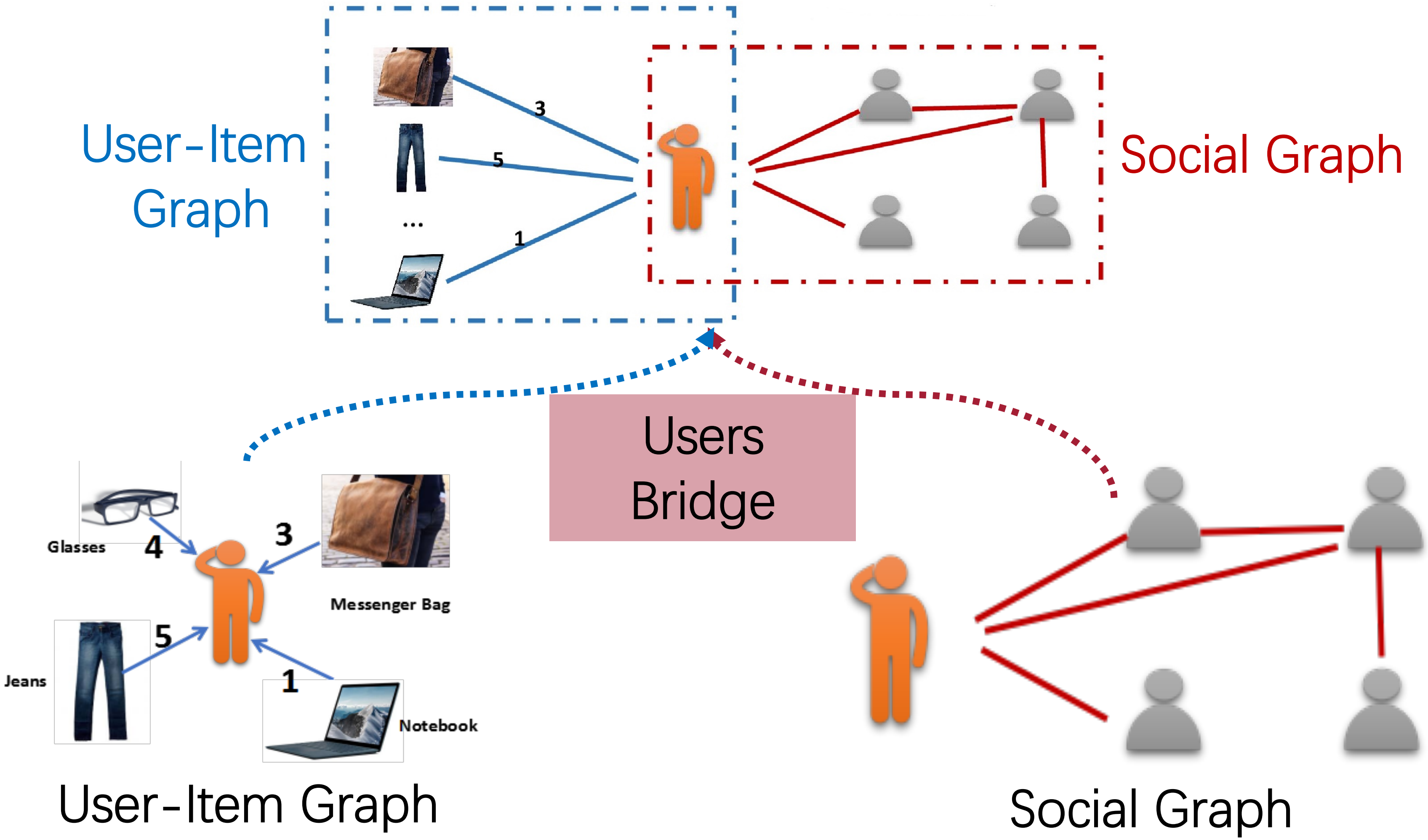


Social Graph

Graph Neural Networks for Social Recommendation. WWW 2019.

# GraphRec

## Graph Data in Social Recommendation

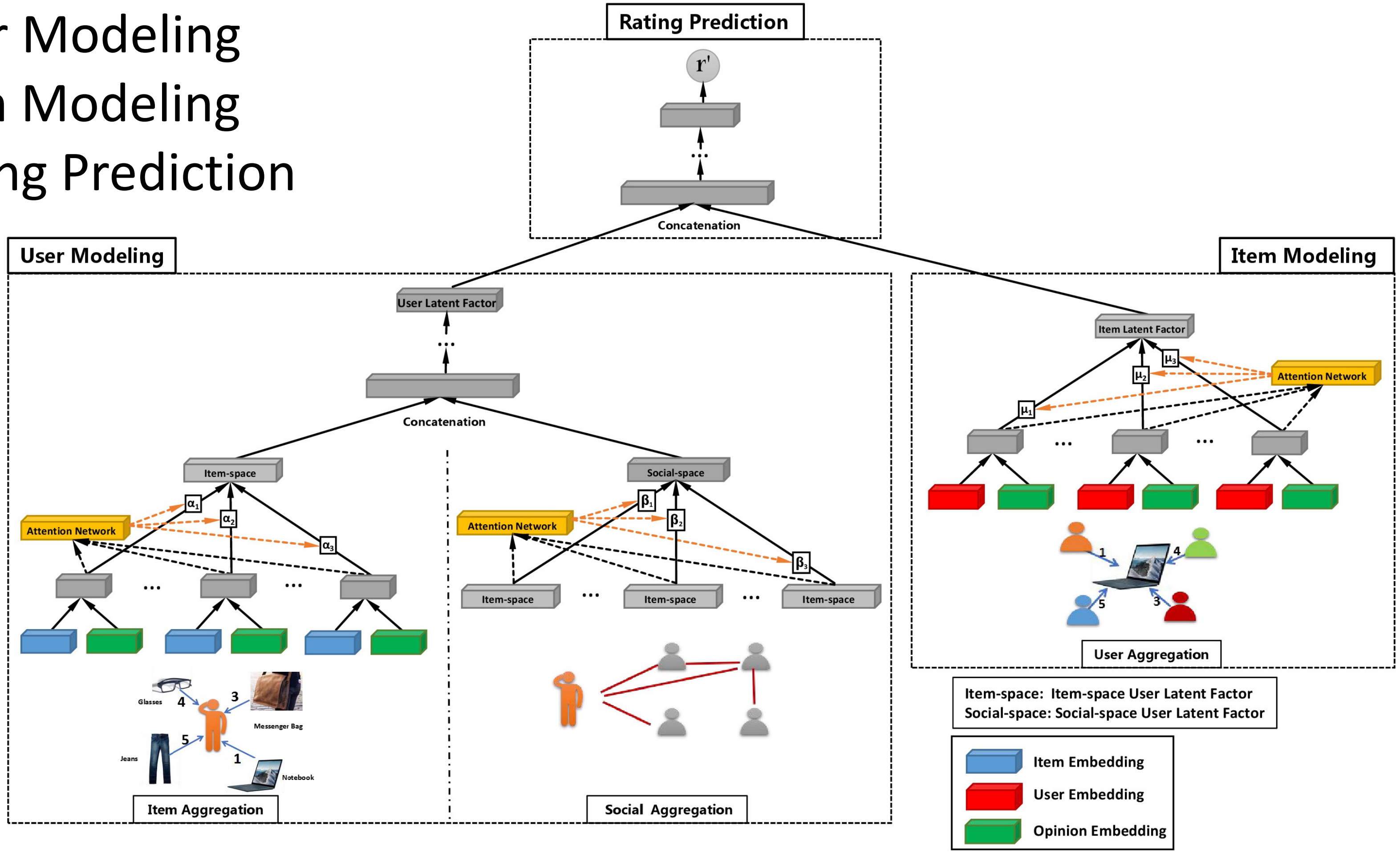



Graph Neural Networks for Social Recommendation. WWW 2019.

# GraphRec

Three Components:

