Graph neural networks for social recommendation

Author: Wenqi Fan, Yao Ma el al.

Presenter: Jingyuan Chou, Wenbo Pang, Zhiming Fan

UVA ENGINEERING

Outline

1. Introduction
2. Motivation and Challenge
3. Problem Formulation
4. Methodology
5. Experiments and Results
6. Model Analysis
7. Summary

Introduction

1. Graph serves as a common form of interactions between users or items.
    a. Social Network is the typical graph in real-world activities
    b. It’s common scenario that users acquire or disseminate information through people around them, such as friends, classmates or parents, etc.
    c. Except for interaction between users, the relationship between item and user can also be viewed as a type of interaction: rating
    d. Interaction between user’s social network can boost the information passing for the corresponding item
        i. User will recommend item which he think is great to his close friends
2. This kind of **message passing** through graph-like network is perfect for **Graph Neural Network.**

Motivation and Challenges

1. **Motivation**: As social connection can boost the message passing through the whole network, it can be naturally formed as a message passing problem on graph
2. **Challenges**:
   a. **How to combine these two graphs?**
   b. **how to capture interactions and opinions between users and items jointly ?**
   c. **Strong ties and weak ties?**

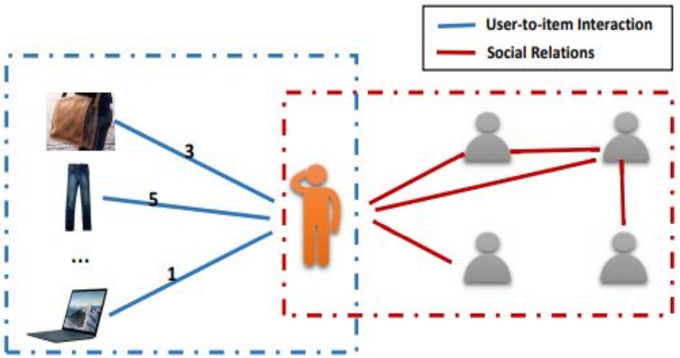


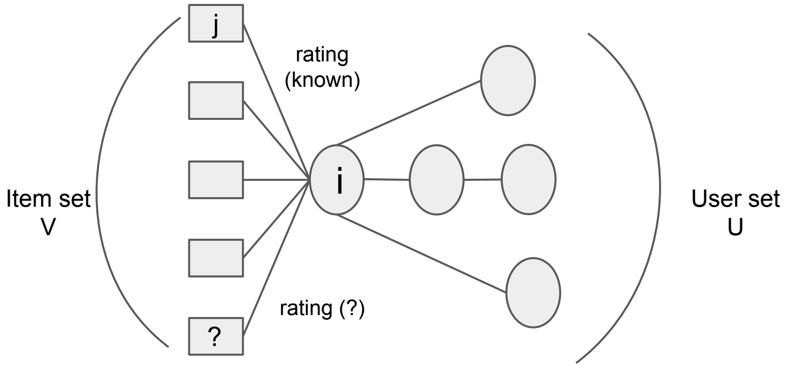
Figure 1: Graph Data in Social Recommendation. It contains two graphs including the user-item graph (left part) and the user-user social graph (right part). Note that the number on the edges of the user-item graph denotes the opinions (or rating score) of users on the items via the interactions.

All lead to representations of users and items!

UVA ENGINEERING

Problem Formulation

1. $U = \{u_1, u_2, ..., u_n\}$ as the set of users, total n users
2. $V = \{v_1, v_2, ..., v_m\}$ as the set of items, total m items
3. **User - item rating graph:**
    a. Rating matrix (user-item graph) is formed as $\mathbf{R}^{n * m}$
    b. $O = \{\langle u_i, v_j \rangle | r_{ij} \neq 0\}$ is the set of known ratings
    c. $\mathcal{T} = \{\langle u_i, v_j \rangle | r_{ij} = 0\}$ is the set of unknown ratings
    d. N(i) is the set of users whom u_i directly connected with
    e. C(i) is the set of items which u_i have interacted with
    f. B(j) is the set of users who have interacted with item v_j
4. **User - User social graph:**
    a. **n * n** matrix **T**, 1 if relationship exist between two users, 0 otherwise
5. Given both two graphs, **R** and **T, we aim to predict missing rating value in R.**



