Incorporating Brand into E-commerce RS with GNN

Gao Zhen

2021/10/25

Outline

- Introduction
- Background
  - GNN Framework
  - GCN/GraphSAGE/GAT
  - NGCF
- Related Work
  - PUP
  - HiGNN
  - HAN
- Future Work

GNN in E-commerce RS

- **Social Recommendation**
  - User interactions are affected by both their own preferences and the social factor.
- **Sequential Recommendation**
  - Given a user's historical sequence, RS aims to predict the next item.
- **Session-based Recommendation**
  - Given a anonymous short session, RS aims to predict the next item.
- **Bundle Recommendation**
- **Multi-behavior Recommendation**

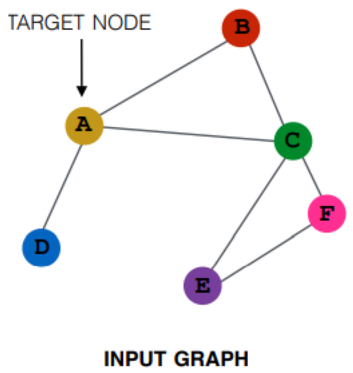
[1] Graph Neural Networks for Recommender Systems: Challenges, Methods, and Directions. 2021

Evaluation Measures in RS

- Rating and Usage Prediction Accuracy
  - Precision、Recall、F1
  - Hit Rate
  - Mean Absolute Error(MAE) 、Root Mean Squared Error(RMSE)
- Ranking
  - Normalized Discounted Cumulative Gain (NDCG)
  - Receiver Operating Characteristic(ROC)、Area under curve(AUC)
  - Mean Reciprocal Rank(MRR)
- Online
  - Click Through Rate(CTR)

[2] A Survey on Recommender System. IJCSIS 2016

GNN Framework



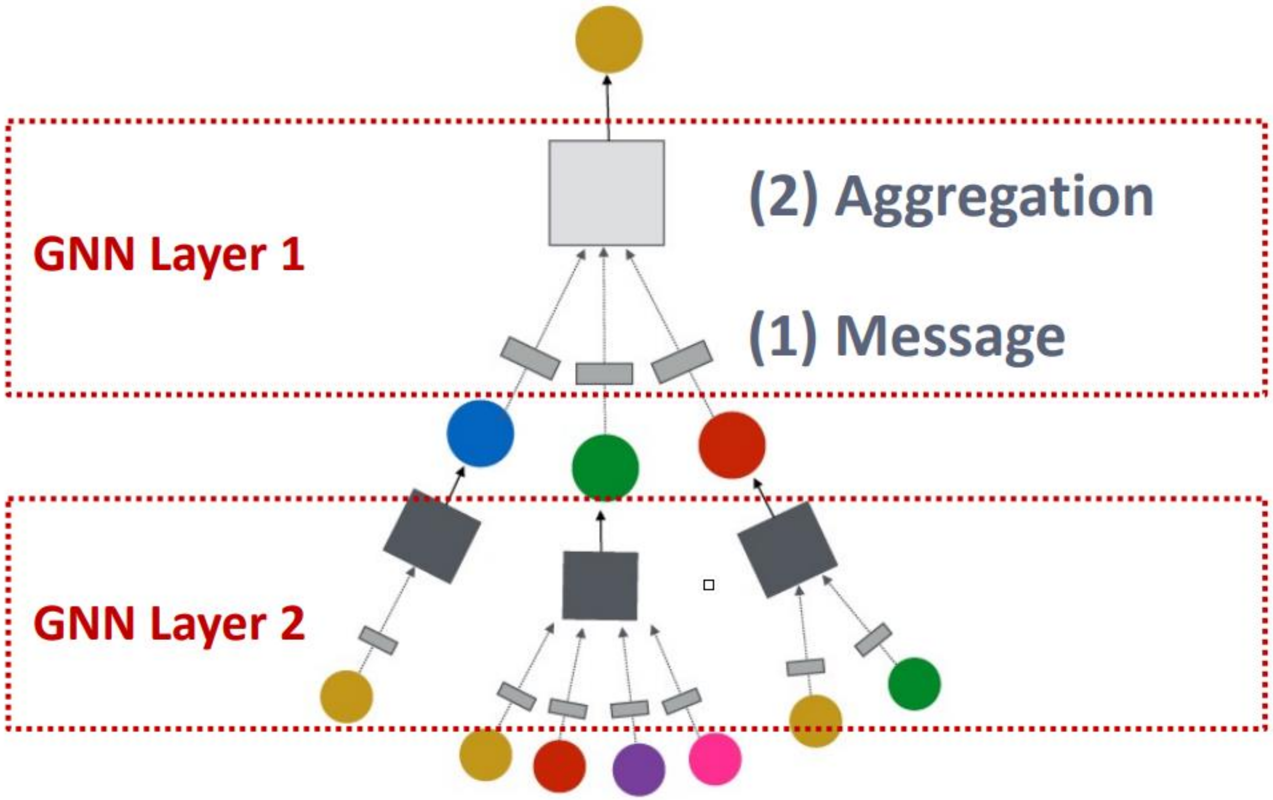
- Message computation

$$\mathbf{m}_u^{(l)} = \mathrm{MSG}^{(l)}\left(\mathbf{h}_u^{(l-1)}\right)$$

- Message aggregation

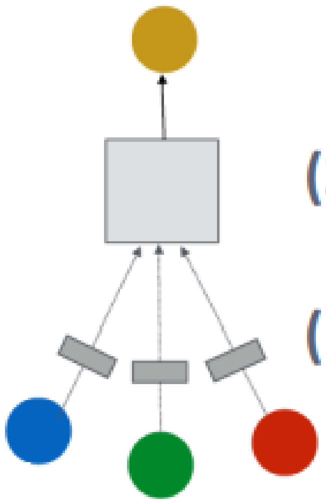
$$\mathbf{h}_v^{(l)} = \mathrm{AGG}^{(l)}\left(\left\{\mathbf{m}_u^{(l)}, u \in N(v)\right\}, \mathbf{m}_v^{(l)}\right)$$

- Nonlinearity activation



Graph Convolutional Networks(GCN)

$$\mathbf{h}_v^{(l)} = \sigma\left(\underbrace{\sum_{u \in N(v)}}_{\text{Aggregation}} \overbrace{\mathbf{W}^{(l)} \frac{\mathbf{h}_u^{(l-1)}}{|N(v)|}}^{\text{Message}}\right)$$

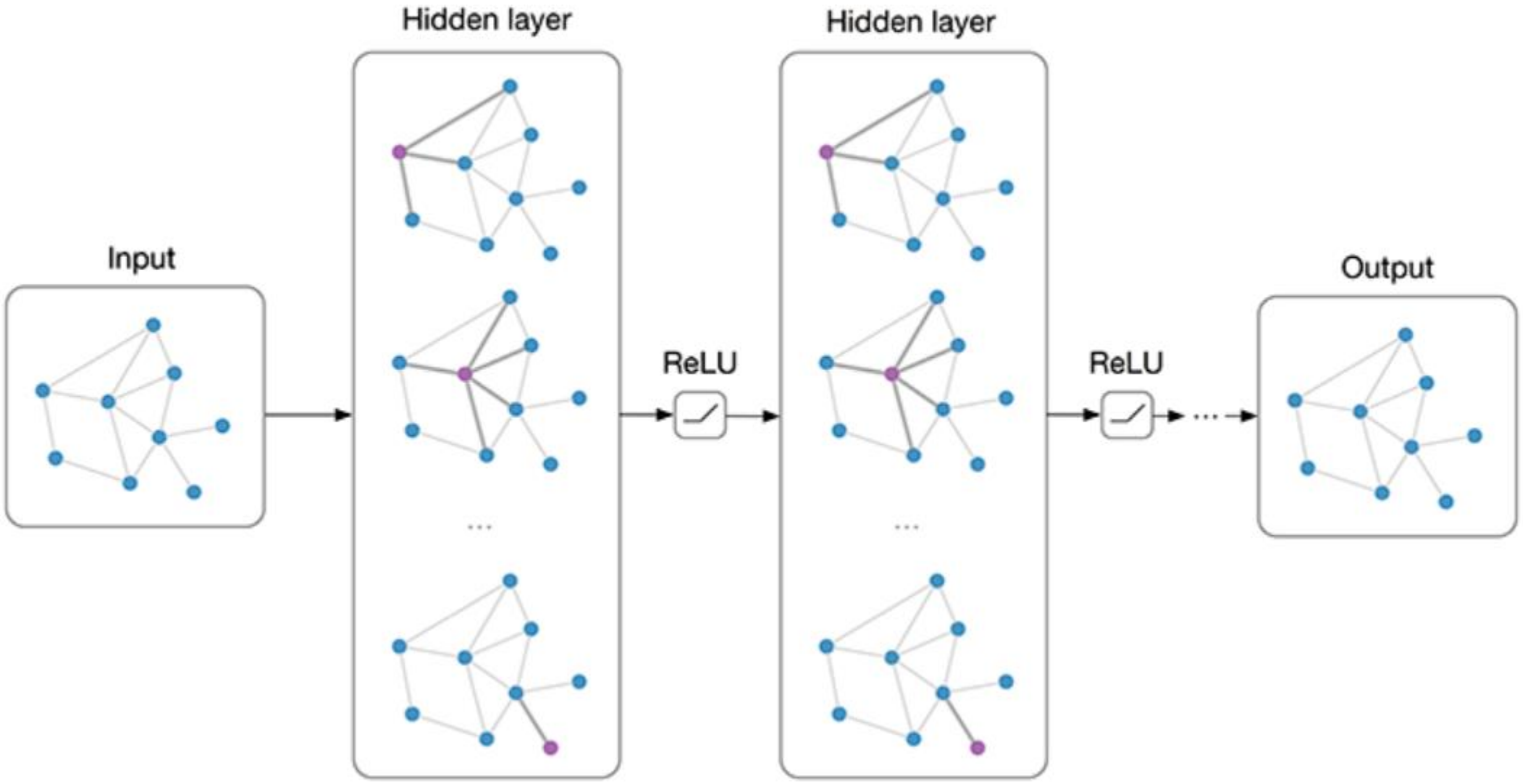


- **Message**:
  - Each Neighbor: $\mathbf{m}_u^{(l)} = \frac{1}{|N(v)|}\mathbf{W}^{(l)}\mathbf{h}_u^{(l-1)}$
  - Normalized by node degree
- **Aggregation**:
  - **Sum** messages from neighbors, then apply activation
  - $\mathbf{h}_v^{(l)} = \sigma\left(\mathrm{Sum}\left(\left\{\mathbf{m}_u^{(l)}, u \in N(v)\right\}\right)\right)$