- User Modeling
- Item Modeling
- Rating Prediction

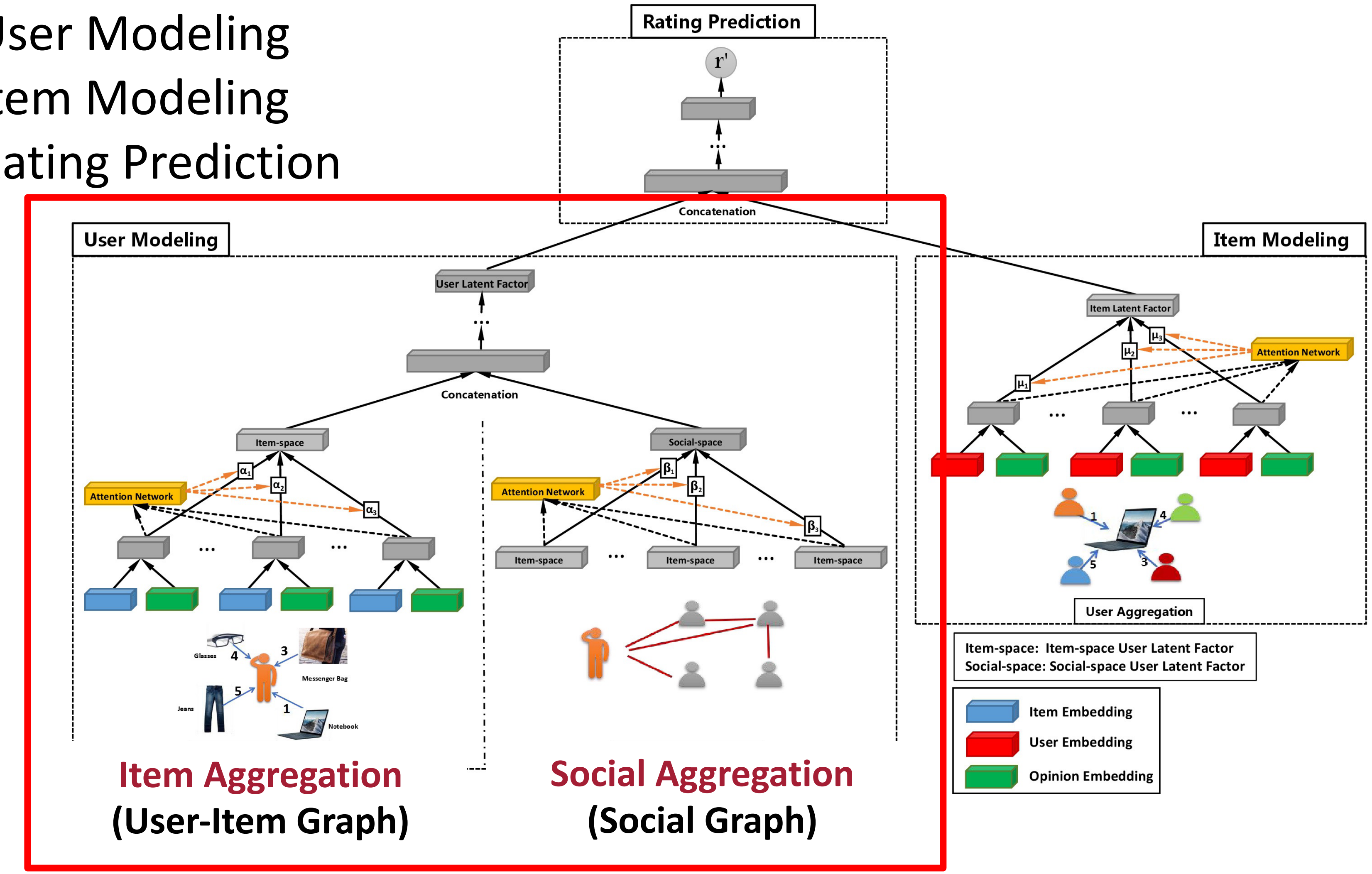



Graph Neural Networks for Social Recommendation. WWW 2019.

# GraphRec

Three Components:

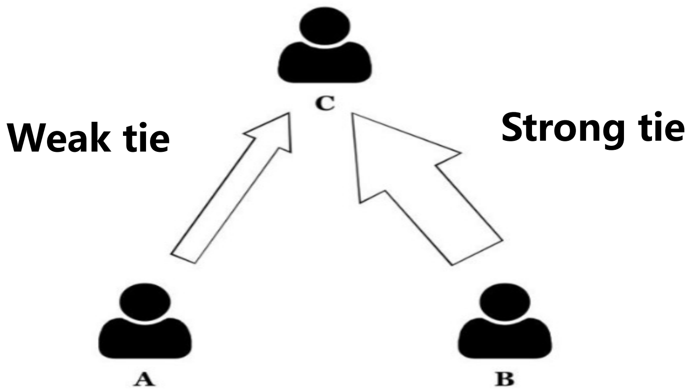
- User Modeling
- Item Modeling
- Rating Prediction




Graph Neural Networks for Social Recommendation. WWW 2019.

# GraphRec: User Modeling

- Social Aggregation in user-user social graph
- Users are likely to share more similar tastes with strong ties than weak ties.

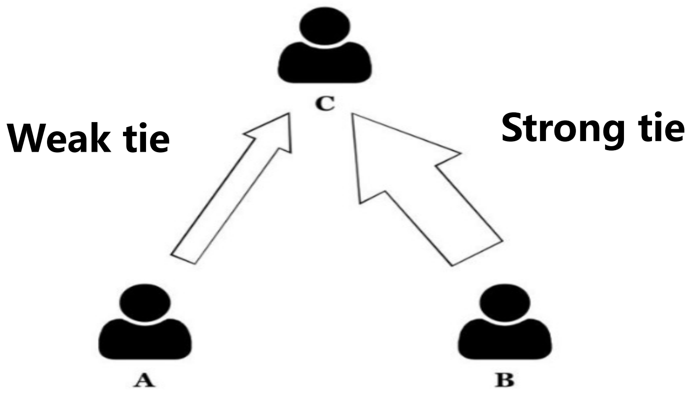



Graph Neural Networks for Social Recommendation. WWW 2019.

# GraphRec: User Modeling

- Social Aggregation in user-user social graph
- Users are likely to share more similar tastes with strong ties than weak ties.

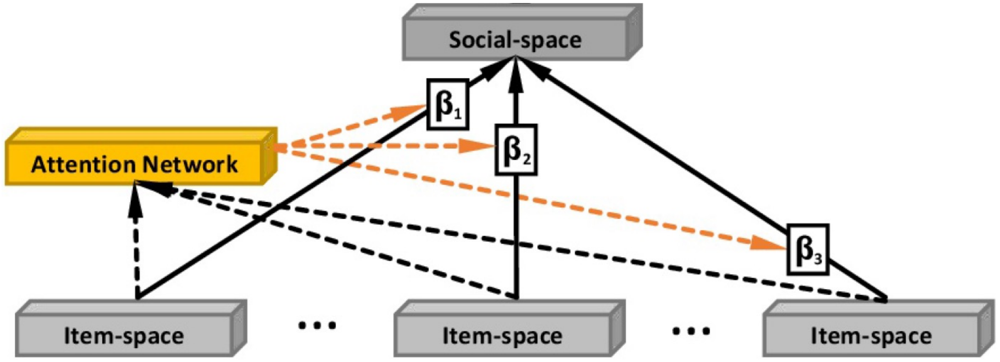
**Attention network to differentiate the importance weight.**



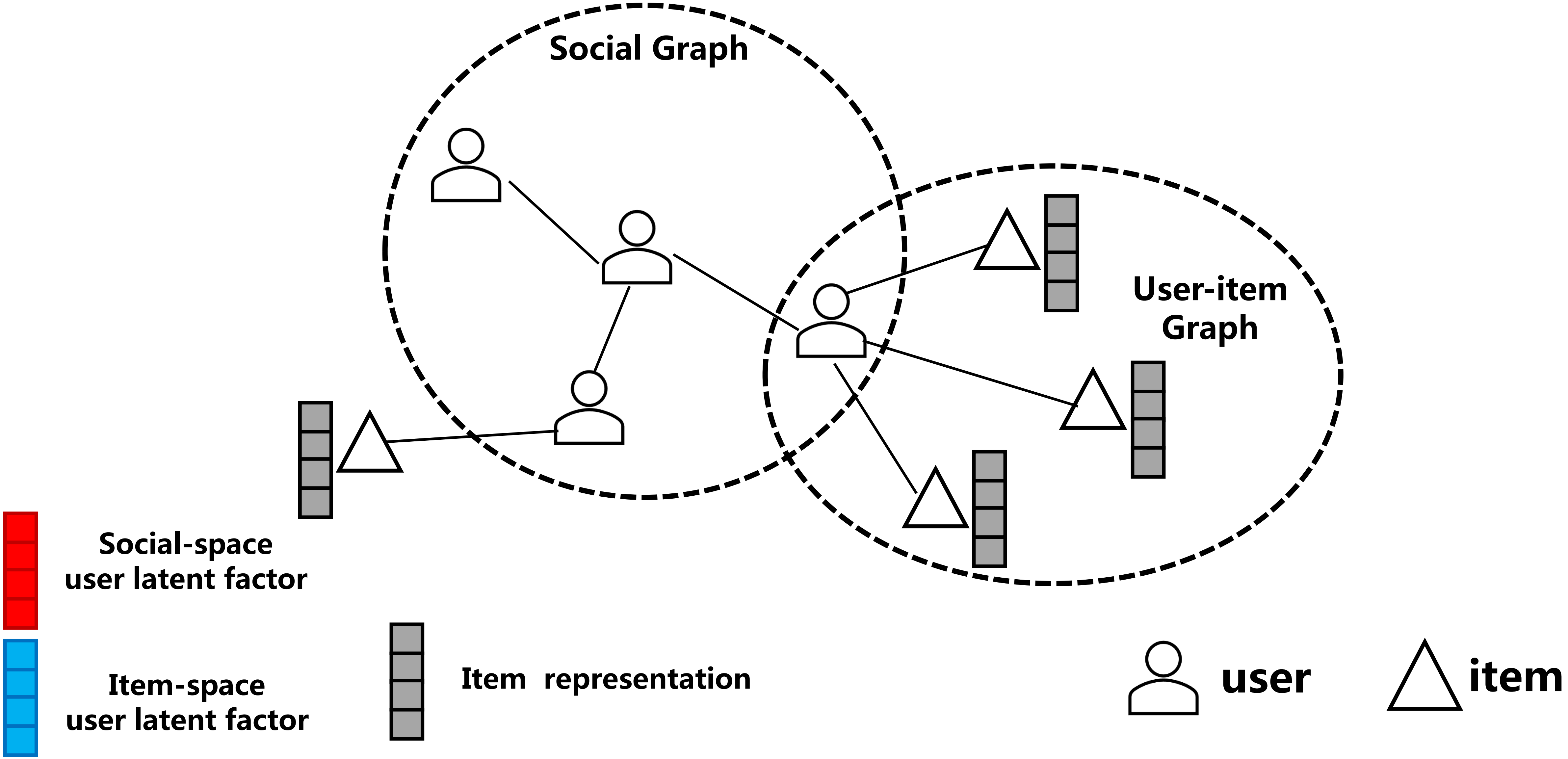

**Aggregating item-space users messages from social neighbors**

$$\mathbf{h}_i^S = \sigma(\mathbf{W} \cdot \left\{ \sum_{o \in N(i)} \boxed{\beta_{io}} \mathbf{h}_o^I \right\} + \mathbf{b})$$

