Candidate–aware Graph Contrastive Learning for Recommendation

Presented by: Wei He

Wei He, Guohao Sun, Jinhu Lu, Xiu Susie Fang

Donghua University

Recommender Systems (RS)

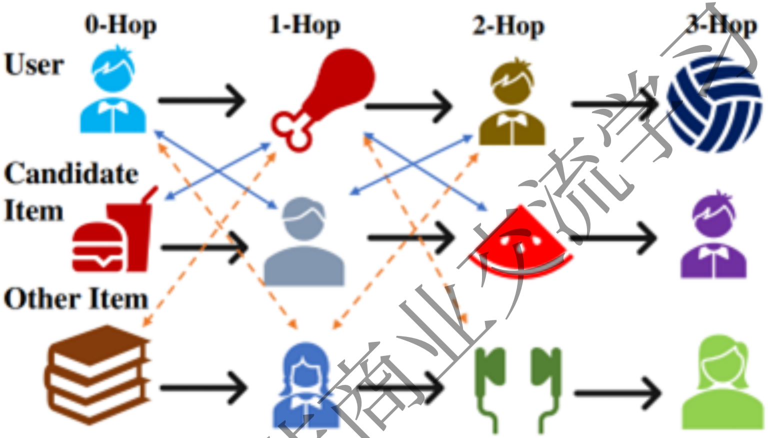
- Predicting whether the user will ***interact*** with the item
  - News, advertising, e-commerce
- Modeling the interaction data as a graph
  - **User-item interaction graph**, **heterogeneous graph**, and **knowledge graph**
- GNN-based methods
  - Stack **multilayer convolution** to **iteratively aggregate** information from the **local neighbor nodes** to enrich the embedding of nodes.
- Heavily rely on **explicit supervision** signals during training
  - ***implicit***, ***sparse***, and ***noisy***, resulting in a lack of ***explicit supervision signals***

Graph Contrastive Learning(GCL) recommendation

- Contrastive learning (CL)
  - Extract **features** of data from **unlabeled data** for the improvement of downstream tasks.
- The key to GCL
  - Generating appropriate ***anchors***, and ***positive*** and ***negative instances*** through the view generator.
- Most GCL-based methods use **heuristic-based view generators** to generate instances.
- Data augmentation with **random perturbations**, such as random node/edge drop, random walk, diffusion, etc.
  - Changes the **structure of the graph** and cannot preserve **semantic information** well, and easy to discard **important information** or introduce **noise**
- The heuristic-based data augmentation strategy is not **universal**
  - Requires **manual intervention**, and selects the **appropriate augmentation strategy** according to the specific application scenario, which limits the **adaptation range** of the model.

Recommender Systems (RS)

- Ignore the relationship between the ***different layer embeddings*** of the ***user*** and ***candidate item***



  - Drumstick should be close to cola and watermelon and away from books and headphones in the embedding space because drumsticks, cola, and watermelon belong to the same class of the food

- The embeddings with ***similar semantic information*** should be ***closer to each other*** than embeddings **without similar semantic information**

PROBLEM AND BACKGROUND

- Historical interaction records
  - The set of users U = $\{u\}$ and items I = $\{i\}$
  - I^+: items that users have interacted with
  - $I^-(I - I^+)$: items have not observed interaction records
- User item interaction graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$
  - The node set $\mathcal{V} = \mathcal{U} \cup \mathcal{I}$ involves all users and items
  - The edge set $\mathcal{E} = \{A_{ui} | A_{ui} = 1\}$ represents observed interactions
- The ***goal*** of CGCL
  - To predict ***whether the user u will interact with the unobserved item i*** by the given input graph $\mathcal{G}$

Graph neural network in recommendation

- **Embedding**
  - The initialization embedding of the user (item)

$$e_u^{(0)} = lookup\,(u)\,, \quad e_i^{(0)} = lookup\,(i)\,.$$

- **Aggregation**
  - The embedding of neighboring nodes in the l layer

$$e_{u,N}^{(l)} = Agg\left(e_i^{(l-1)}, i \in N_u\right),$$
$$e_{i,N}^{(l)} = Agg\left(e_u^{(l-1)}, u \in N_i\right).$$

- **Propagate**
  - The embedding of aggregate neighbor nodes to update its own embedding

$$e_u^{(l)} = Prop\left(e_u^{(l-1)}, e_{u,N}^{(l)}\right),$$
$$e_i^{(l)} = Prop\left(e_i^{(l-1)}, e_{i,N}^{(l)}\right).$$

Graph neural network in recommendation

- **Readout**
  - The final representation of the user and item

$$e_u = Readout\left(e_u^{(0)}, \cdots, e_u^{(L)}\right),$$
$$e_i = Readout\left(e_i^{(0)}, \cdots, e_i^{(L)}\right).$$