Methodology Overview

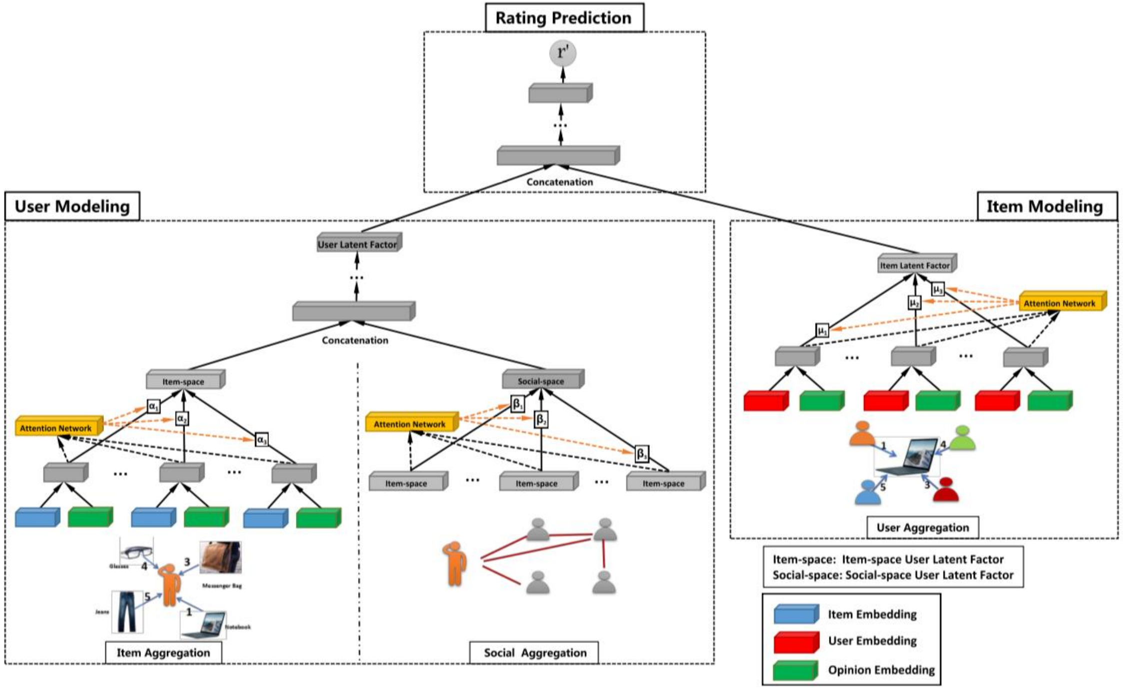
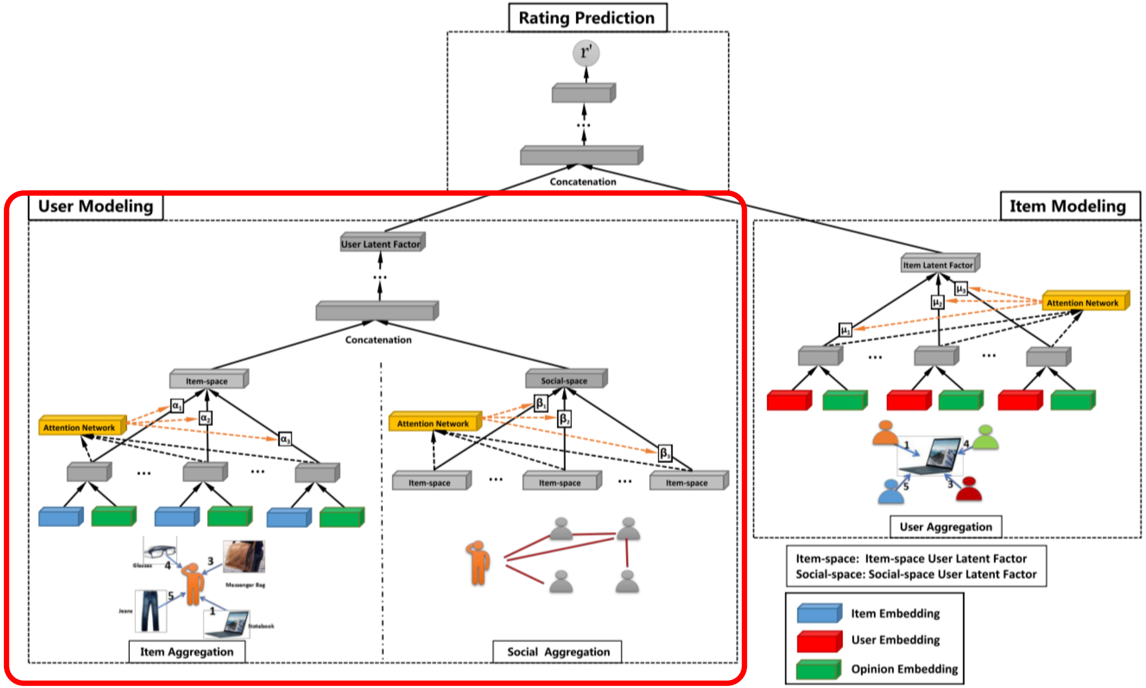


Figure 2: The overall architecture of the proposed model. It contains three major components: user modeling, item modeling, and rating prediction.