**attentive weight**

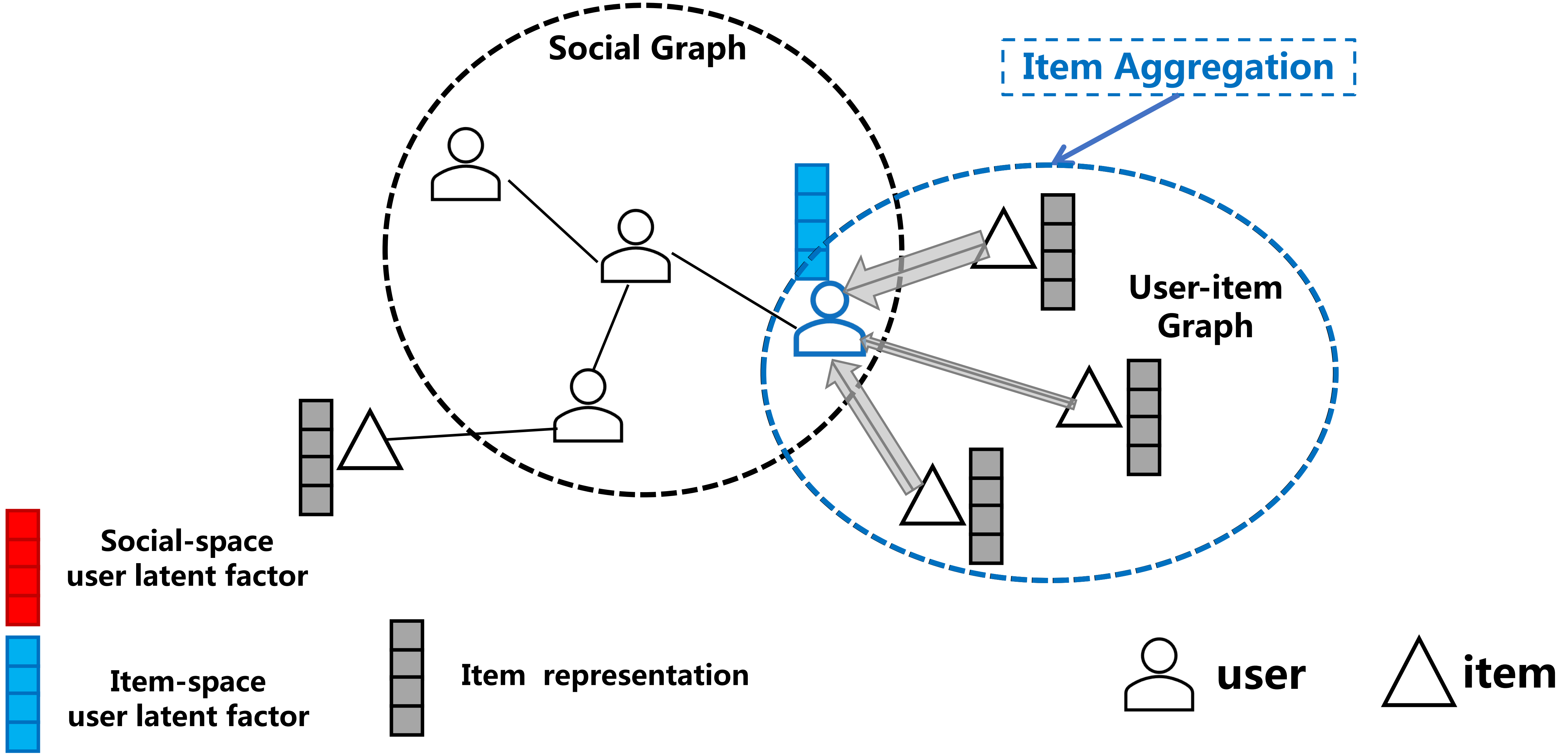


# User Modeling: Social Aggregation



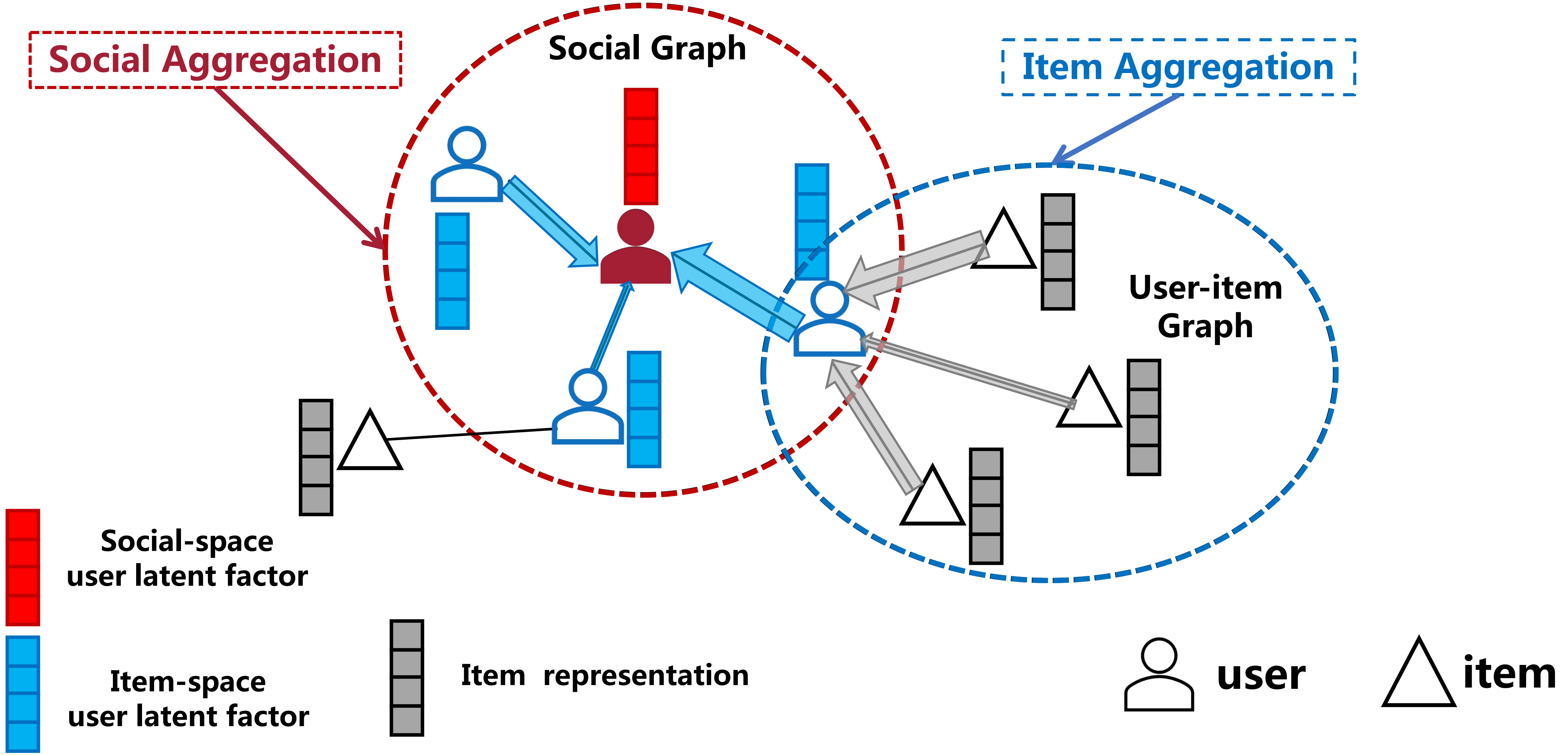

Graph Neural Networks for Social Recommendation. WWW 2019.

# User Modeling: Social Aggregation




Graph Neural Networks for Social Recommendation. WWW 2019.

# User Modeling: Social Aggregation




Graph Neural Networks for Social Recommendation. WWW 2019.

# GNNs based Recommendation

## Collaborative Filtering

- Graph Convolutional Neural Networks for Web-Scale Recommender Systems (KDD’18)
- Graph Convolutional Matrix Completion (KDD’18 Deep Learning Day )
- Neural Graph Collaborative Filtering (SIGIR’19)
- LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation (SIGIR’20)

## Collaborative Filtering with Side Information (Users/Items)

- **Social Recommendation (Users)**
  - Graph Neural Network for Social Recommendation (WWW’19)
  - A Neural Influence Diffusion Model for Social Recommendation (SIGIR’19)
  - A Graph Neural Network Framework for Social Recommendations (TKDE’20)
- **Knowledge-graph-aware Recommendation (Items)**
  - Knowledge Graph Convolutional Networks for Recommender Systems with Label Smoothness Regularization (KDD’19 and WWW’19)
  - KGAT: Knowledge Graph Attention Network for Recommendation (KDD’19)

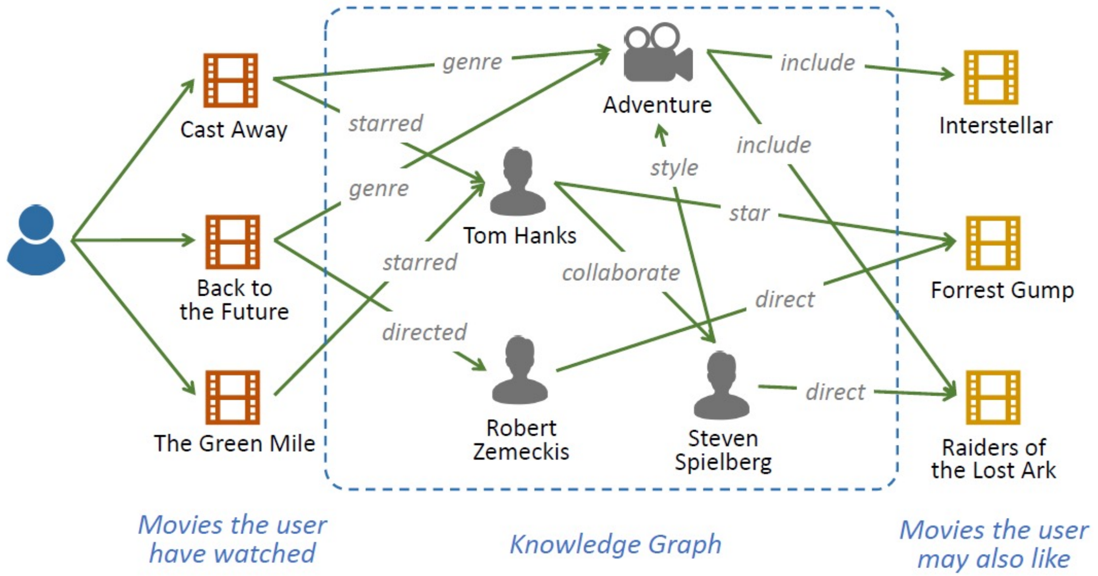
# KGCN (WWW'19)

## Side information about items: Knowledge Graph (KG)

**Heterogeneous Graph:**

- Nodes: entities (Items)
- Edges: relations

**Triples: (head, relation, tail)**




Knowledge Graph Convolutional Networks for Recommender Systems, WWW 2019.