[3] Semi-Supervised Classification with Graph Convolutional Networks. ICLR 2017

GCN

- $f(H^{(l)}, A) = \sigma\left(AH^{(l)}W^{(l)}\right)$



- Limitations:
  - No self-loops
  - Not normalized
- Solutions:
  - $\hat{A} = A + I$
  - Use a symmetric normalization

$$f(H^{(l)}, A) = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right)$$

[3] Semi-Supervised Classification with Graph Convolutional Networks. ICLR 2017

Graph Sampling and aggregation(GraphSAGE)

$$\mathbf{h}_v^{(l)} = \sigma\left(\mathbf{W}^{(l)} \cdot \text{CONCAT}\left(\mathbf{h}_v^{(l-1)}, \text{AGG}\left(\left\{\mathbf{h}_u^{(l-1)}, \forall u \in N(v)\right\}\right)\right)\right)$$

- Message is computed within the AGG(·)
  - AGG(·)： Mean/Pool/LSTM
- Two-stage aggregation
  - Stage 1： Aggregate from node neighbors

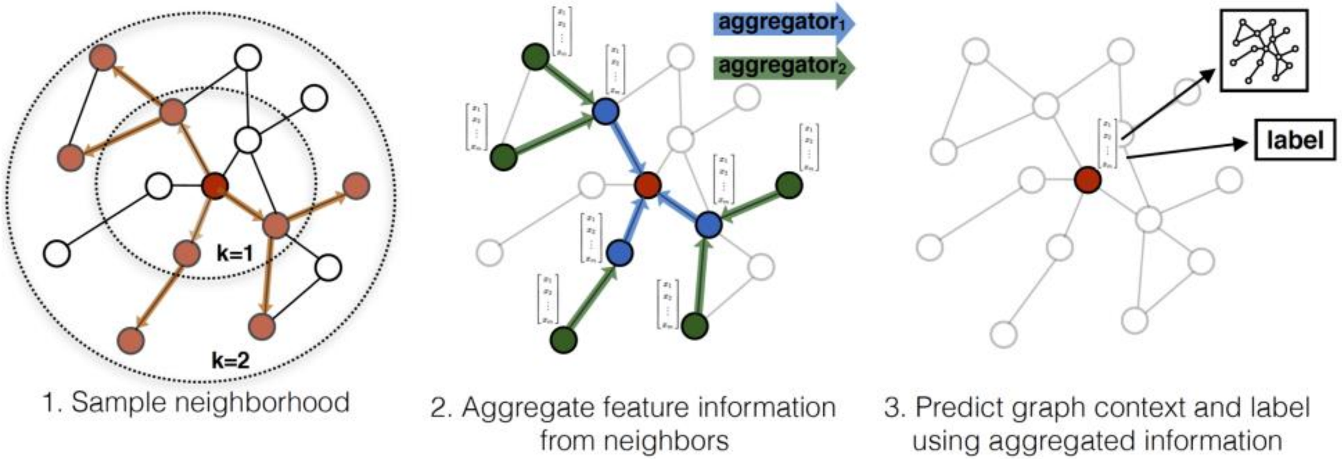
$$\mathbf{h}_{N(v)}^{(l)} \leftarrow \text{AGG}\left(\left\{\mathbf{h}_u^{(l-1)}, \forall u \in N(v)\right\}\right)$$

  - Stage 2： Further aggregate over the node itself

$$\mathbf{h}_v^{(l)} \leftarrow \sigma\left(\mathbf{W}^{(l)} \cdot \text{CONCAT}(\mathbf{h}_v^{(l-1)}, \mathbf{h}_{N(v)}^{(l)})\right)$$

[4] Inductive representation learning on large graphs. ICLR 2017

GraphSAGE VS GCN



- GCN is transductive and can only generate embeddings for a single fixed graph, does not efficiently generalize to unseen nodes
- GraphSAGE is an inductive framework that leverages node attribute information to efficiently generate representations on previously unseen data.
- GraphSAGE uses multiple aggregators rather than the simple convolution in GCN

[4] Inductive representation learning on large graphs. ICLR 2017

Graph Attention Networks(GAT)

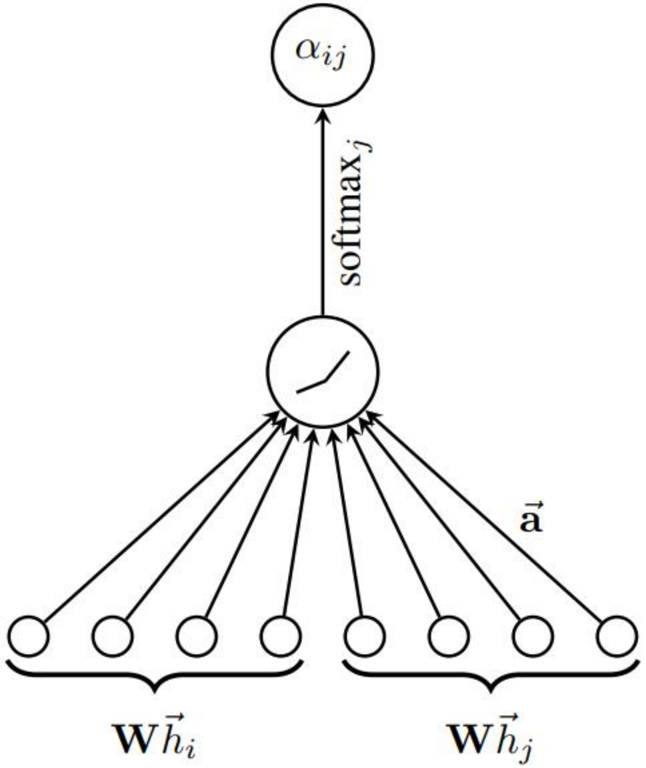
$$\mathbf{h}_v^{(l)} = \sigma(\textstyle\sum_{u \in N(v)} \boxed{\alpha_{vu}} \mathbf{W}^{(l)} \mathbf{h}_u^{(l-1)})$$

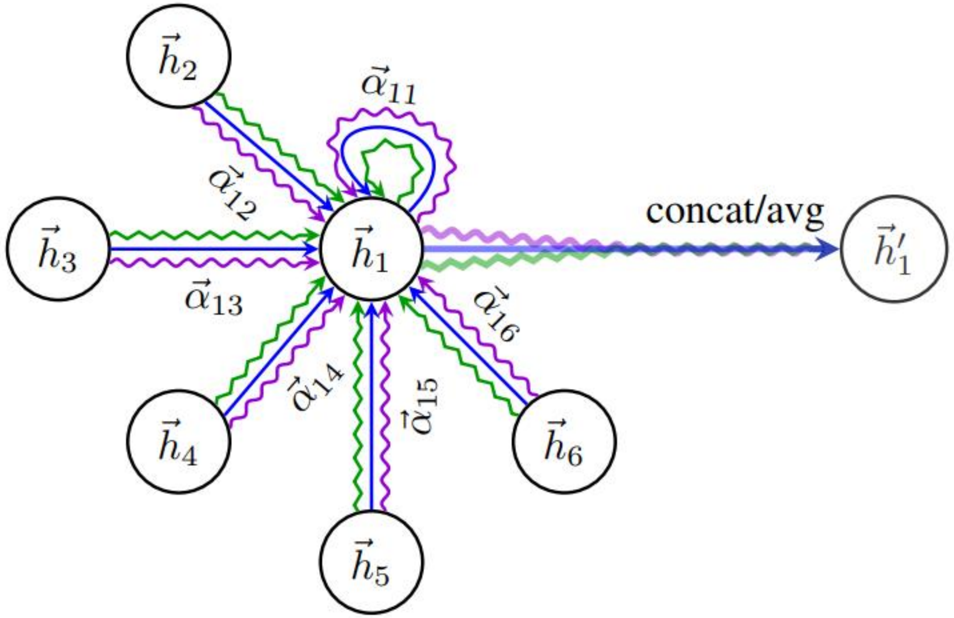
Attention weights

- In GCN/GraphSAGE
  - $\alpha_{vu} = \frac{1}{|N(v)|}$
- Attention mechanism $\boldsymbol{a}$ have trainable parameters
- Attention coefficients $e_{vu} = a(\mathbf{W}^{(l)} \mathbf{h}_u^{(l-1)}, \mathbf{W}^{(l)} \boldsymbol{h}_v^{(l-1)})$
- Attention weight $\alpha_{vu} = \frac{\exp(e_{vu})}{\sum_{k \in N(v)} \exp(e_{vk})}$

[5] Graph Attention Networks. ICLR 2018

GAT





$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)}$$

$$\vec{h}'_i = \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W}\vec{h}_j\right)$$

Neural graph collaborative filtering(NGCF)

- Existing CF methods don’t model high-order connectivity explicitly
  - Embedding function only considers descriptive features (e.g., ID, attributes)
  - User-item interactions are not considered

- CF modeling with high-order connectivity via GNN
  - Embedding Propagation, inspired by GNNs

[6] Neural graph collaborative filtering. SIGIR 2019

NGCF

- First-order Propagation
  - Message Construction： generate message from one neighbor

message passed from i to u

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

discount factor

- message dependent on the affinity, distinct from GCN, GraphSage, etc.
- Pass more information to similar nodes