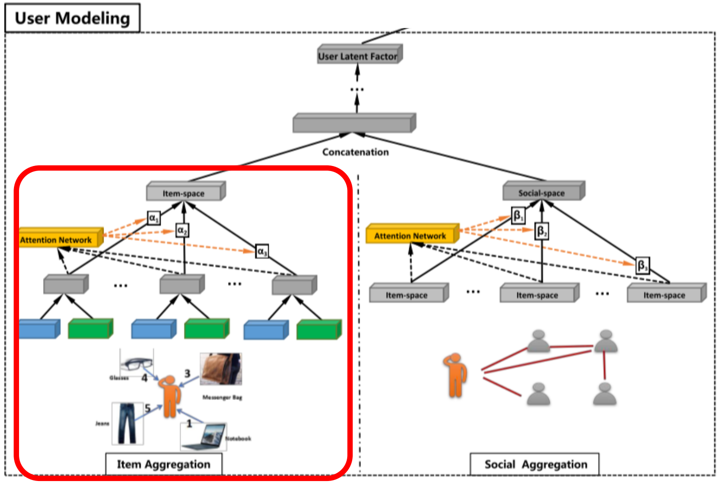
Methodology Overview

✓ User Modeling



User Modeling

- **Aim**: user latent factors: $\boldsymbol{h}_i \in \mathbb{R}^d$ for user u_i
- **Two aggregation**: Item Aggregation & Social Aggregation



✓ Item Aggregation

(1) Item Aggregation

- To learn item-space user latent factor $\boldsymbol{h}_i^I \in \mathbb{R}^d$ from the **user-item graph**
- **user-item graph** : interactions & users' opinions (rate score)

➢ General calculation

$$\mathbf{h}_i^I = \sigma(\mathbf{W} \cdot \text{Aggre}_{\text{items}}(\{\mathbf{x}_{ia}, \forall a \in C(i)\}) + \mathbf{b})$$

$$\mathbf{x}_{ia} = g_v([\mathbf{q}_a \oplus \mathbf{e}_r])$$

➢ Mean Aggregation

$$\mathbf{h}_i^I = \sigma\left(\mathbf{W} \cdot \left\{\sum_{a \in C(i)} \alpha_i \mathbf{x}_{ia}\right\} + \mathbf{b}\right) \text{ where } \alpha_i \text{ is fixed to } \frac{1}{|C(i)|}$$

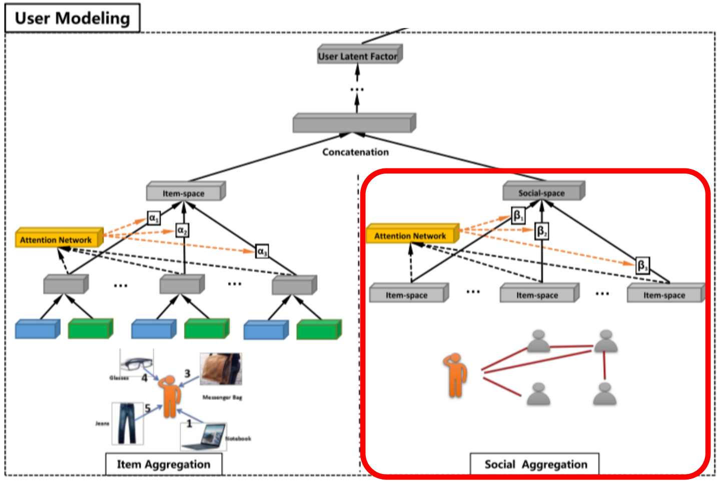
➢ Attention mechanism

$$\alpha_{ia}^* = \mathbf{w}_2^T \cdot \sigma(\mathbf{W}_1 \cdot [\mathbf{x}_{ia} \oplus \mathbf{p}_i] + \mathbf{b}_1) + b_2$$

$$\alpha_{ia} = \frac{\exp(\alpha_{ia}^*)}{\sum_{a \in C(i)} \exp(\alpha_{ia}^*)}$$

User Modeling

- **Aim**: user latent factors: $\boldsymbol{h}_i \in \mathbb{R}^d$ for user u_i
- **Two aggregation**: Item Aggregation & Social Aggregation