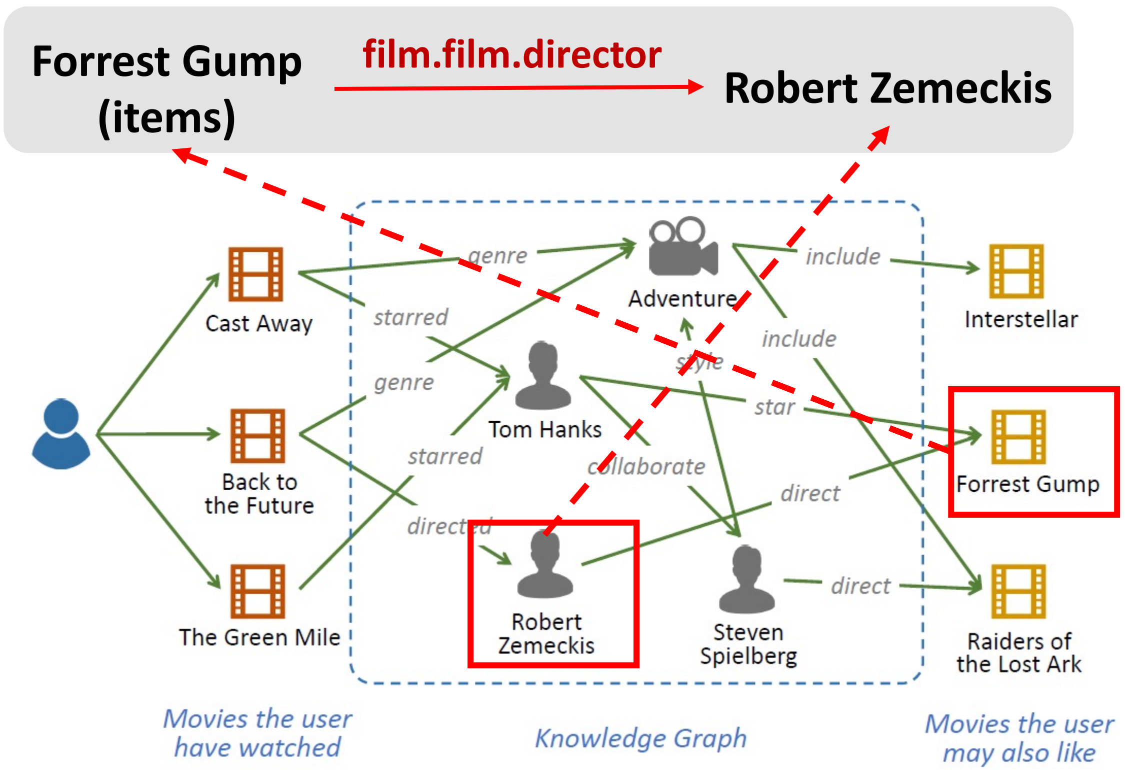
# KGCN (WWW'19)

## Side information about items: Knowledge Graph (KG)

**Heterogeneous Graph:**

- Nodes: entities (Items)
- Edges: relations

**Triples: (head, relation, tail)**




Knowledge Graph Convolutional Networks for Recommender Systems, WWW 2019.

# KGCN (WWW'19)

## Side information about items: Knowledge Graph (KG)

**Heterogeneous Graph:**

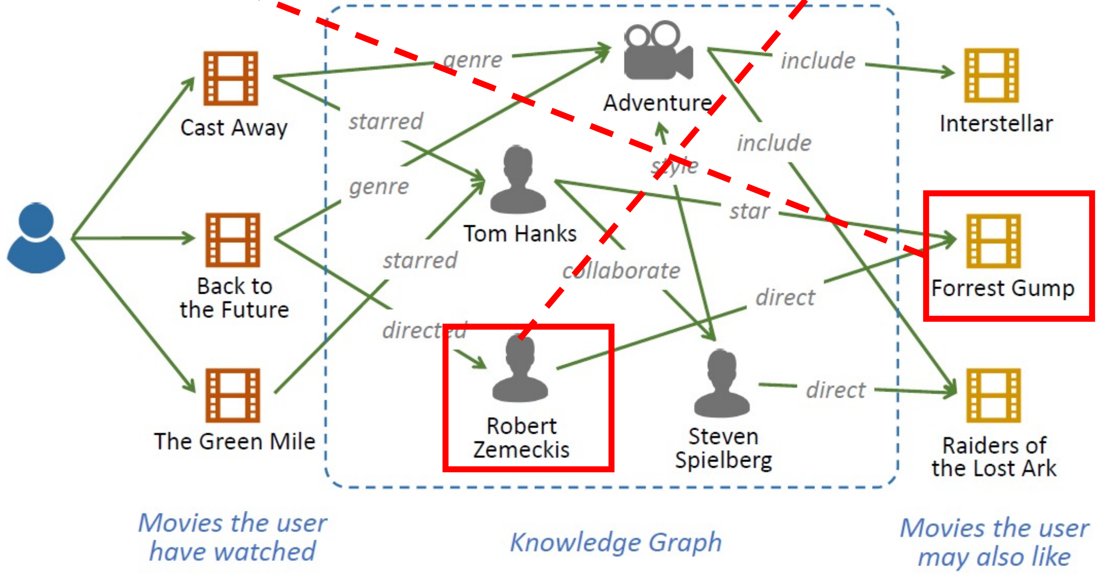
- Nodes: entities (Items)
- Edges: relations

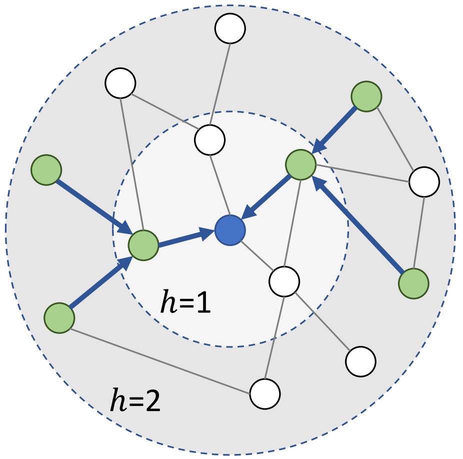
**Triples: (head, relation, tail)**

**Forrest Gump (items)** $\xrightarrow{\text{film.film.director}}$ **Robert Zemeckis**

$$\hat{y}_{uv} = f(\mathbf{u}, \boxed{\mathbf{v}^{u}})$$

**GNNs?**

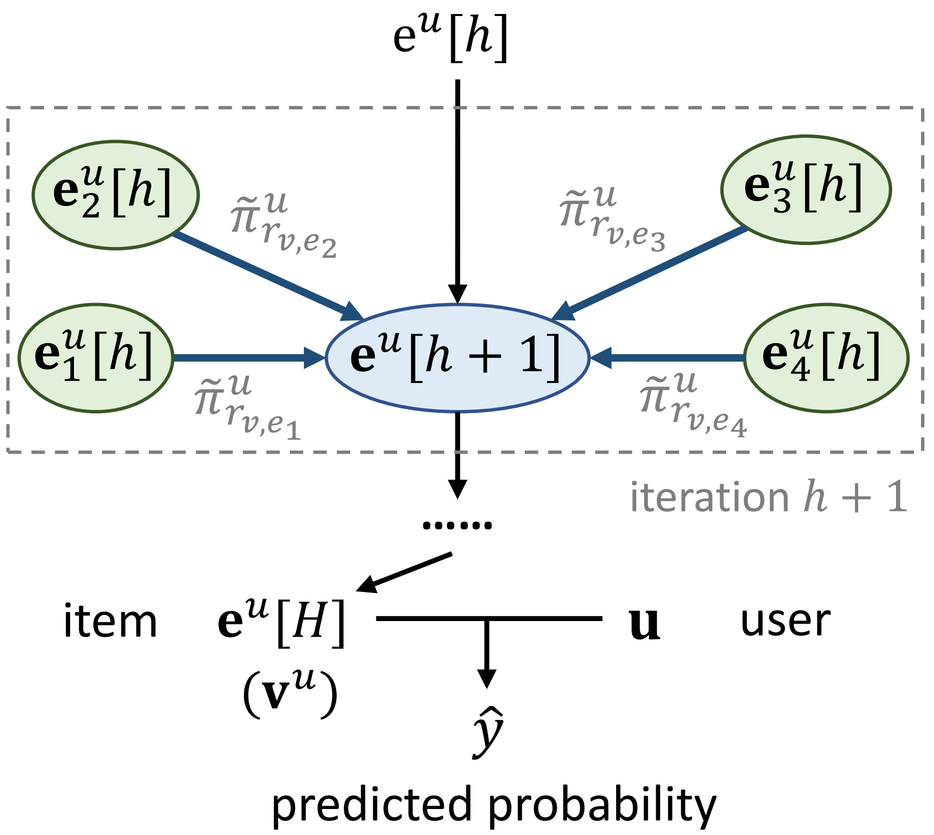



Knowledge Graph Convolutional Networks for Recommender Systems, WWW 2019.