  - Message Aggregation：

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

self-connections

all neighbors of u

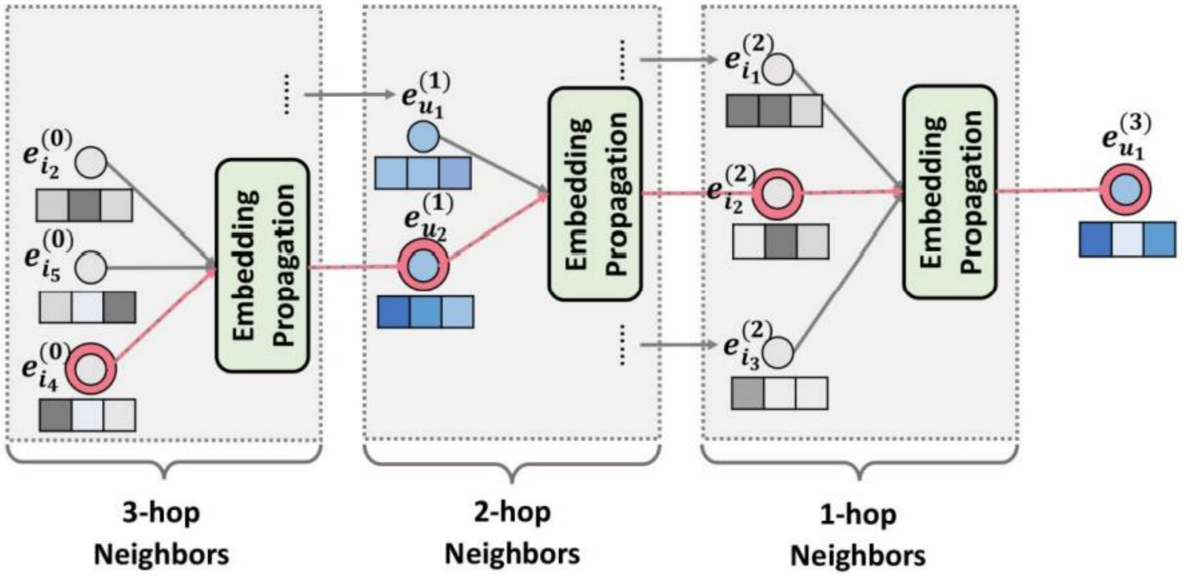
NGCF

- High-order Propagation：stack more embedding propagation layers

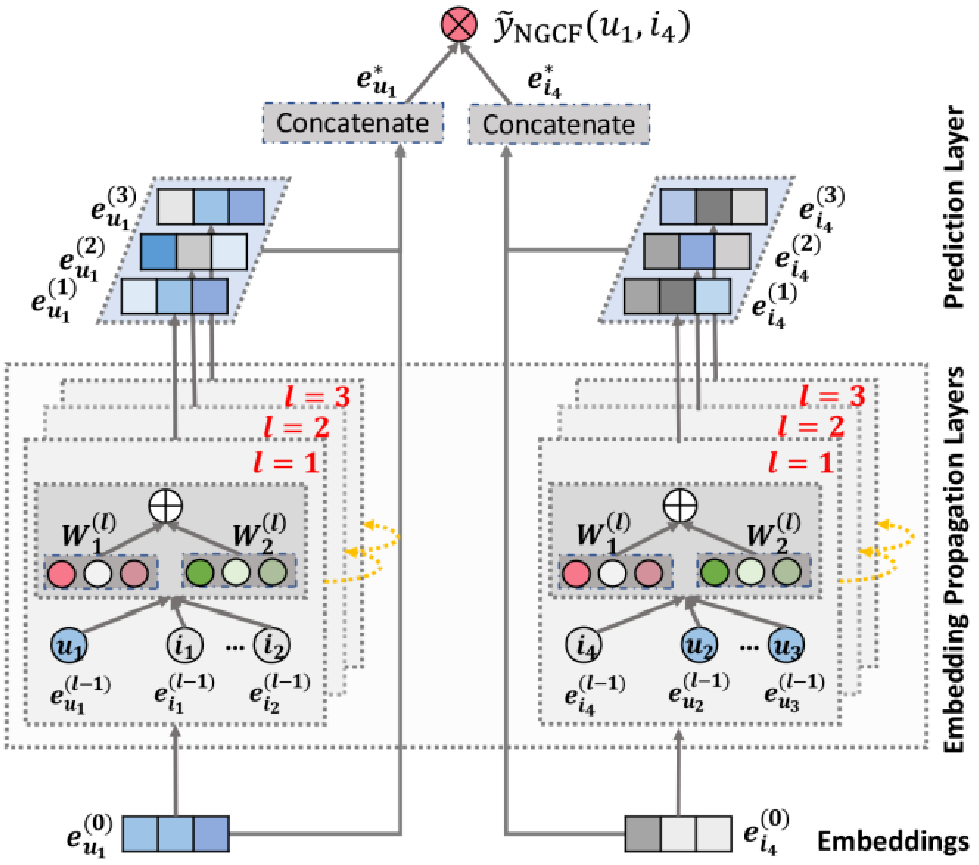
$$\mathbf{e}_u^{(l)} = \text{LeakyReLU}\Big(\mathbf{m}_{u\leftarrow u}^{(l)} + \sum_{i\in\mathcal{N}_u} \mathbf{m}_{u\leftarrow i}^{(l)}\Big),$$

representation of u at the l-th layer

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



NGCF



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

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

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

[6] Neural graph collaborative filtering. SIGIR 2019

Price-aware User Preference modeling(PUP)

- Motivation
  - Price is a significant factor in affecting user behaviors and product sales in marketing research. Nevertheless, and surprisingly, it has received relatively little scrutiny in recommendation.

- Difficulties and Solutions
  - Unstated price awareness
    - Infer a user's personalized awareness on item price from her purchase history via GCN
  - Category-dependent
    - Squeeze out the two important attributes (price and category) as entity nodes to capture the category-dependent price awareness

[7] Price-aware recommendation with graph convolutional networks. ICDE 2020

Category-dependent Price Awareness

- *category willing to pay* (CWTP)
    - the highest price a given user is willing to pay for items of a given category
    - One CWTP per category
    - Compute the entropy of CWTPs for each user

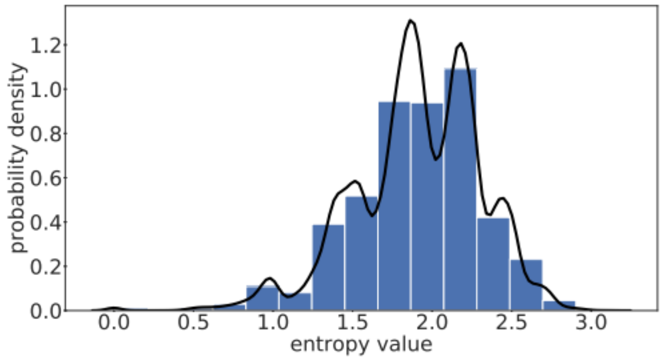


Fig. 1: Histogram of users' CWTP entropy value. High entropy value means users consider price differently in distinct categories.

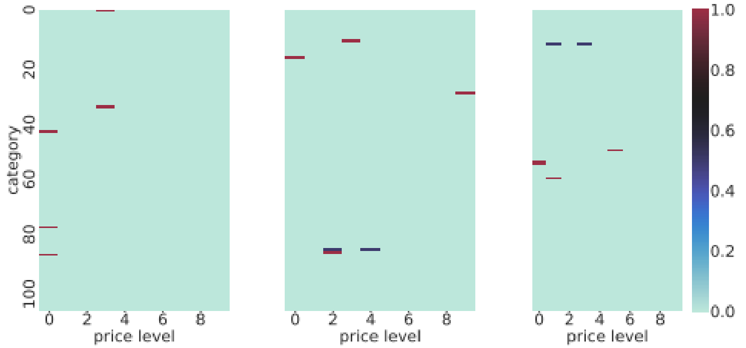
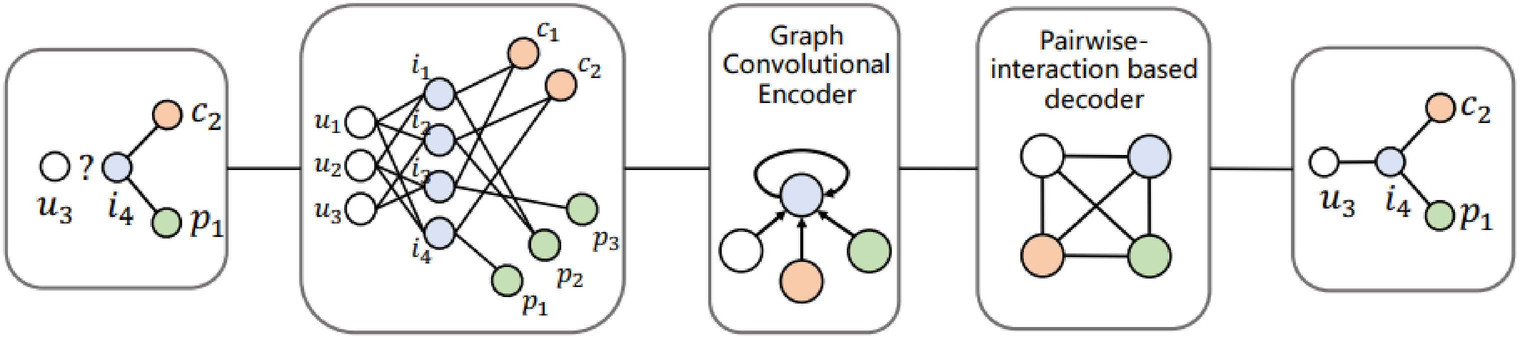


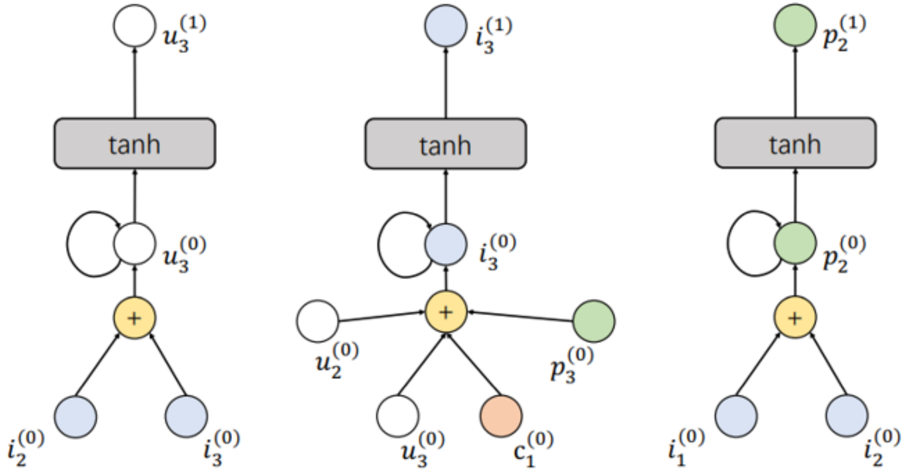
Fig. 2: Price-category purchase heatmap of three randomly selected users