✓ Social Aggregation

(2) Social Aggregation

- To learn social-space user latent factor $\boldsymbol{h}_i^S \in \mathbb{R}^d$ from the **social graph**

➢ To aggregate the item-space user latent factors of u_i's neighbor users

$$\mathbf{h}_i^S = \sigma(\mathbf{W} \cdot \text{Aggre}_{\text{neigbhors}}(\{\mathbf{h}_o^I, \forall o \in N(i)\}) + \mathbf{b})$$

➢ Mean Aggregation

$$\mathbf{h}_i^S = \sigma\left(\mathbf{W} \cdot \left\{\sum_{o \in N(i)} \beta_i \mathbf{h}_o^I\right\} + \mathbf{b}\right) \text{ where } \beta_i \text{ is fixed to } \frac{1}{|N(i)|}$$

➢ Attention mechanism

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

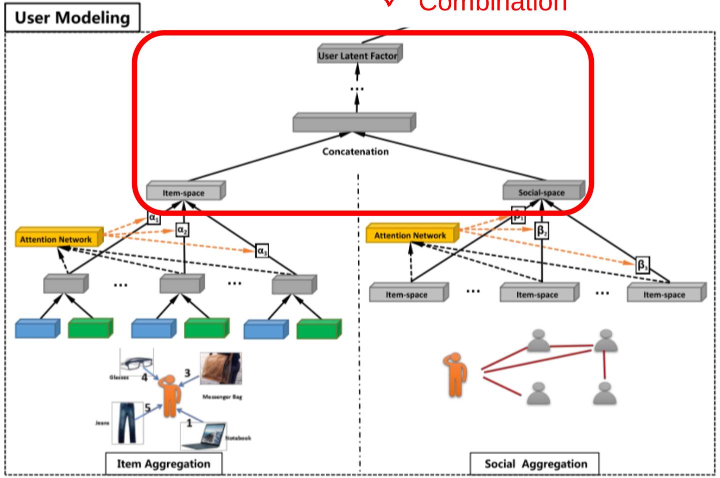
$$\beta_{io}^* = \mathbf{w}_2^T \cdot \sigma(\mathbf{W}_1 \cdot [\mathbf{h}_o^I \oplus \mathbf{p}_i] + \mathbf{b}_1) + b_2$$

$$\beta_{io} = \frac{\exp(\beta_{io}^*)}{\sum_{o \in N(i)} \exp(\beta_{io}^*)}$$

User Modeling

- **Aim**: user latent factors: $\boldsymbol{h}_i \in \mathbb{R}^d$ for user u_i

✓ Combination



Learning User Latent Factor

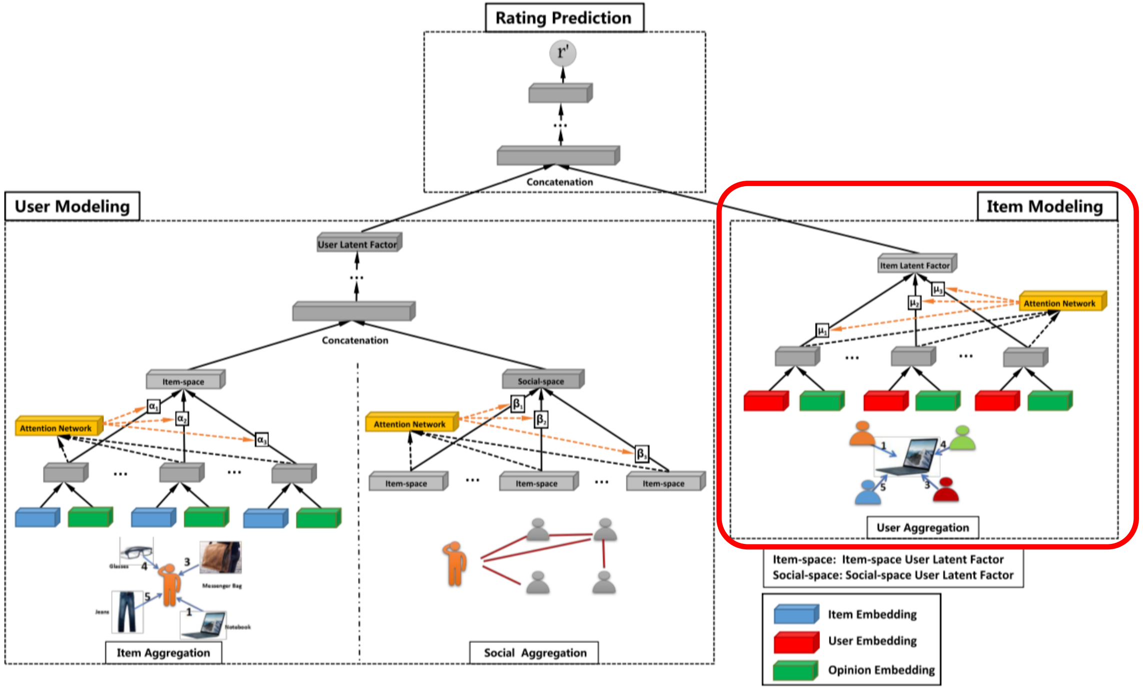
- Combine two factors $\boldsymbol{h}_i^I$ and $\boldsymbol{h}_i^S$ to the final user latent factor via a standard MLP