# KGCN (WWW'19)

- Representation Aggregation of neighboring entities




Transform a heterogeneous KG into a user-personalized weighted graph

**Knowledge Graph Convolutional Networks for Recommender Systems.** WWW 2019.

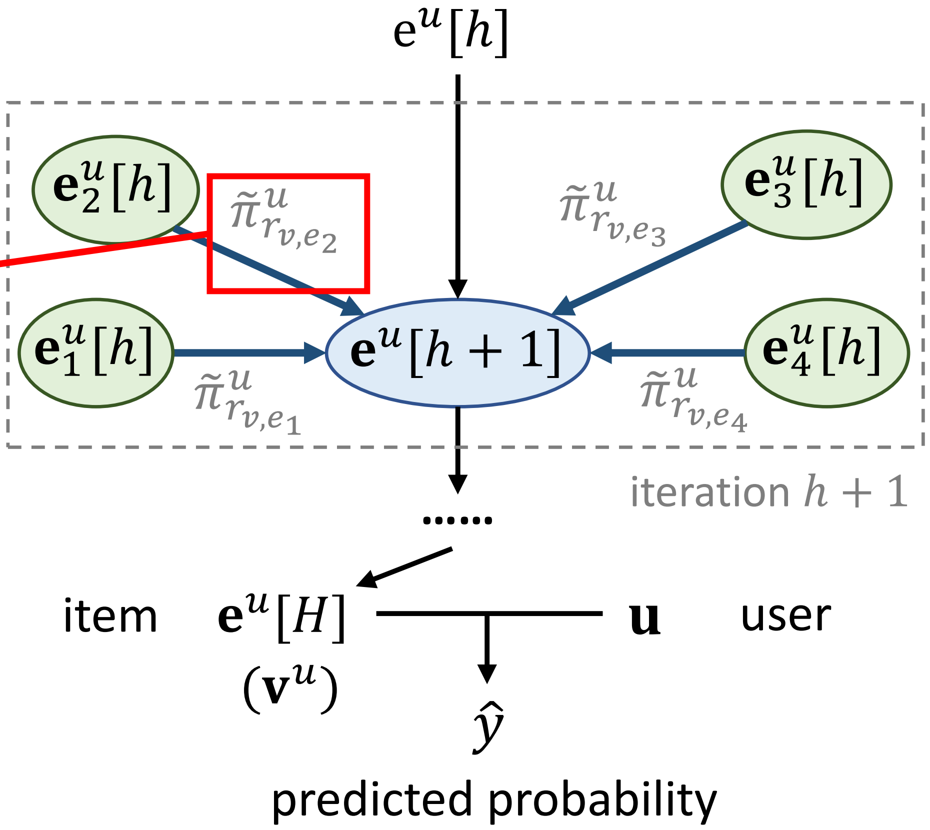
# KGCN (WWW'19)

- Representation Aggregation of neighboring entities

$$\pi_r^u = g(\mathbf{u}, \mathbf{r})$$ user-specific relation (e.g., inner product)

$$\tilde{\pi}_{r_{v,e}}^u = \frac{\exp(\pi_{r_{v,e}}^u)}{\sum_{e \in \mathcal{N}(v)} \exp(\pi_{r_{v,e}}^u)}$$

Normalized




Transform a heterogeneous KG into a user-personalized weighted graph

# KGCN (WWW'19)

- Representation Aggregation of neighboring entities

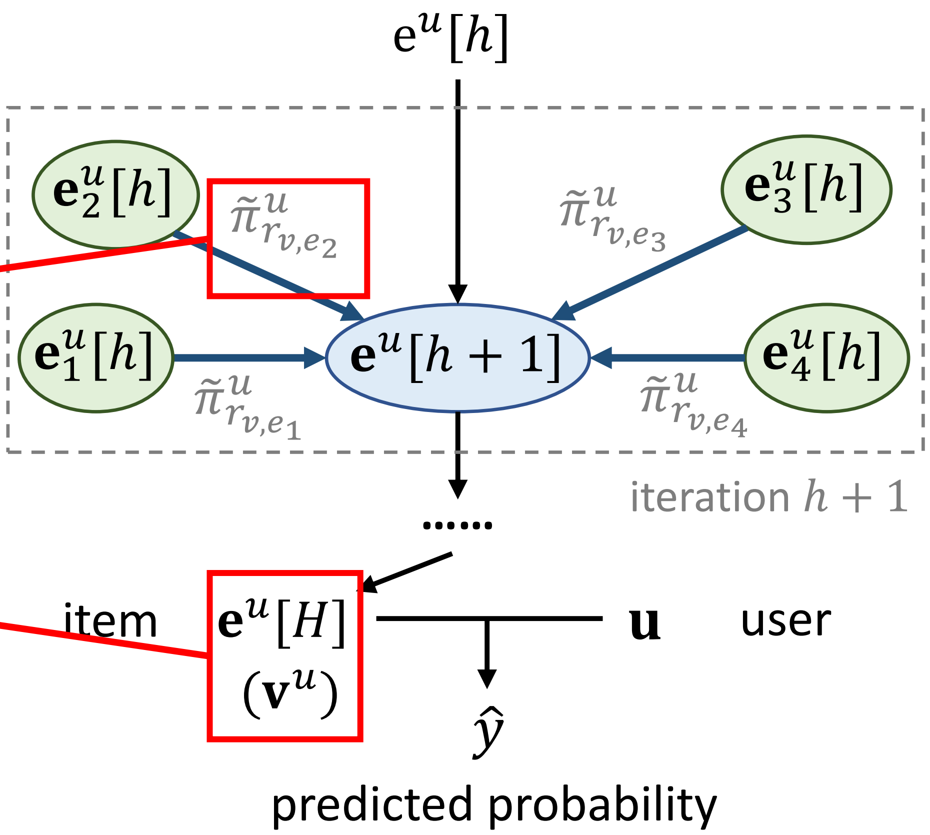
$$\pi_r^u = g(\mathbf{u}, \mathbf{r})$$ user-specific relation (e.g., inner product)

$$\tilde{\pi}^u_{r_{v,e}} = \frac{\exp\left(\pi^u_{r_{v,e}}\right)}{\sum_{e \in \mathcal{N}(v)} \exp\left(\pi^u_{r_{v,e}}\right)}$$

Normalized

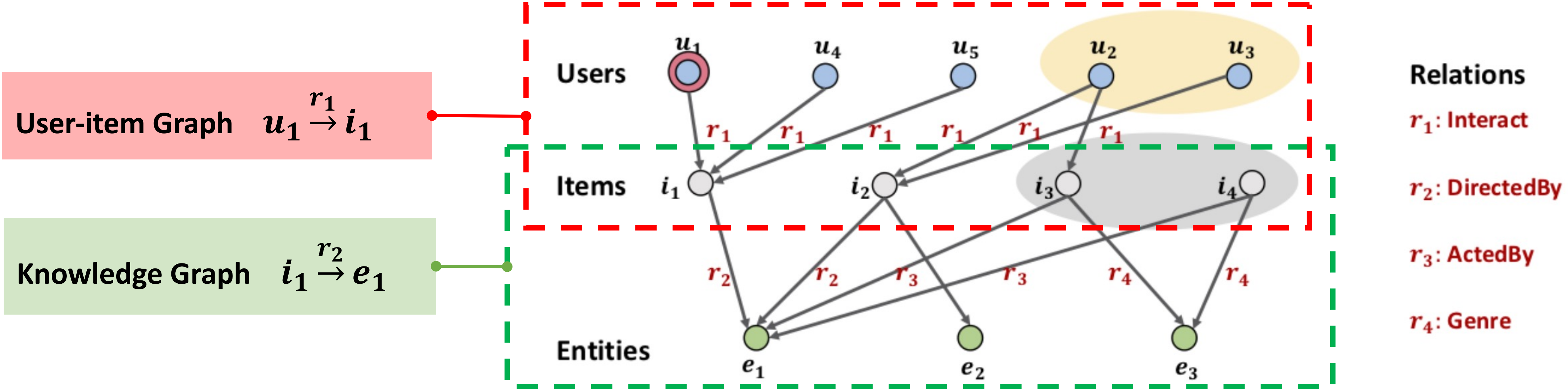
$$\mathbf{v}^u_{\mathcal{N}(v)} = \sum_{e \in \mathcal{N}(v)} \tilde{\pi}^u_{r_{v,e}} \mathbf{e}$$