PUP Model



- Four types of nodes in unified heterogeneous graph
  - User/Item/Price(level)/Category
- Graph Convolutional Encoder
- Pairwise-interaction Based Decoder

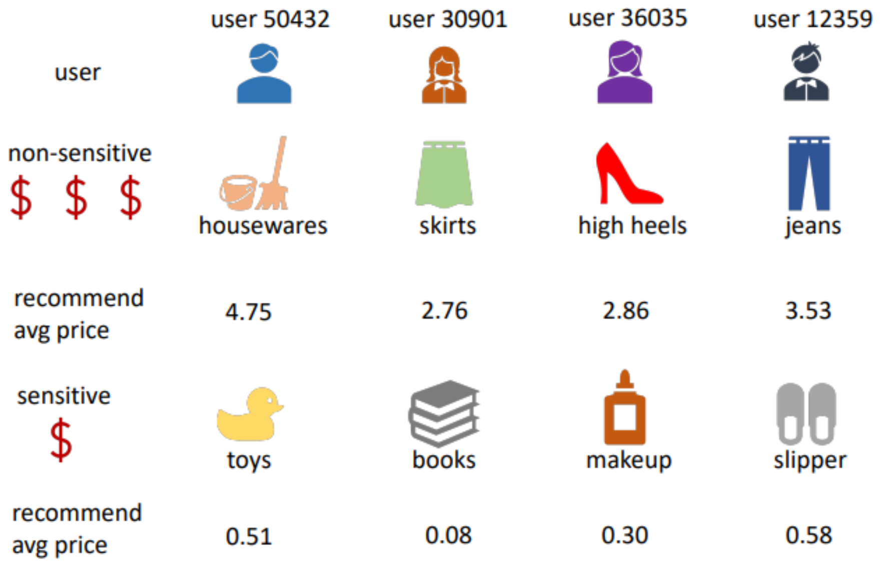
$$s = s_{\text{global}} + \alpha s_{\text{category}}$$
$$s_{\text{global}} = e_u^T e_i + e_u^T e_p + e_i^T e_p$$
$$s_{\text{category}} = e_u^T e_c + e_u^T e_p + e_c^T e_p,$$



Experiments

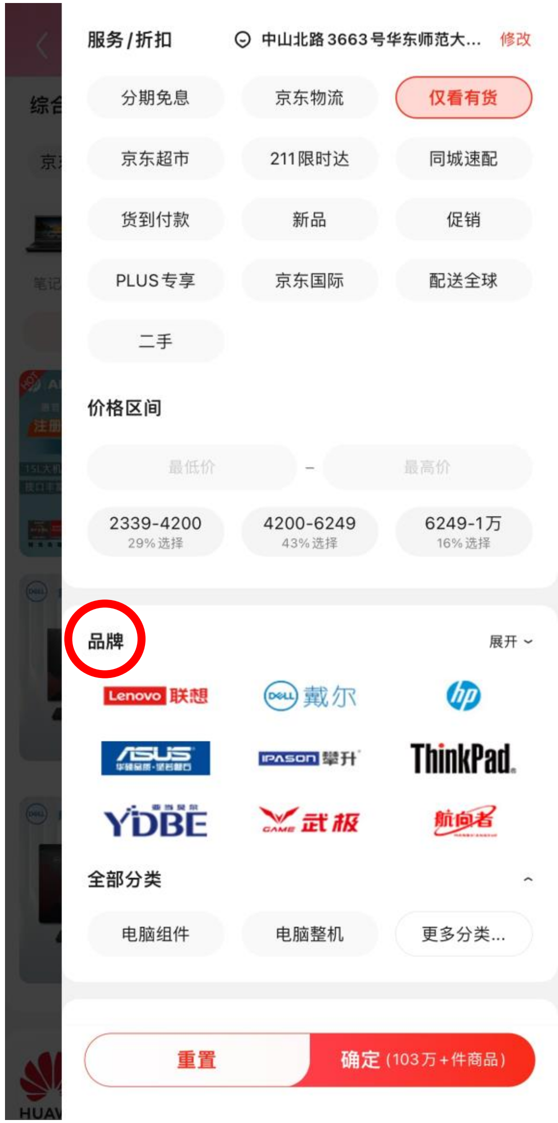
	Yelp dataset				Beibei dataset			
method	Recall@50	NDCG@50	Recall@100	NDCG@100	Recall@50	NDCG@50	Recall@100	NDCG@100
ItemPop	0.0401	0.0182	0.0660	0.0247	0.0087	0.0027	0.0175	0.0046
BPR-MF	0.1621	0.0767	0.2538	0.1000	0.0256	0.0103	0.0379	0.0129
PaDQ	0.1241	0.0572	0.2000	0.0767	0.0131	0.0056	0.0186	0.0068
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%

method	Recall@50	NDCG@50	Recall@100	NDCG@100
PUP w/o c,p	0.0726	0.0211	0.1155	0.0285
PUP w/ c	0.0633	0.0222	0.0944	0.0276
PUP w/ p	0.0854	0.0277	0.1275	0.0350
PUP	0.0890	0.0293	0.1336	0.0370



Inspired by PUP

- Brand also plays a critical role in determining whether the user will make the final purchase decision.
- Filter priority in JD and Taobao:
  - Service/Discount > Price > **Brand** > Category



Hierarchical Bipartite Graph Neural Networks(HiGNN)

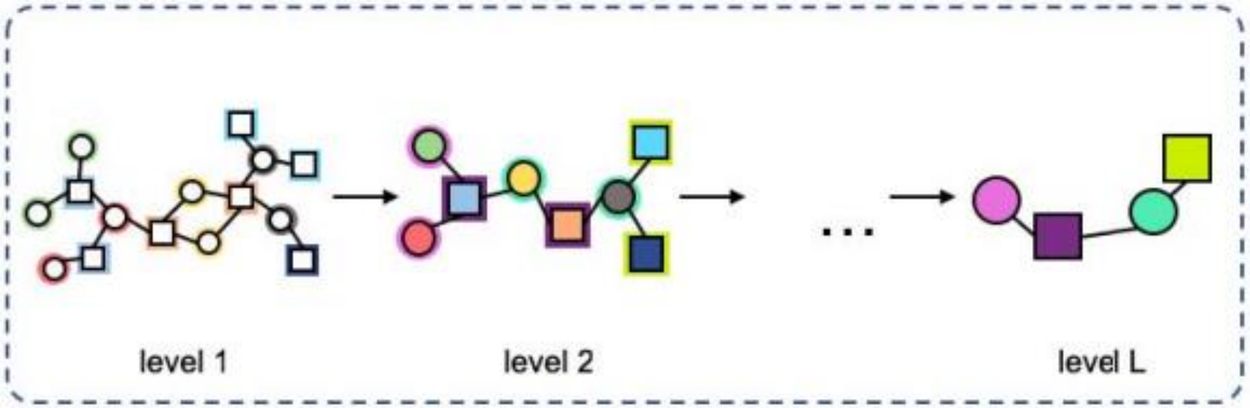
- Introduce bipartite GraphSAGE on a user-item graph
- Quadruple $G = (U, I, E, S)$
  - users $U = \{u_1, u_2, \ldots, u_M\}$
  - items $I = \{i_1, i_2, \ldots, i_N\}$
  - each edge $\{e = (u_m, i_n) | u_m \in U, i_n \in I\}$ is associated with a weight S(e)
- User embedding
  - $\boldsymbol{h}^p_{N(u)} \leftarrow \boldsymbol{M}^u_i \cdot \mathrm{AGGREGATE}^p_u(\{\boldsymbol{h}^{p-1}_i, \forall i \in N(u)\})$
  - $\boldsymbol{h}^p_u \leftarrow \sigma(\boldsymbol{W}^p_u \cdot \mathrm{CONCAT}(\boldsymbol{h}^{p-1}_u, \boldsymbol{h}^p_{N(u)}))$
- Item embedding
  - $\boldsymbol{h}^p_{N(i)} \leftarrow \boldsymbol{M}^i_u \cdot \mathrm{AGGREGATE}^p_i(\{\boldsymbol{h}^{p-1}_u, \forall u \in N(i)\})$
  - $\boldsymbol{h}^p_i \leftarrow \sigma(\boldsymbol{W}^p_i \cdot \mathrm{CONCAT}(\boldsymbol{h}^{p-1}_i, \boldsymbol{h}^p_{N(i)}))$

[8] Hierarchical bipartite graph neural networks: Towards large-scale e-commerce applications. ICDE 2020