➢ Calculation

$$
\begin{aligned}
\mathbf{c}_1 &= \left[\mathbf{h}_i^I \oplus \mathbf{h}_i^S\right] \\
\mathbf{c}_2 &= \sigma(\mathbf{W}_2 \cdot \mathbf{c}_1 + \mathbf{b}_2) \\
&\cdots \\
\mathbf{h}_i &= \sigma(\mathbf{W}_l \cdot \mathbf{c}_{l-1} + \mathbf{b}_l)
\end{aligned}
$$

where l is the index of a hidden layer.

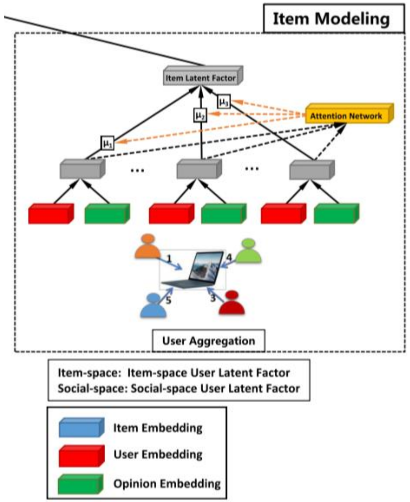
Methodology Overview



✓ Item Modeling

Item Modeling

- **Aim**: item latent factors: $\boldsymbol{z}_j \in \mathbb{R}^d$ for item v_j
- **Aggregation**: User Aggregation



User Aggregation

- For each item v_j, we need to aggregate information from the set of users who have interacted with v_j, , denoted as $B(j)$

➢ Denote: opinion-aware interaction user representation $\mathbf{f}_{jt}$
basic user embedding $\mathbf{p}_t$
opinion embedding $\mathbf{e}_r$ via a MLP $\boldsymbol{g}_u$

$$\mathbf{f}_{jt} = g_u([\mathbf{p}_t \oplus \mathbf{e}_r])$$

➢ To learn item latent factor $\boldsymbol{z}_j$

$$\mathbf{z}_j = \sigma(\mathbf{W} \cdot \text{ Aggre}_{\text{users}}(\{\mathbf{f}_{jt}, \forall t \in B(j)\}) + \mathbf{b})$$

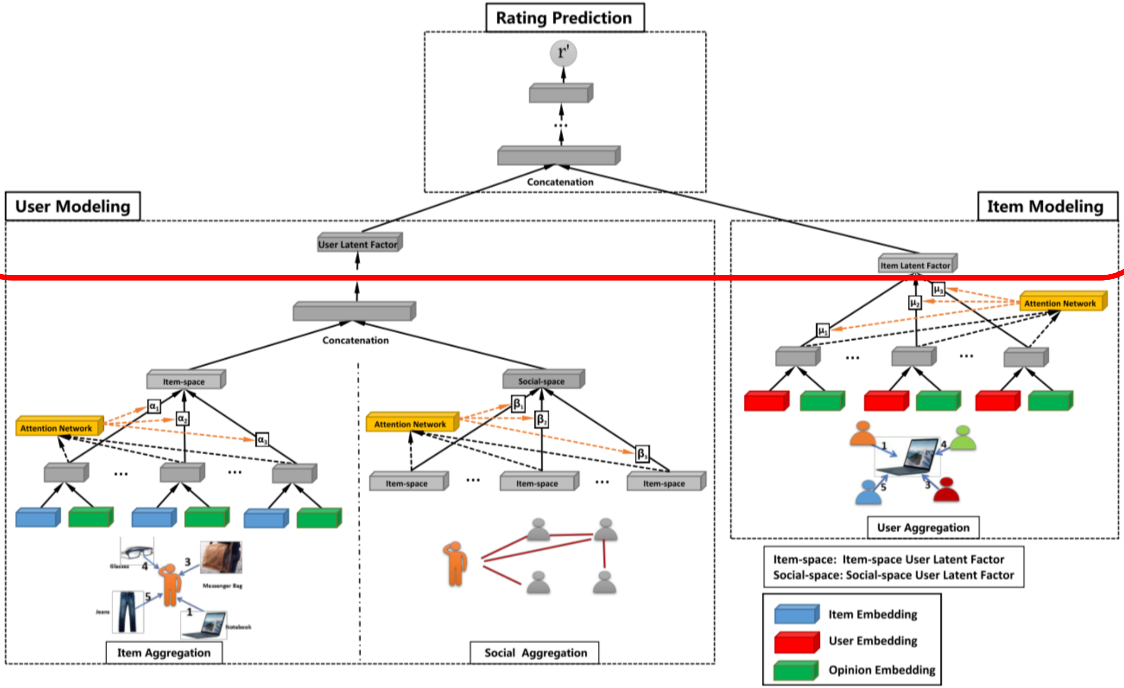
➢ Attention mechanism to differentiate users