$$\hat{y}_{uv} = f(\mathbf{u}, \mathbf{v}^u)$$




Transform a heterogeneous KG into a user-personalized weighted graph

# KGAT




**KGAT: Knowledge graph attention network for recommendation.** KDD 2019.

# KGAT

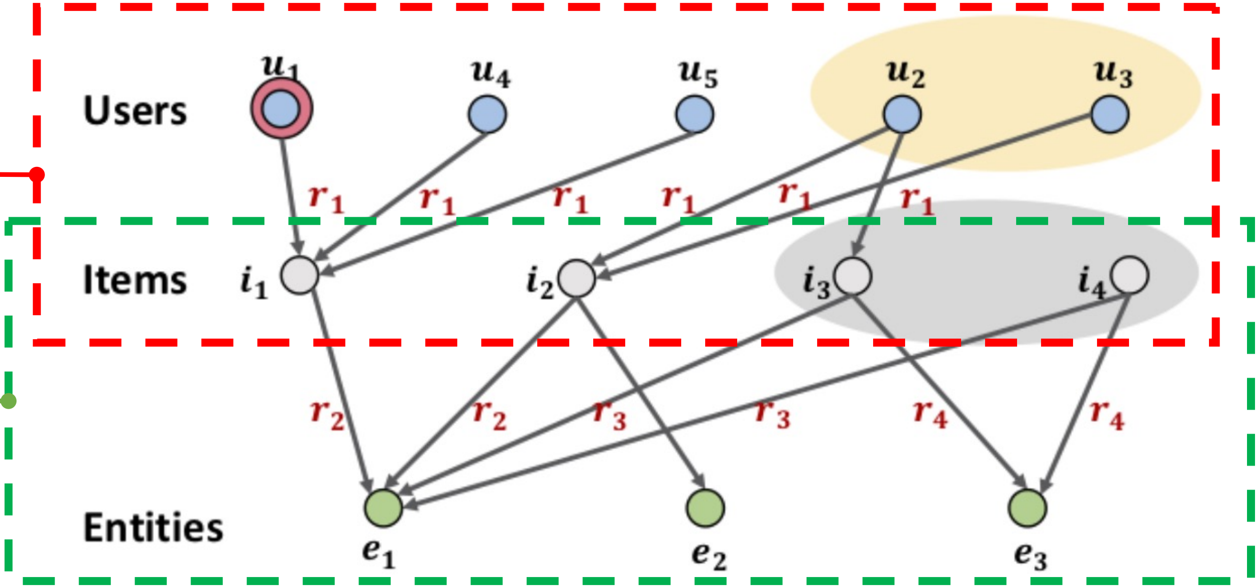
User-item Graph $u_1 \xrightarrow{r_1} i_1$

+

Knowledge Graph $i_1 \xrightarrow{r_2} e_1$

=

**Collaborative Knowledge Graph (CKG)**




**Relations**

$r_1$: Interact

$r_2$: DirectedBy

$r_3$: ActedBy

$r_4$: Genre

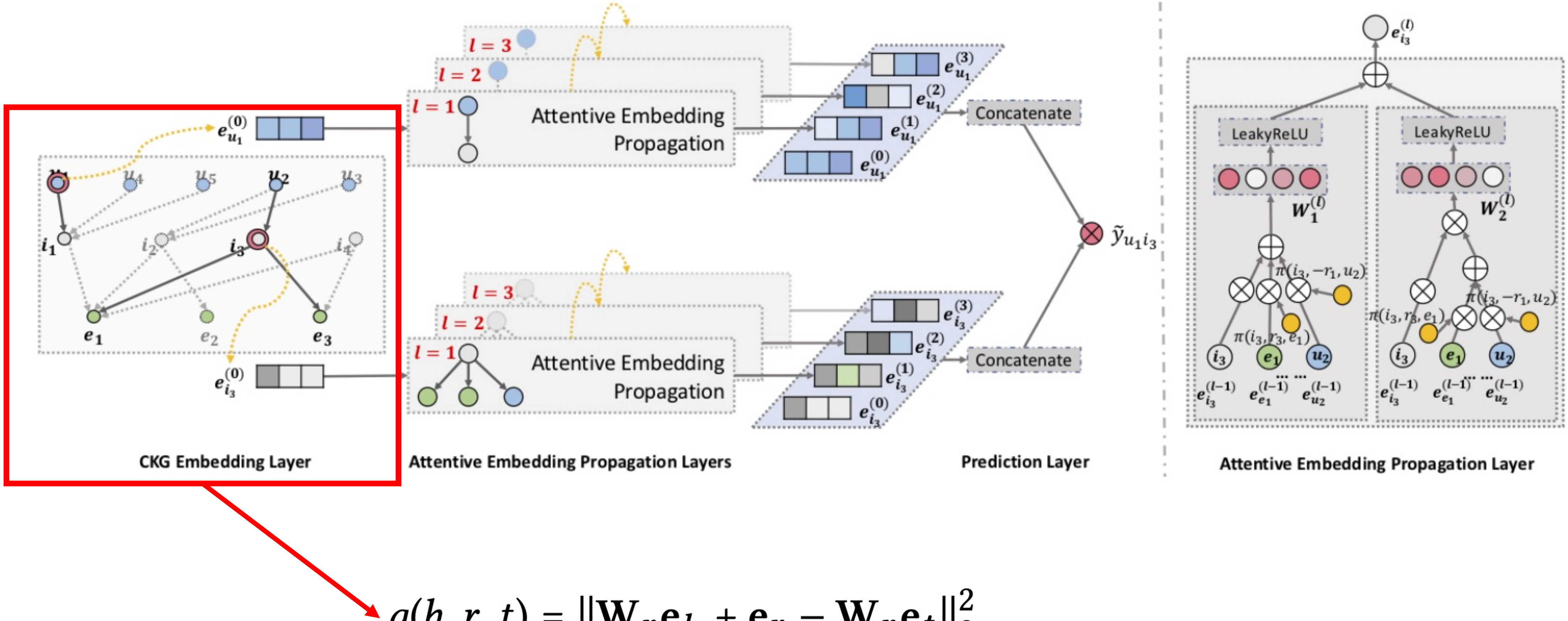
To fully exploit high-order relations in CKG
e.g., the long-range connectivities:

$$u_1 \xrightarrow{r_1} i_1 \xrightarrow{r_2} e_1 \xrightarrow{-r_2} i_2 \xrightarrow{-r_1} \{u_2, u_3\}$$

$$u_1 \xrightarrow{r_1} i_1 \xrightarrow{r_2} e_1 \xrightarrow{-r_3} \{i_3, i_4\}$$

**KGAT: Knowledge graph attention network for recommendation.** KDD 2019.

# KGAT

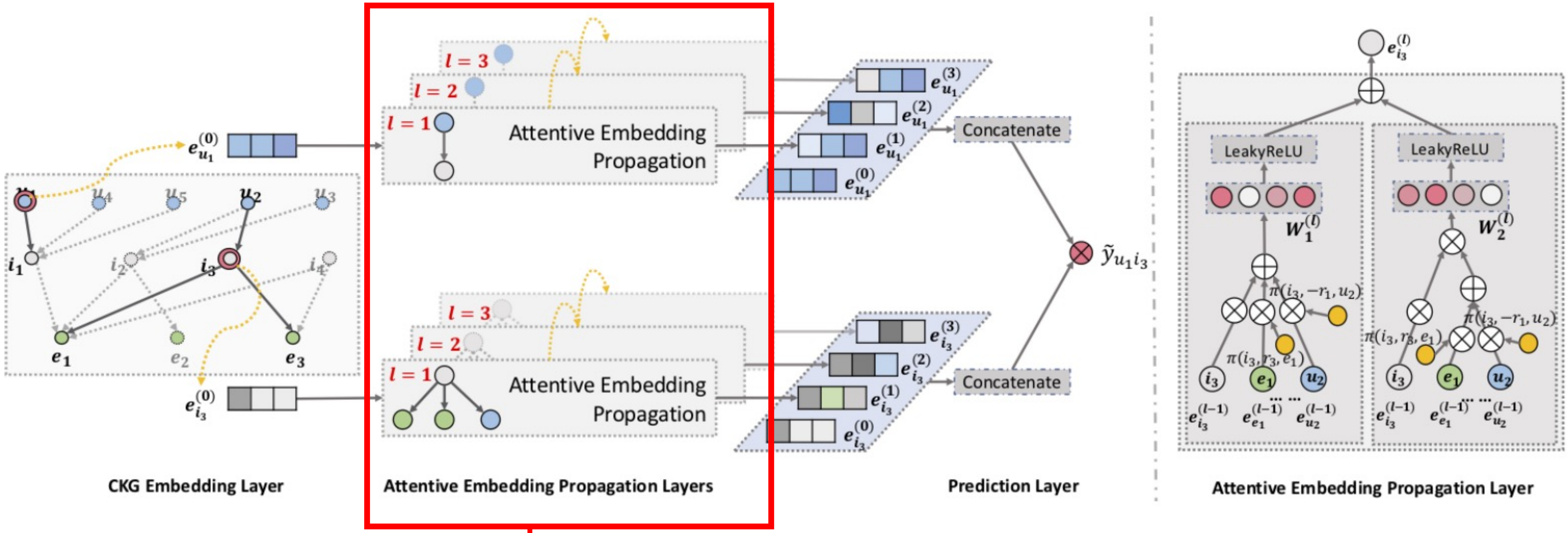



$$g(h, r, t) = \|\mathbf{W}_r \mathbf{e}_h + \mathbf{e}_r - \mathbf{W}_r \mathbf{e}_t\|_2^2$$

$$\mathcal{L}_{\mathrm{KG}} = \sum_{(h,r,t,t') \in \mathcal{T}} -\ln \sigma\Big(g(h, r, t') - g(h, r, t)\Big)$$