Stack Multiple GNN Modules

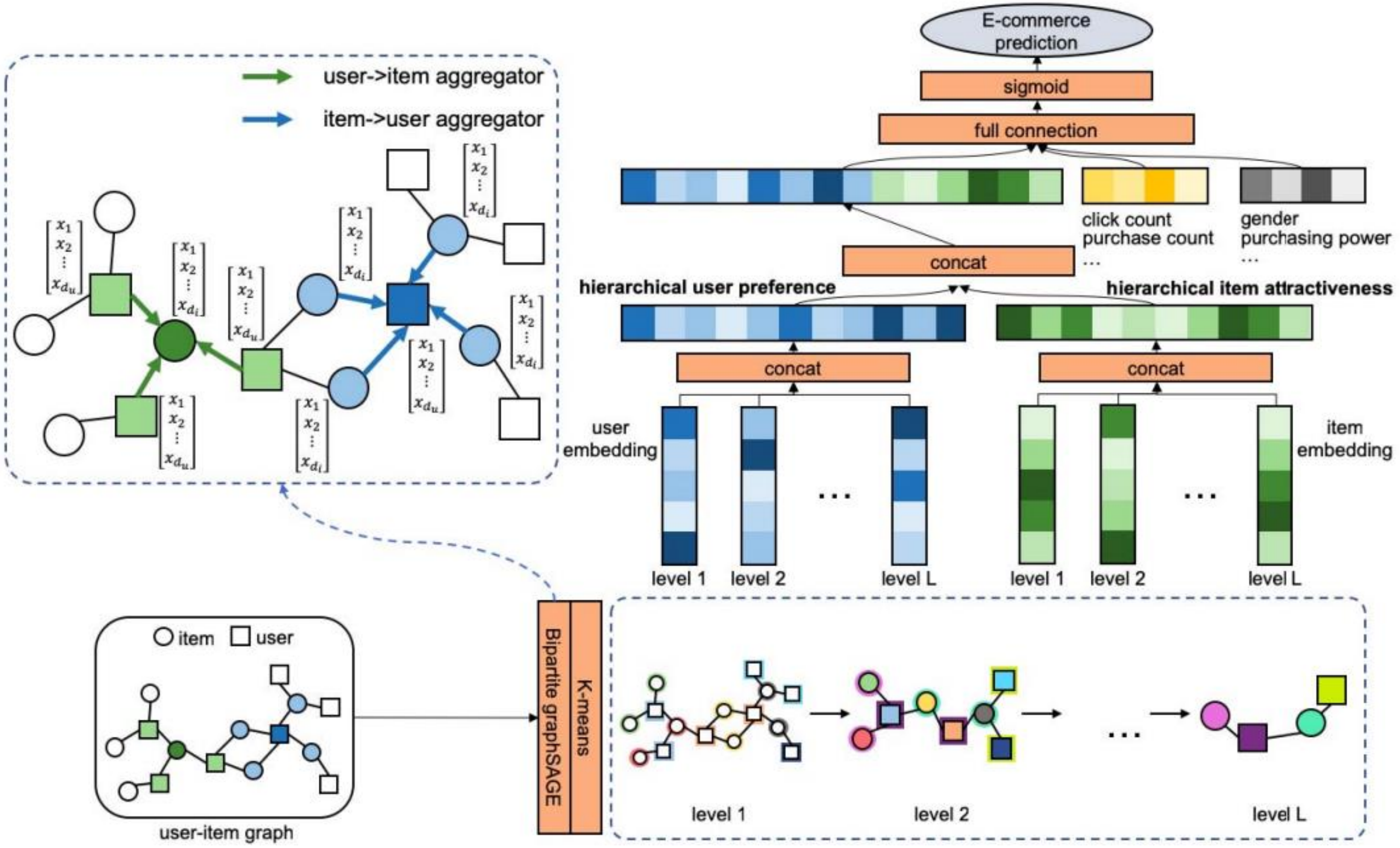
- Set of user embeddings $\boldsymbol{Z}_u = \{\boldsymbol{z}_u, \forall u \in U\}$
- Set of item embeddings $\boldsymbol{Z}_i = \{\boldsymbol{z}_i, \forall i \in I\}$
- Consider user clusters Cu and item clusters Ci clustered by K-means as new users and items in a new coarsened user-item graph.
- edge weight of (Cu, Ci)

$$S(C_u, C_i) = \sum_e S(e), \forall e = (u, i) \in G, u \in C_u, i \in C_i$$

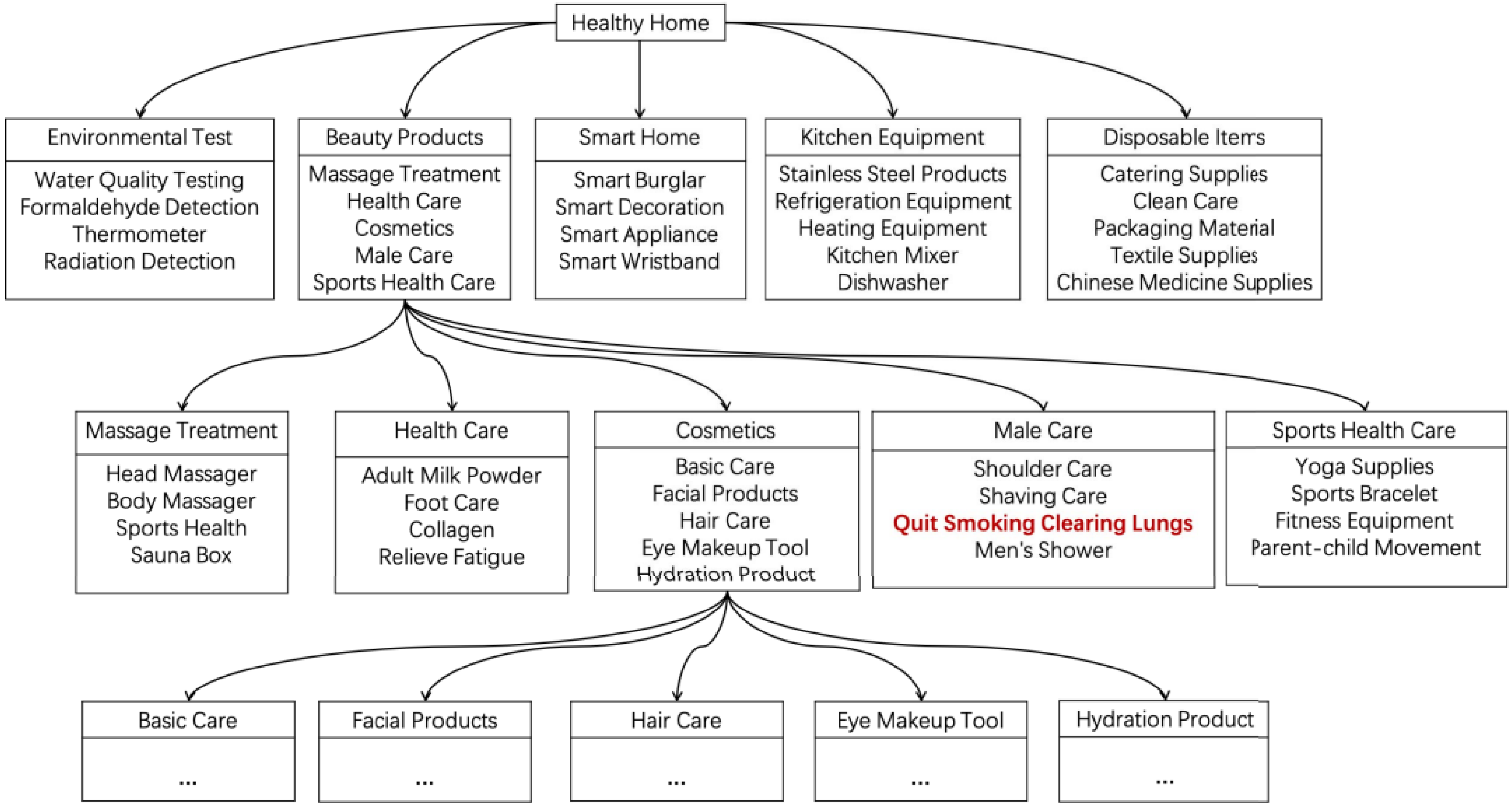


[8] Hierarchical bipartite graph neural networks: Towards large-scale e-commerce applications. ICDE 2020

HiGNN Framework



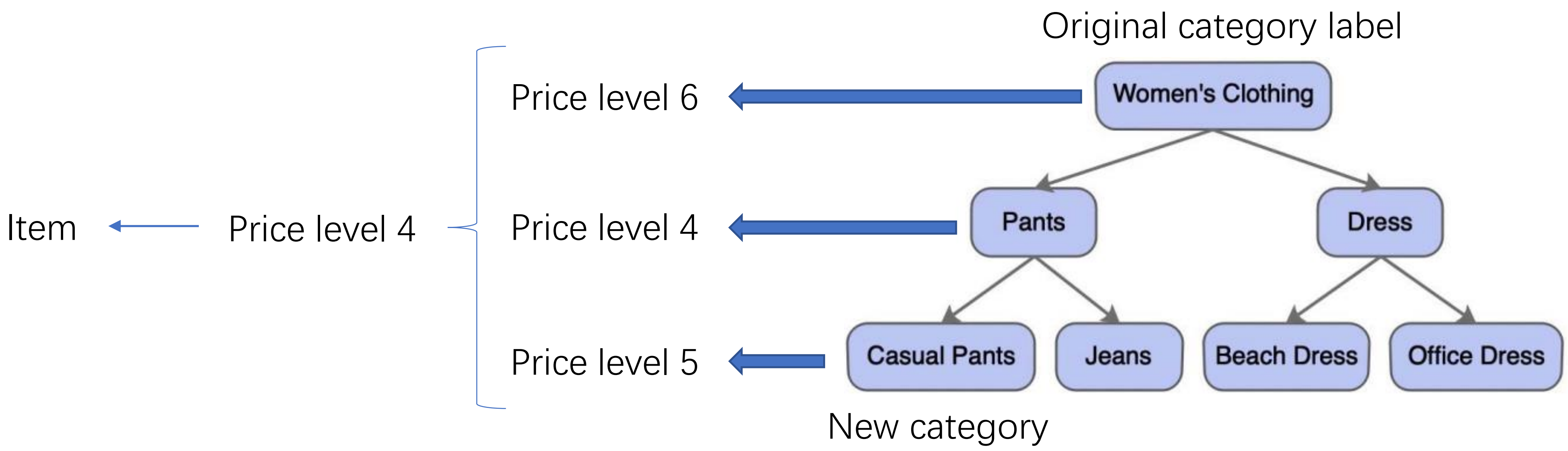
Topic-driven taxonomy



(a) The sub-topics under the topics ‘Healthy Home’, ‘Beauty Products’, and ‘Cosmetics’.