$$\mathbf{z}_j = \sigma\left(\mathbf{W} \cdot \left\{\sum_{t \in B(j)} \mu_{jt}\mathbf{f}_{jt}\right\} + \mathbf{b}\right)$$

$$\mu_{jt}^* = \mathbf{w}_2^T \cdot \sigma(\mathbf{W}_1 \cdot [\mathbf{f}_{jt} \oplus \mathbf{q}_j] + \mathbf{b}_1) + b_2$$

$$\mu_{jt} = \frac{\exp(\mu_{jt}^*)}{\sum_{t \in B_{(j)}} \exp(\mu_{jt}^*)}$$

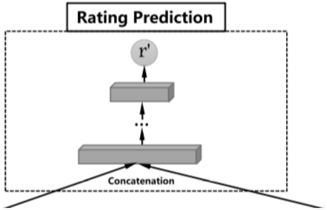
Methodology Overview



✓ Rating Prediction

Rating Prediction & Model Training

Rating Prediction



- Predict with the two latent factors: $\boldsymbol{h}_i$ and $\boldsymbol{z}_j$

- Concatenation & MLP

$$
\begin{aligned}
\mathbf{g}_1 &= [\mathbf{h}_i \oplus \mathbf{z}_j] \\
\mathbf{g}_2 &= \sigma(\mathbf{W}_2 \cdot \mathbf{g}_1 + \mathbf{b}_2) \\
&\cdots \\
\mathbf{g}_{l-1} &= \sigma(\mathbf{W}_l \cdot \mathbf{g}_{l-1} + \mathbf{b}_l) \\
r'_{ij} &= \mathbf{w}^T \cdot \mathbf{g}_{l-1}
\end{aligned}
$$

Model Training

- Objective function

$$\text{Loss} = \frac{1}{2|O|} \sum_{i,j \in O} \left(r'_{ij} - r_{ij}\right)^2$$

- Optimizer: RMSprop
- Randomly Initialization
- Dropout strategy

Experiment Settings

- Datasets
  - Social Network Websites: **Ciao, Epinions**.
  - Both allow users to rate items, browse/write reviews and add friends.

Table 2: Statistics of the datasets

Dataset	Ciao	Epinions
# of Users	7,317	18,088
# of Items	10,4975	261,649
# of Ratings	283,319	764,352
# of Density (Ratings)	0.0368%	0.0161%
# of Social Connections	111,781	355,813
# of Density (Social Relations)	0.2087%	0.1087%

- Evaluation Metrics
  - **MAE**(Mean Absolute Error):

$$loss = \sum_{i=1}^{n} |y_i - \tilde{y}_i|$$

  - **RMSE**(Root Mean Square Error)

$$loss = \sum_{i=1}^{n} \sqrt{y_i^2 - \tilde{y}_i^2}$$

- Comparison Baselines
  - **Group1: Traditional RS w/o Social Network**
  - **Group2: Traditional RS with Social Network**
  - **Group3: Deep Neural Network RS w/o Social Network**
  - **Group4: Deep Neural Network RS with Social Network**

	Group 1	Group 2	Group 3	Group 4
Models	PMF	SoRec	NeuMF	DeepSoR
		SoReg		GCMC+SN
		Social MF		GraphRec (Proposed)
		Trust MF		

Comparison Results

Observation 1: Group 2 always outperform Group 1 to verify the effectiveness of social network info.