**KGAT: Knowledge graph attention network for recommendation.** KDD 2019.

# KGAT



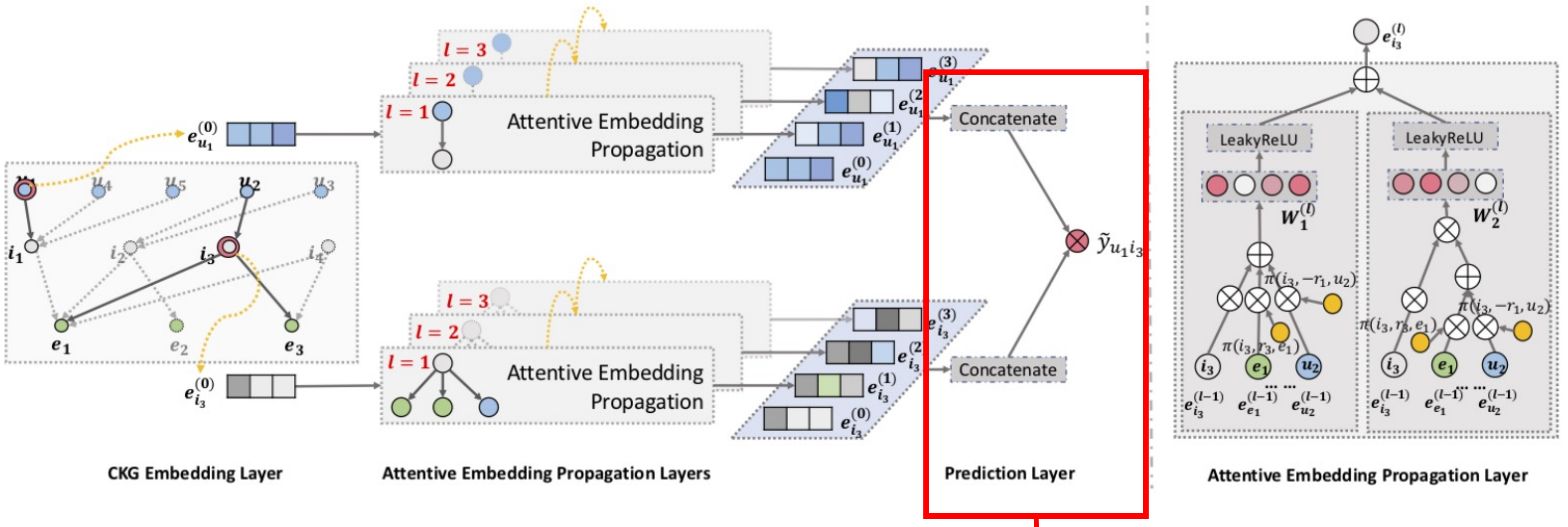

Information Propagation: $\mathbf{e}_{\mathcal{N}_h} = \sum_{(h,r,t)\in\mathcal{N}_h} \pi(h,r,t)\mathbf{e}_t$

Knowledge-aware Attention: $\pi(h,r,t) = (\mathbf{W}_r\mathbf{e}_t)^\top \tanh\big((\mathbf{W}_r\mathbf{e}_h + \mathbf{e}_r)\big)$

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

**KGAT: Knowledge graph attention network for recommendation.** KDD 2019.

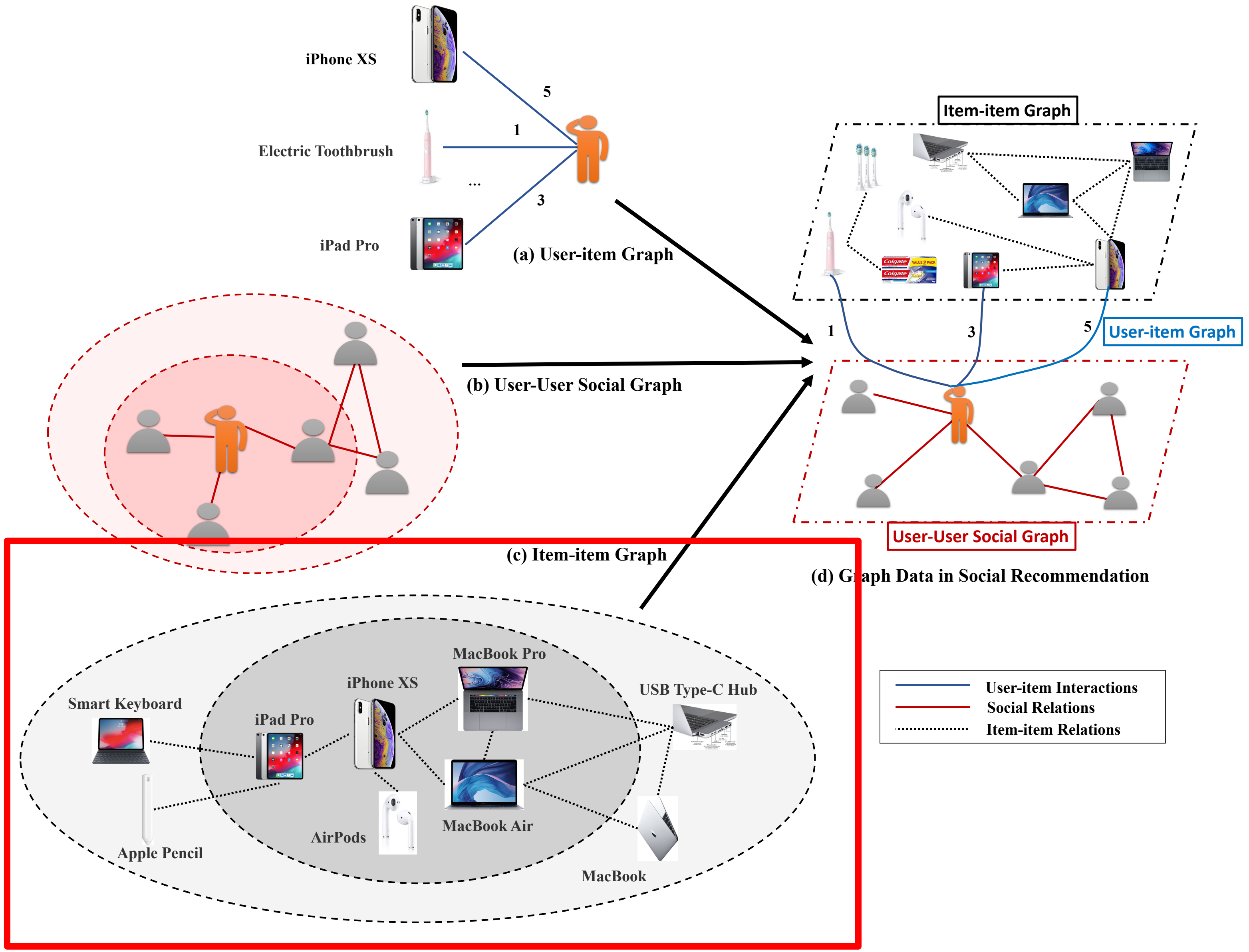
# KGAT




$$\mathbf{e}_u^* = \mathbf{e}_u^{(0)} \| \cdots \| \mathbf{e}_u^{(L)}, \quad \mathbf{e}_i^* = \mathbf{e}_i^{(0)} \| \cdots \| \mathbf{e}_i^{(L)} \qquad \hat{y}(u,i) = {\mathbf{e}_u^*}^\top \mathbf{e}_i^*$$

**KGAT: Knowledge graph attention network for recommendation.** KDD 2019.

# GraphRec+




A Graph Neural Network Framework for Social Recommendations, TKDE 2020
Graph Neural Networks for Social Recommendation, WWW 2019.