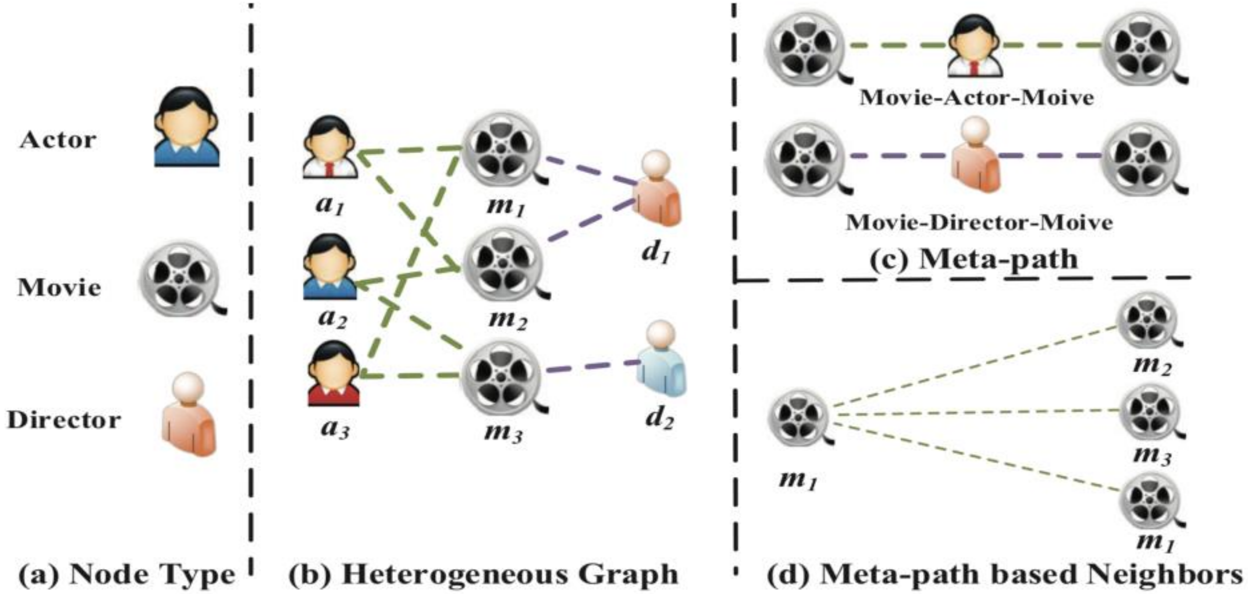
Inspired by HiGNN

- Hierarchical taxonomy by stacking multiple GNN modules
- Assign price level for each layer of clusters
- Aggregate all price level as item's price level



Heterogeneous Graph Attention Network(HAN)

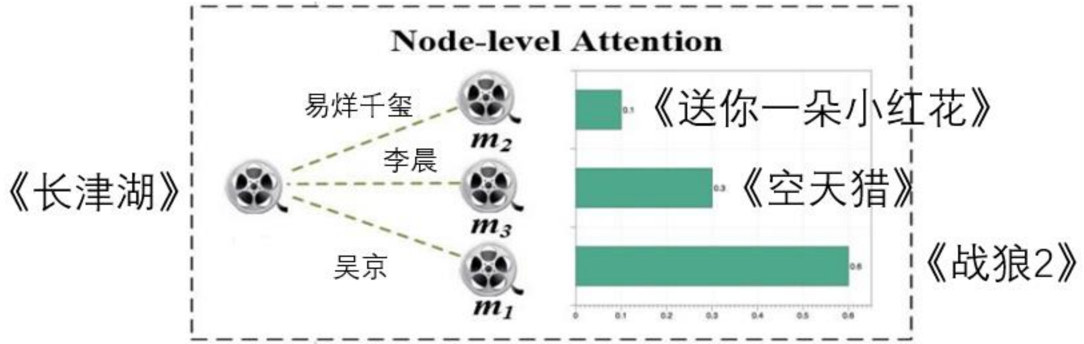
- Heterogeneous Graph
  - Multiple types of nodes or links
  - Rich semantic information
    - Meta-path: a relation sequence connecting two objects (e.g., Movie-Actor-Movie).

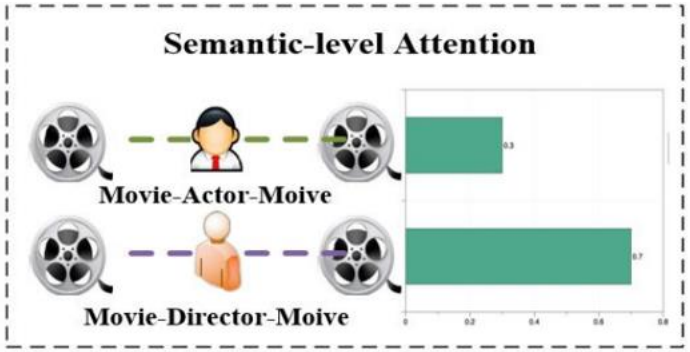


[9] Heterogeneous Graph Attention Network. WWW 2019

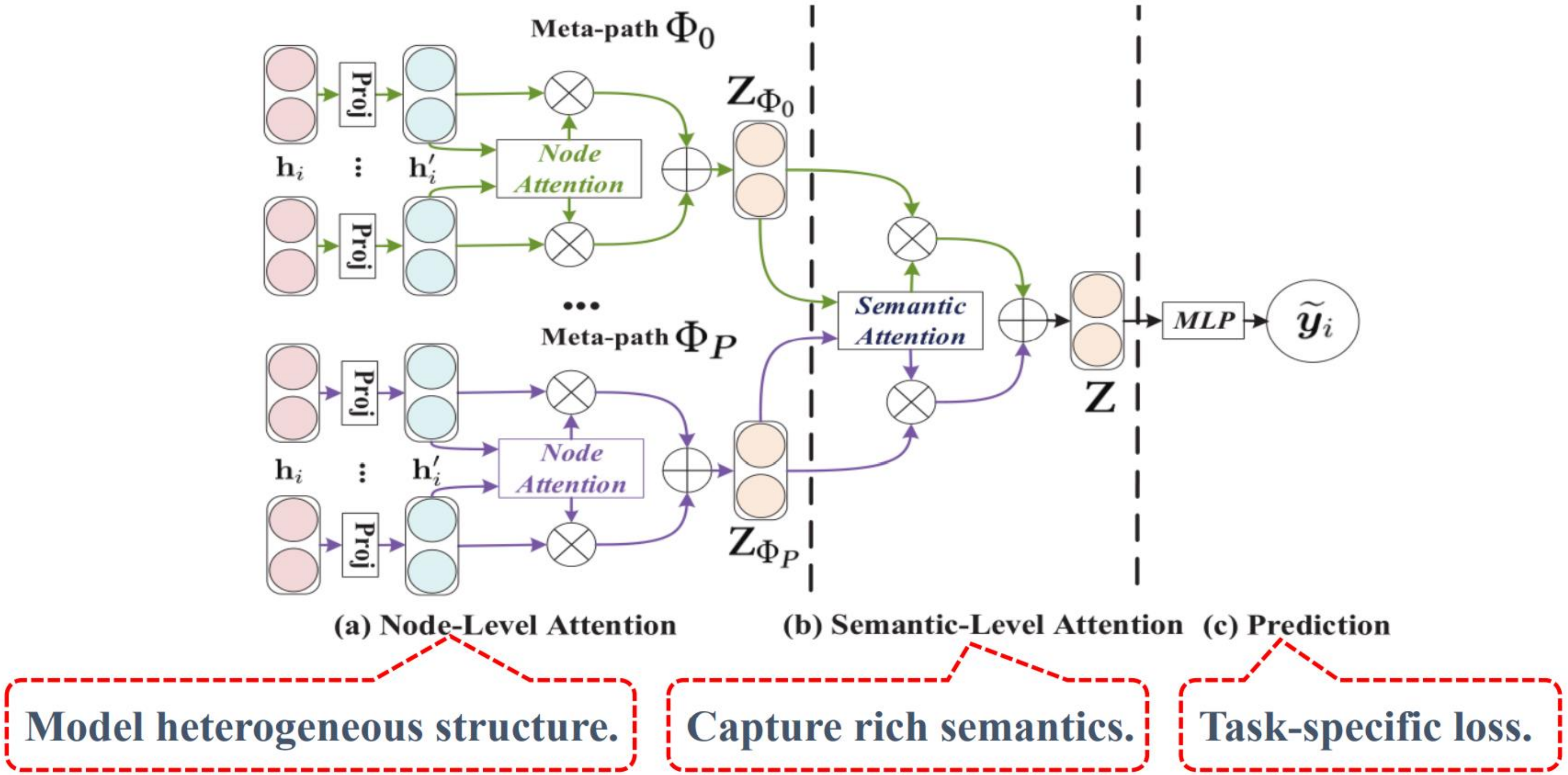
HAN's Goal

- Node-level Attention
  - Discover the differences of meta-path based neighbors
- Semantic-level Attention
  - Find some meaningful meta-paths





Overall Framework



[9] Heterogeneous Graph Attention Network. WWW 2019

Node-Level Attention and Aggregating

- Type-Specific Transformation

$$\mathbf{h}'_i = \mathbf{M}_{\phi_i} \cdot \mathbf{h}_i,$$

Type-specific transformation matrix

- Importance of Neighbors

$$e_{ij}^{\Phi} = att_{node}\left(\mathbf{h}'_i, \mathbf{h}'_j; \Phi\right)$$

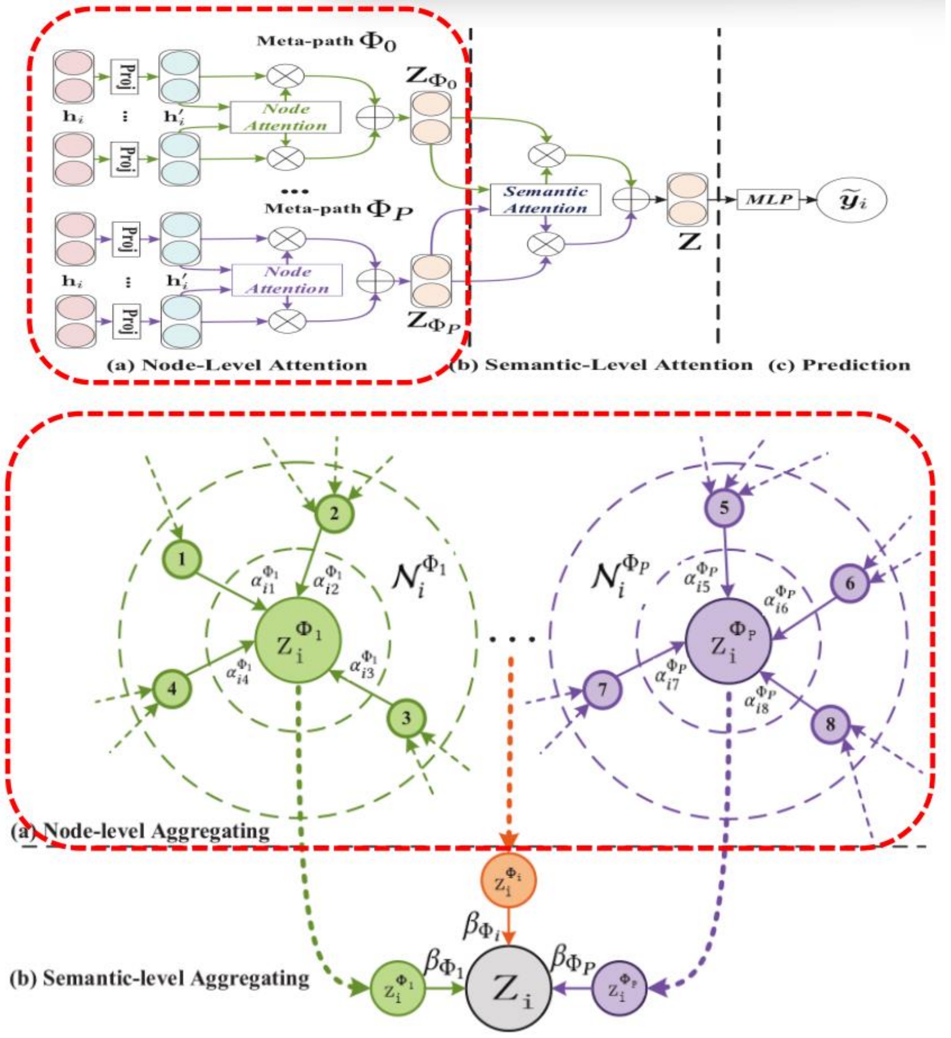
$$e_{ij}^{\Phi} = \sigma\left(\mathbf{a}_{\Phi}^{\mathrm{T}} \cdot \left[\mathbf{h}'_i \| \mathbf{h}'_j\right]\right)$$