Observation 2: Group 3 > Group 1 && Group 4 > Group2 to verify the power of deep neural network

Observation 3: Among the baseline, GCMC+SN shows strong perf., which implies the power of GNN

Observation 4: Proposed GraphRec is the best because of the intro. of interactions and opinions in user-item graph

Table 3: Performance comparison of different recommender systems

Training	Metrics	Algorithms								
		PMF	SoRec	SoReg	SocialMF	TrustMF	NeuMF	DeepSoR	GCMC+SN	**GraphRec**
Ciao (60%)	MAE	0.952	0.8489	0.8987	0.8353	0.7681	0.8251	0.7813	0.7697	0.7540
	RMSE	1.1967	1.0738	1.0947	1.0592	1.0543	1.0824	1.0437	1.0221	1.0093
Ciao (80%)	MAE	0.9021	0.8410	0.8611	0.8270	0.7690	0.8062	0.7739	0.7526	0.7387
	RMSE	1.1238	1.0652	1.0848	1.0501	1.0479	1.0617	1.0316	0.9931	0.9794
Epinions (60%)	MAE	1.0211	0.9086	0.9412	0.8965	0.8550	0.9097	0.8520	0.8602	0.8441
	RMSE	1.2739	1.1563	1.1936	1.1410	1.1505	1.1645	1.1135	1.1004	1.0878
Epinions (80%)	MAE	0.9952	0.8961	0.9119	0.8837	0.8410	0.9072	0.8383	0.8590	0.8168
	RMSE	1.2128	1.1437	1.1703	1.1328	1.1395	1.1476	1.0972	1.0711	1.0631

60%: 60% data in training set
80%: 80% data in training set

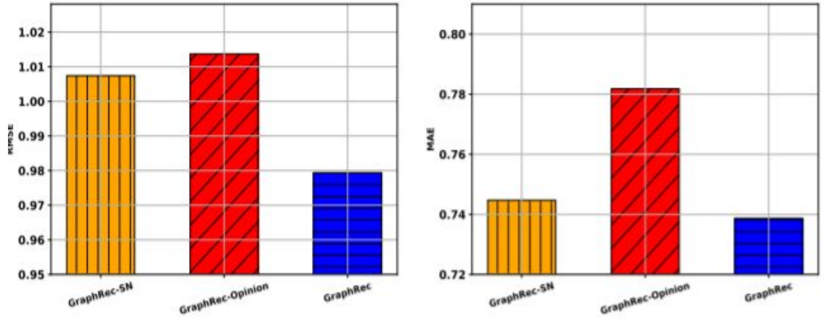
Model Analysis

Several A/B tests to verify the impact of different model components and hyper-parameters

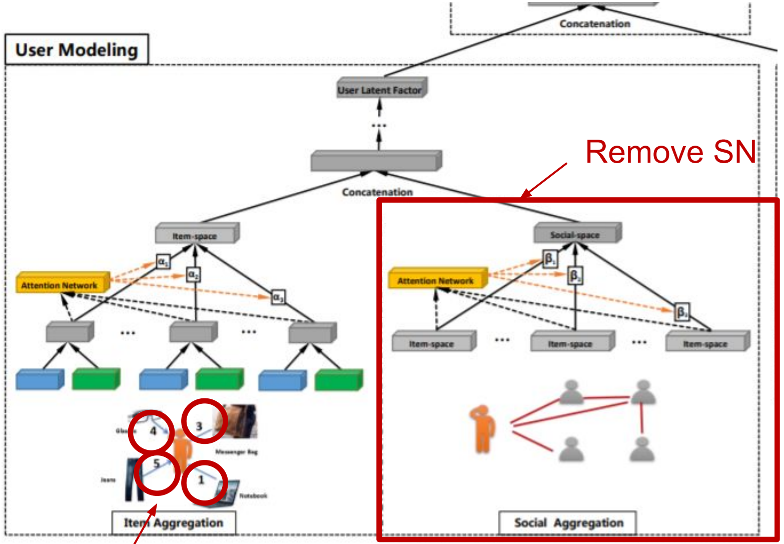
Methodology: Remove the test components to observe the change of performance.