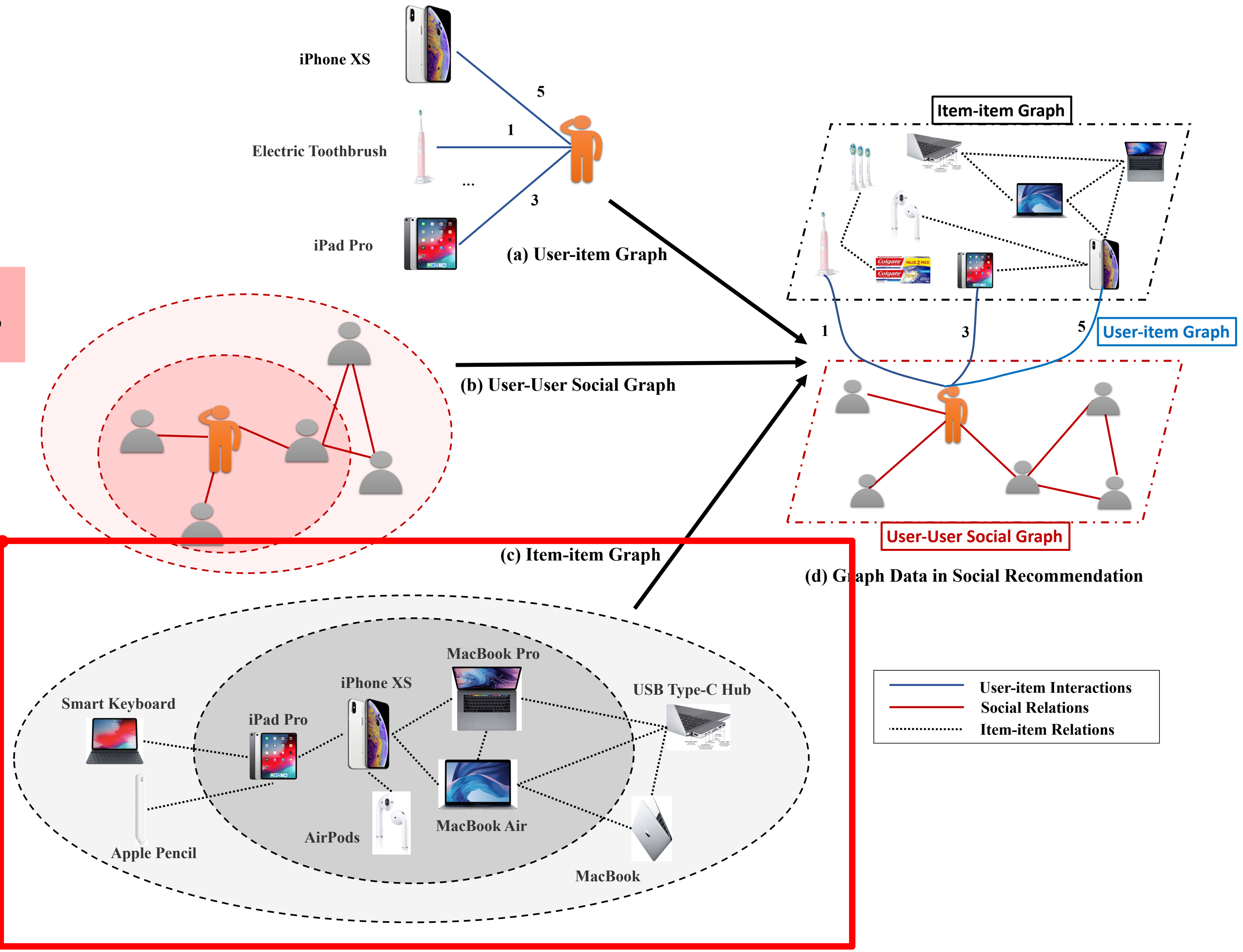
# GraphRec+

## Item-item Graph

**Substitutable and Complementary Items**

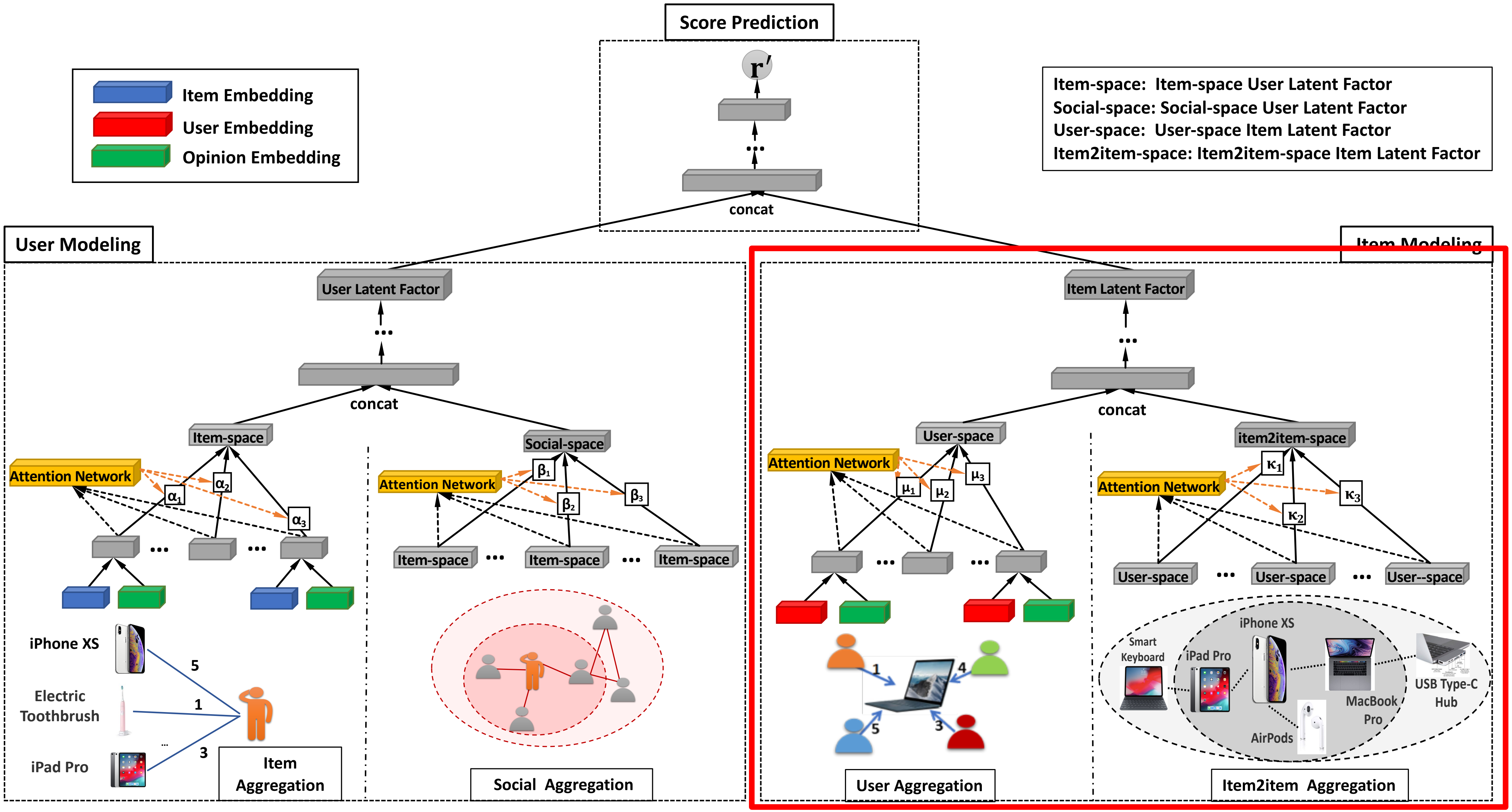
**E.g.,**

- ‘users who bought A also bought B’
- ‘users who viewed A also viewed B’




A Graph Neural Network Framework for Social Recommendations, TKDE 2020
Graph Neural Networks for Social Recommendation, WWW 2019.

# GraphRec+




A Graph Neural Network Framework for Social Recommendations, TKDE 2020
Graph Neural Networks for Social Recommendation, WWW 2019.

# Conclusion: Future Directions

- **Depth**
  When the deeper GNNs can help in recommender systems?

# Conclusion: Future Directions

- **Depth**

  When the deeper GNNs can help in recommender systems?

- **Security (Data Poisoning Attack & Defense)**
  - Edge

    user-item interactions

    social relations

    knowledge graph
  - Node (users/items) Features
  - Local Graph Structure



Attacking Black-box Recommendations via Copying Cross-domain User Profiles, ICDE 2021.

# Conclusion: Future Directions

- **Depth**

  When the deeper GNNs can help in recommender systems?

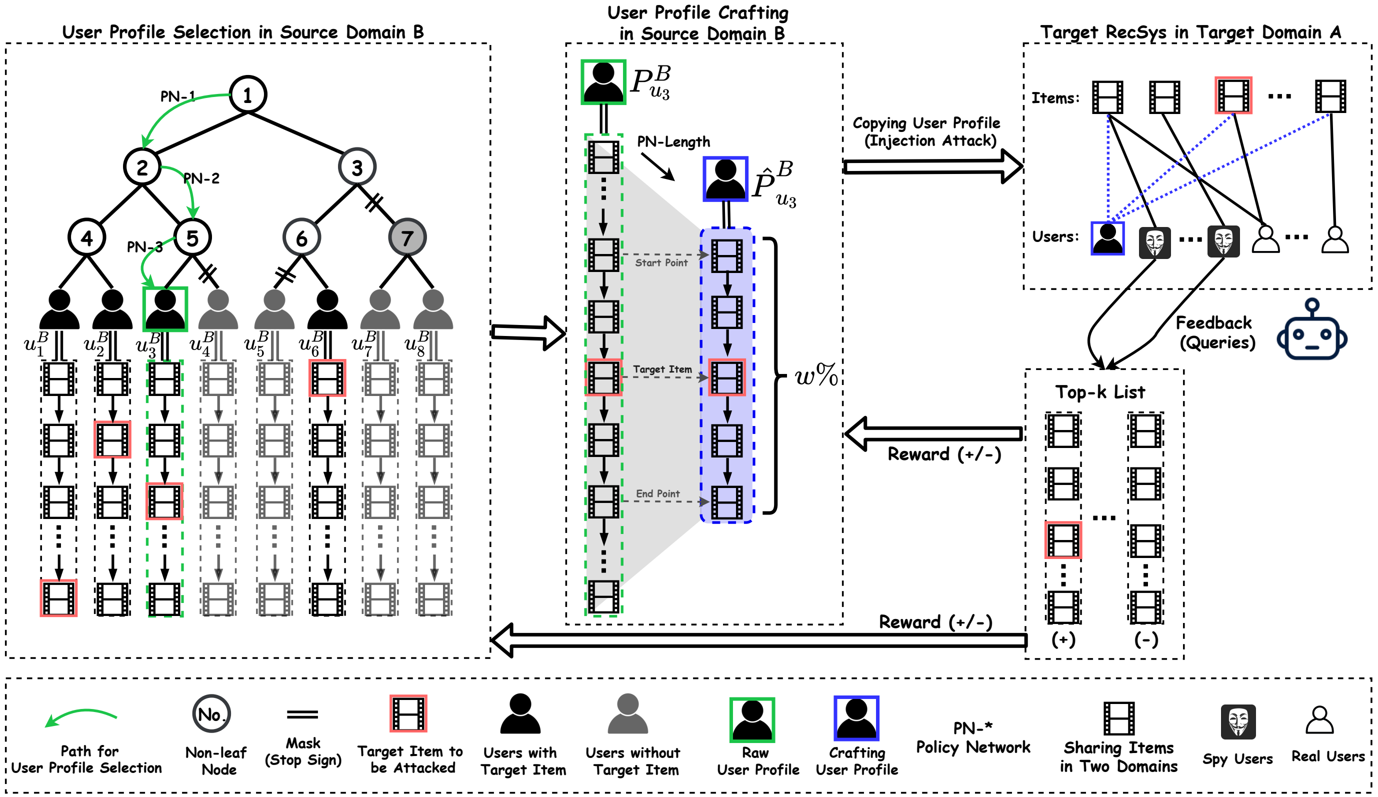
- **Security (Data Poisoning Attack & Defense)**
  - Edge

    user-item interactions

    social relations

    knowledge graph
  - Node (users/items) Features
  - Local Graph Structure

CopyAttack




Attacking Black-box Recommendations via Copying Cross-domain User Profiles, ICDE 2021.

# Ads

- **Wenqi Fan,** Computing, The Hong Kong Polytechnic University
- wenqifan@polyu.edu.hk
- https://wenqifan03.github.io


I am actively recruiting self-motivated Ph.D. students, Master, and Research Assistants. Visiting scholars and interns are also welcome. Send me an email with your CV if you are interested.