$$\alpha_{ij}^{\Phi} = softmax_j\left(e_{ij}^{\Phi}\right)$$

Node-level attention vector

- Node-Level Aggregating

$$\mathbf{z}_i^{\Phi} = \sigma\left(\sum_{j \in \mathcal{N}_i^{\Phi}} \alpha_{ij}^{\Phi} \cdot \mathbf{h}'_j\right).$$



Semantic-Level Attention and Aggregating

- Semantic-Level Attention

$$(\beta_{\Phi_0}, \beta_{\Phi_1}, \ldots, \beta_{\Phi_P}) = att_{sem}(\mathbf{Z}_{\Phi_0}, \mathbf{Z}_{\Phi_1}, \ldots, \mathbf{Z}_{\Phi_P})$$

- Importance of Neighbors

Semantic-level attention vector

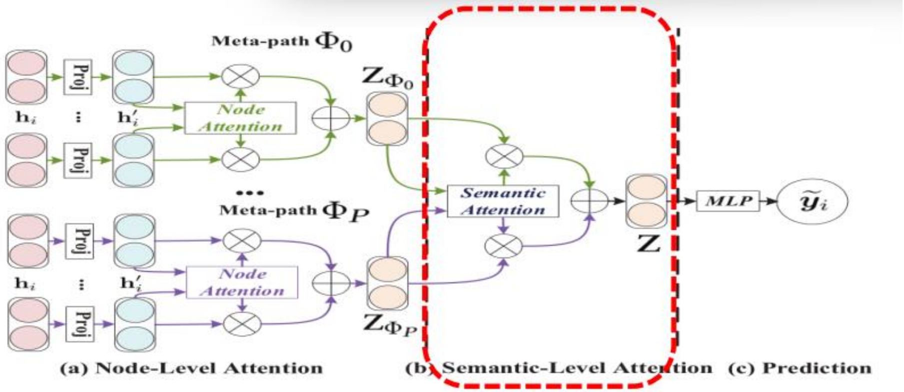
$$w_{\Phi_i} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \mathbf{q}^{\mathrm{T}} \cdot \tanh(\mathbf{W} \cdot \mathbf{z}_i^{\Phi} + \mathbf{b})$$

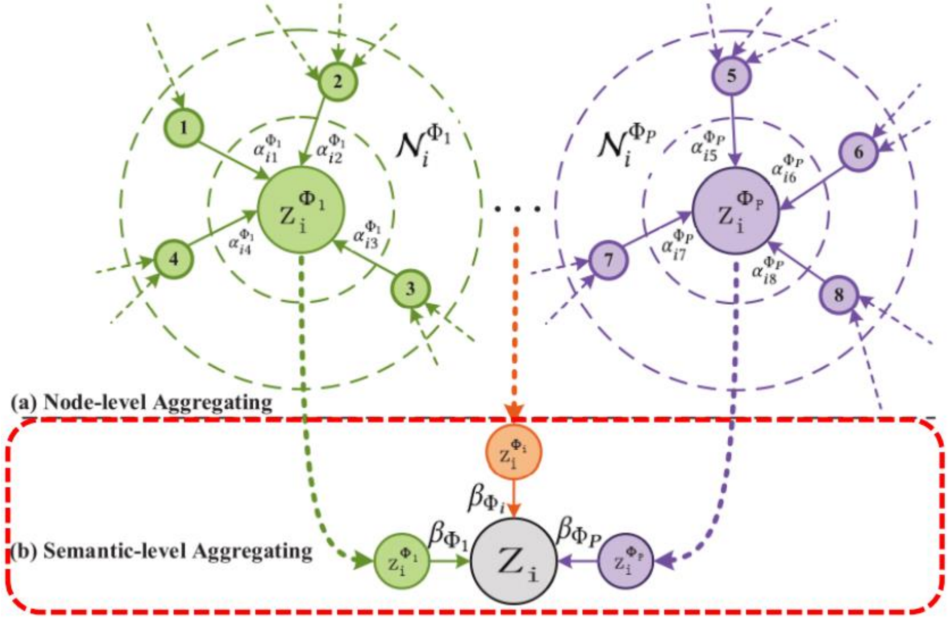
$$\beta_{\Phi_i} = \frac{\exp(w_{\Phi_i})}{\sum_{i=1}^{P} \exp(w_{\Phi_i})}$$