Analysis 1 : Effect of Social Network and User Opinion

- only remove social network info: GraphRec-SN
- only remove interaction opinions: GraphRec-Opinion



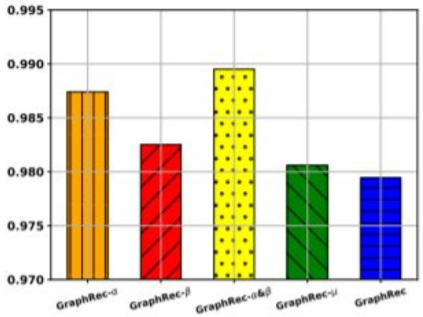
(a) Ciao-RMSE (b) Ciao-MAE

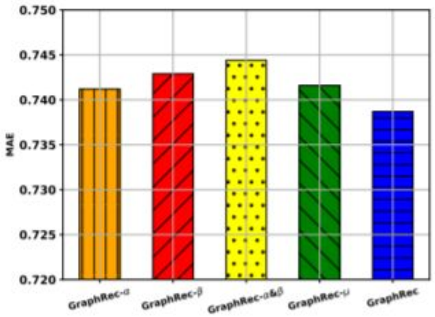


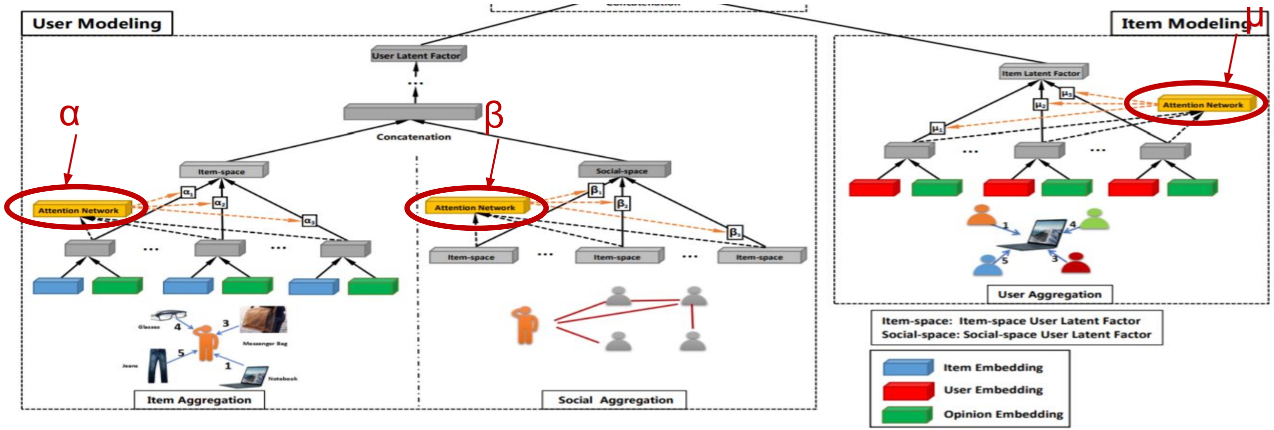
Model Analysis

Analysis 2 : Effect of Attention Mechanisms

- Only remove item attention: GraphRec-α
- Only remove social attention: GraphRec-β
- Both remove α&β: GraphRec- α&β
- only remove user attention: GraphRec-μ



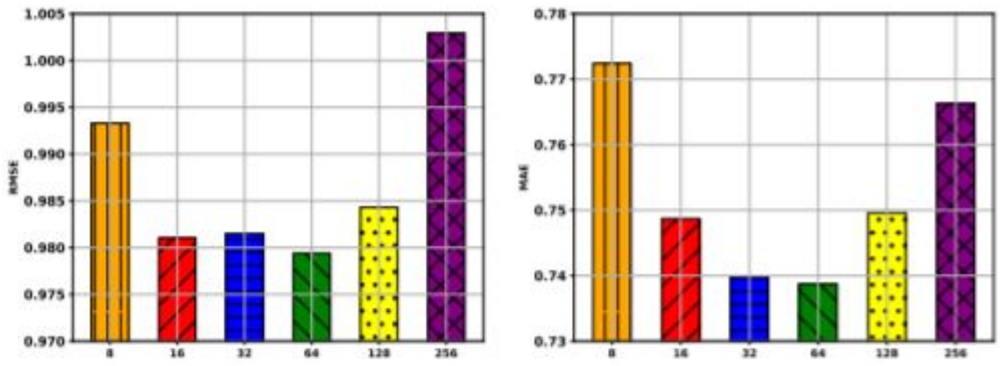




Model Analysis

Analysis 3 : Effect of Embedding Size

- Test the performance change with various embedding size {8, 16, 32, 64, 128, 256}
- Larger embedding size(8->64) will decrease the loss but increase the computation complexity.



(a) Ciao-RMSE (b) Ciao-MAE

UVA ENGINEERING

Summary

Strength:

- Leverage the graph topological info and GNN
- Integrated user-item info, social info, rating info in the neural network
- Introduction of attention mechanism to obtain the various contribution weights
- Elaborative comparison experiments and A/B tests for model components

Weakness:

- Lack of some theatrical proof and derivative
- Model Computation Complexity ?
- Evaluation Metric is good enough ?

Q & A