- Node-Level Aggregating

$$\mathbf{Z} = \sum_{i=1}^{P} \beta_{\Phi_i} \cdot \mathbf{Z}_{\Phi_i}$$

Semantic weight





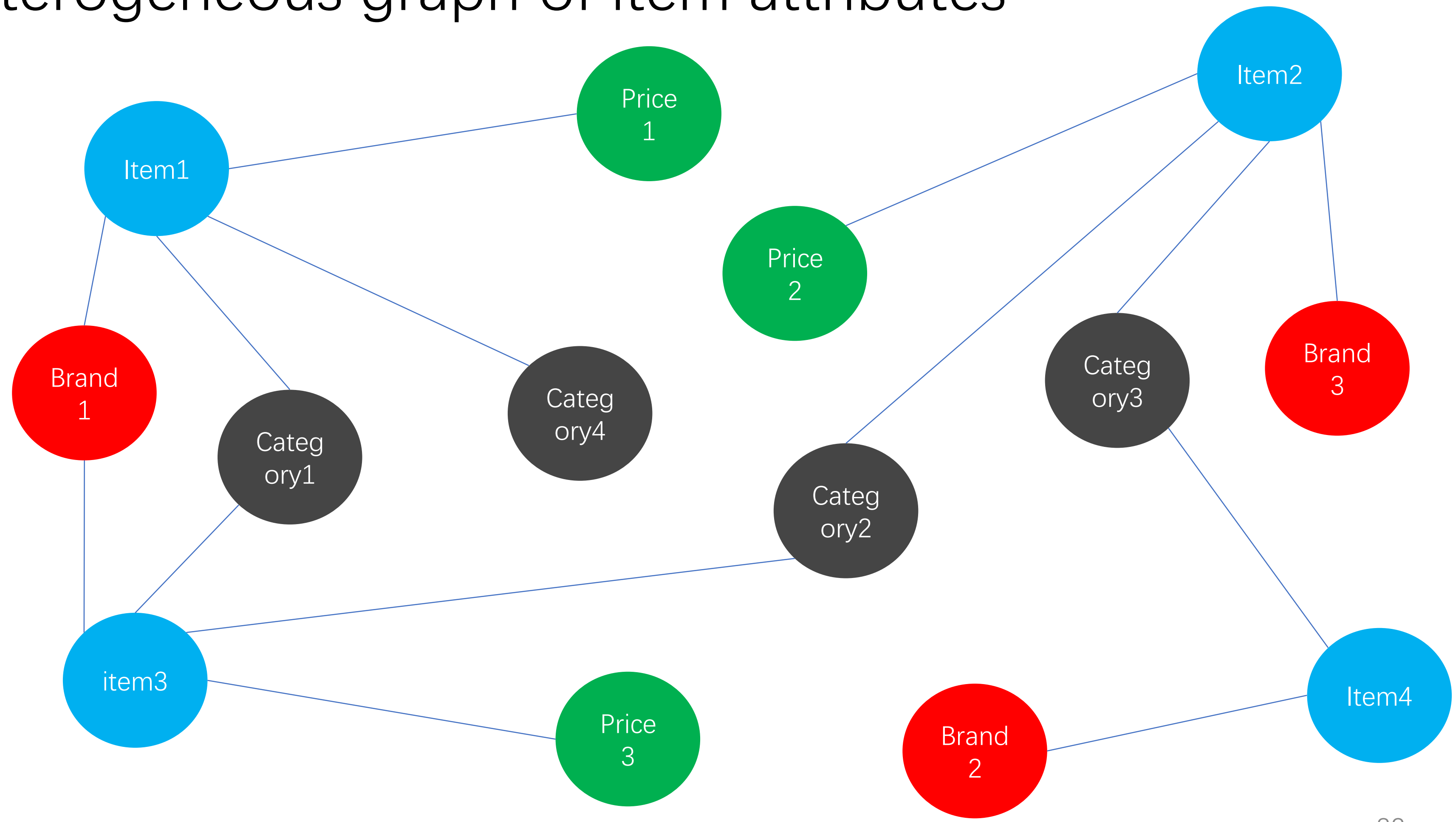
Experiments

Datasets	Metrics	Training	DeepWalk	ESim	metapath2vec	HERec	GCN	GAT	HAN_{nd}	HAN_{sem}	HAN
ACM	Macro-F1	20%	77.25	77.32	65.09	66.17	86.81	86.23	88.15	89.04	**89.40**
		40%	80.47	80.12	69.93	70.89	87.68	87.04	88.41	89.41	**89.79**
		60%	82.55	82.44	71.47	72.38	88.10	87.56	87.91	**90.00**	89.51
		80%	84.17	83.00	73.81	73.92	88.29	87.33	88.48	90.17	**90.63**
	Micro-F1	20%	76.92	76.89	65.00	66.03	86.77	86.01	87.99	88.85	**89.22**
		40%	79.99	79.70	69.75	70.73	87.64	86.79	88.31	89.27	**89.64**
		60%	82.11	82.02	71.29	72.24	88.12	87.40	87.68	**89.85**	89.33
		80%	83.88	82.89	73.69	73.84	88.35	87.11	88.26	89.95	**90.54**
DBLP	Macro-F1	20%	77.43	91.64	90.16	91.68	90.79	90.97	91.17	92.03	**92.24**
		40%	81.02	92.04	90.82	92.16	91.48	91.20	91.46	92.08	**92.40**
		60%	83.67	92.44	91.32	92.80	91.89	90.80	91.78	92.38	**92.80**
		80%	84.81	92.53	91.89	92.34	92.38	91.73	91.80	92.53	**93.08**
	Micro-F1	20%	79.37	92.73	91.53	92.69	91.71	91.96	92.05	92.99	**93.11**
		40%	82.73	93.07	92.03	93.18	92.31	92.16	92.38	93.00	**93.30**
		60%	85.27	93.39	92.48	93.70	92.62	91.84	92.69	93.31	**93.70**
		80%	86.26	93.44	92.80	93.27	93.09	92.55	92.69	93.29	**93.99**
IMDB	Macro-F1	20%	40.72	32.10	41.16	41.65	45.73	49.44	49.78	**50.87**	50.00
		40%	45.19	31.94	44.22	43.86	48.01	50.64	52.11	50.85	**52.71**
		60%	48.13	31.68	45.11	46.27	49.15	51.90	51.73	52.09	**54.24**
		80%	50.35	32.06	45.15	47.64	51.81	52.99	52.66	51.60	**54.38**
	Micro-F1	20%	46.38	35.28	45.65	45.81	49.78	55.28	54.17	55.01	**55.73**
		40%	49.99	35.47	48.24	47.59	51.71	55.91	56.39	55.15	**57.97**
		60%	52.21	35.64	49.09	49.88	52.29	56.44	56.09	56.66	**58.32**
		80%	54.33	35.59	48.81	50.99	54.61	56.97	56.38	56.49	**58.51**

[9] Heterogeneous Graph Attention Network. WWW 2019

Inspired by HAN

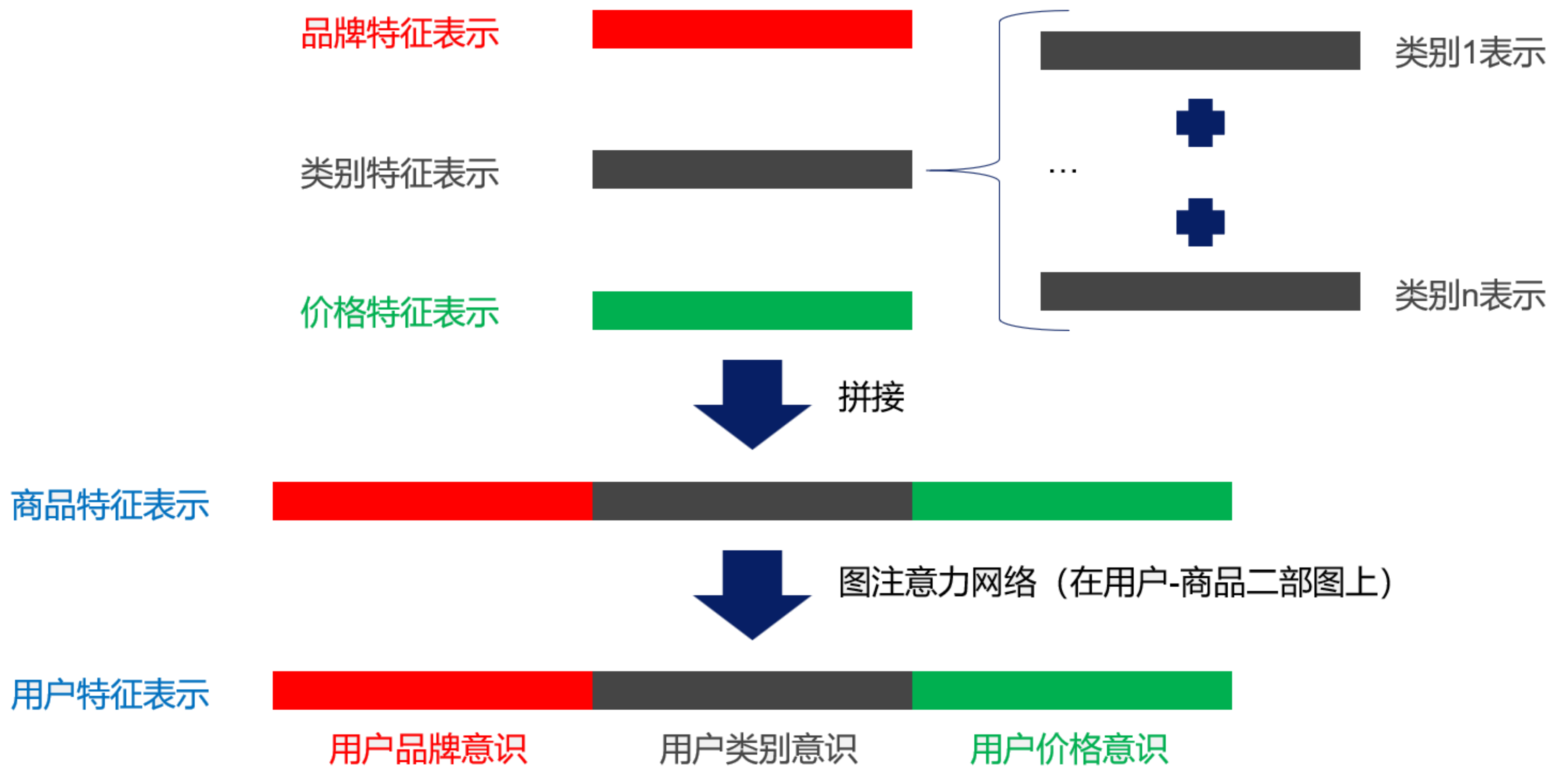
- Construct a heterogeneous graph of item attributes
  - Item
  - Brand
  - Price(level)
- Each node captures rich semantic information



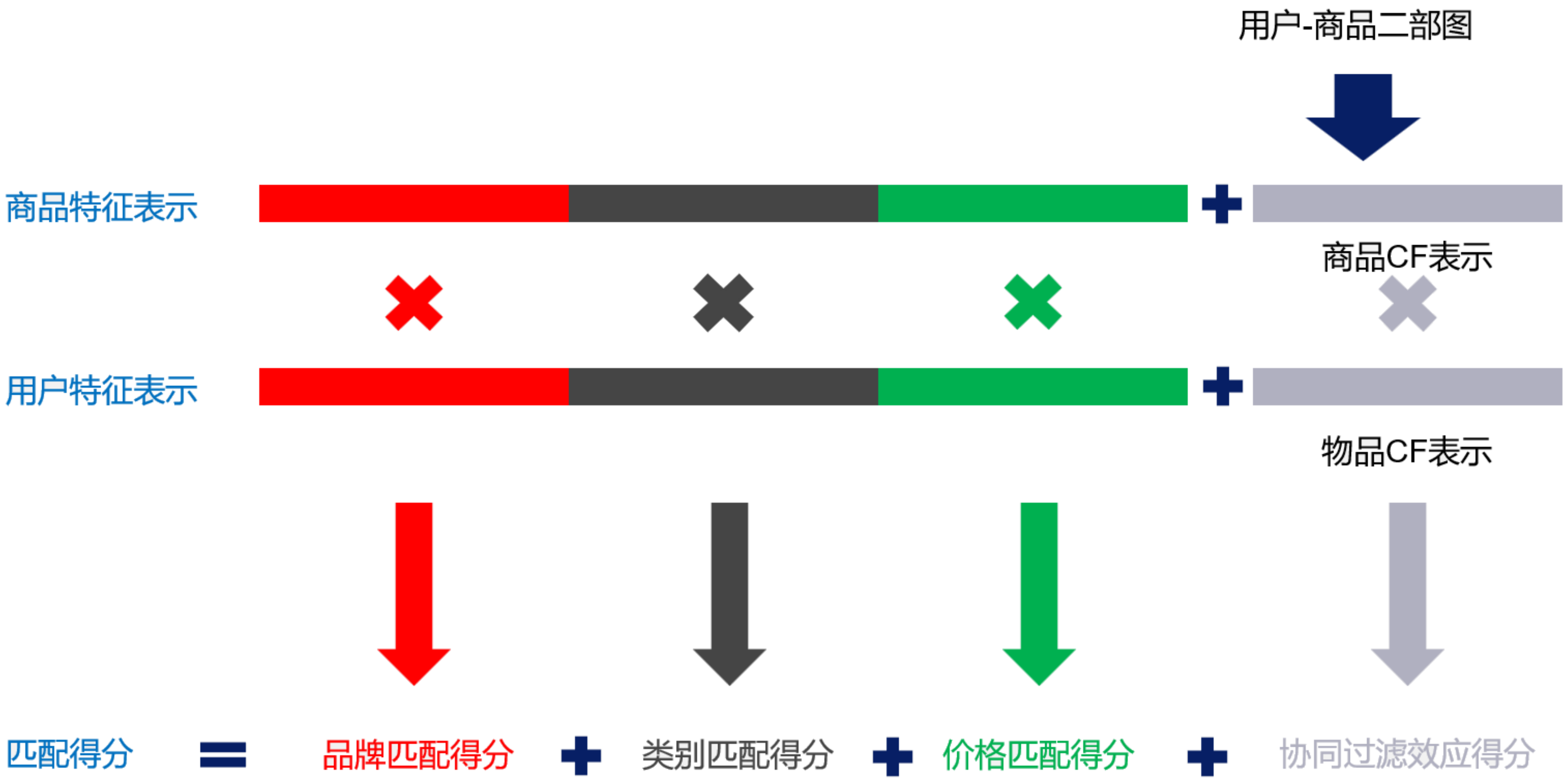
Future work

- Incorporating Brand into E-commerce RS with GNN
  - Modeling brand factor in heterogeneous graph, and learn the brand awareness of users
  - Hierarchically Cluster items via HiGNN and assign a more reasonable price level to the item
  - Learn item feature and user feature representations in the same feature space, and make RS explainable

商品特征与用户特征表示(TO DO)



匹配得分的可解释性(TO DO)



Thanks