- **Prediction**
  - The likelihood of interaction between the user u and the item i

$$\hat{y}_{ui} = f(e_u, e_i).$$

- **Optimize**
  - The parameters of the model are optimized by minimizing the loss function

$$\mathcal{L}_{Rec} = g(y_{ui}, \hat{y}_{ui}).$$

LightGCN Backbone

- The **aggregation process** of LightGCN

$$e_{u,N}^{(l+1)} = \sum_{i \in N_u} \frac{1}{\sqrt{|N_u||N_i|}} e_i^{(l)},$$
$$e_{i,N}^{(l+1)} = \sum_{u \in N_i} \frac{1}{\sqrt{|N_i||N_u|}} e_u^{(l)}.$$

- The **propagating process** of information

$$e_u^{(l+1)} = e_{u,N}^{(l+1)}, \quad e_i^{(l+1)} = e_{i,N}^{(l+1)}.$$

- The **readout function** of LightGCN

$$e_u = \frac{1}{L+1} \sum_{l=0}^{L} e_u^{(l)}, \quad e_i = \frac{1}{L+1} \sum_{l=0}^{L} e_i^{(l)}.$$

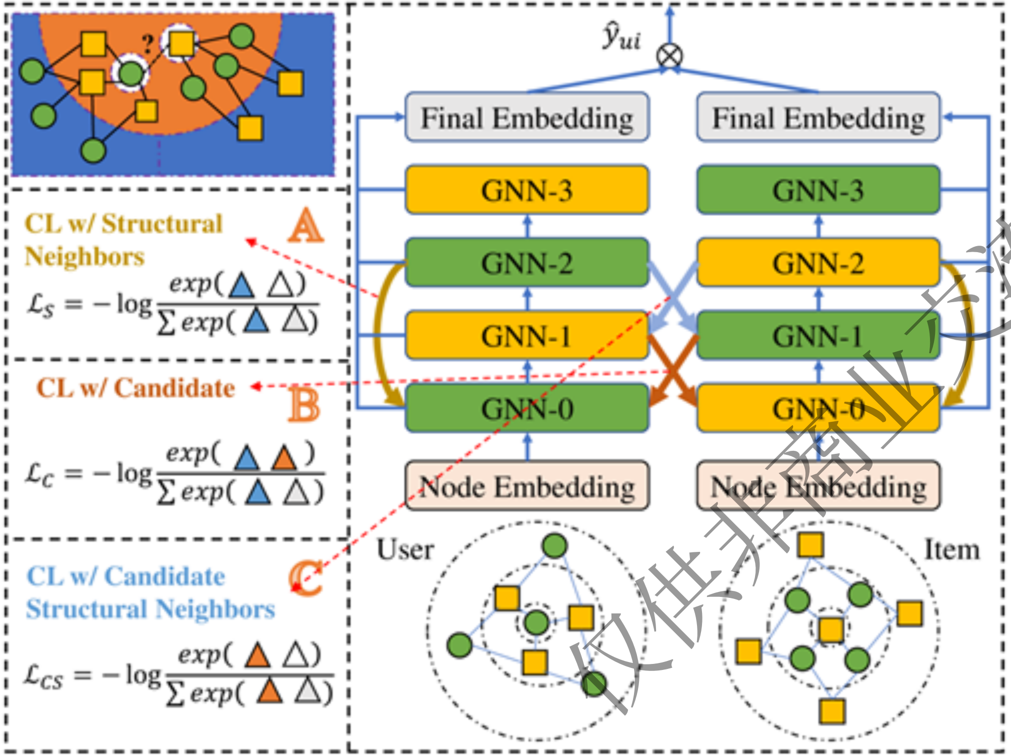
- The **likelihood of interaction** between the user u and the item i

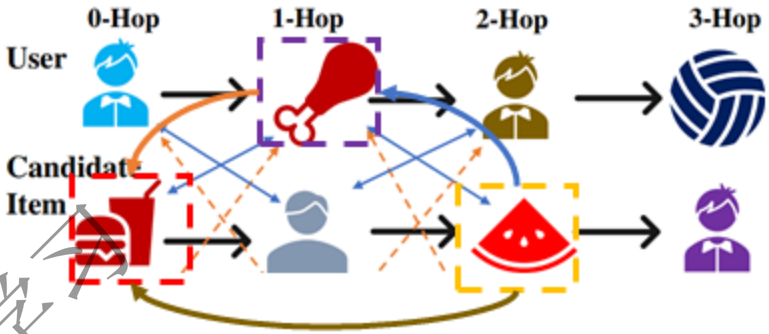
$$\hat{y}_{ui} = e_u^T e_i.$$

- The used BPR loss function

$$\mathcal{L}_{Rec} = \frac{1}{|O|} \sum_{(u,i,j) \in O} -\log \sigma\left(\hat{y}_{ui} - \hat{y}_{uj}\right).$$

THE PROPOSED METHOD





- A: The **structural neighbor contrastive learning** objects
- B: The **candidate contrastive learning** objects
- C: The **candidate structural neighbor contrastive learning** objects

Contrastive Learning with Structural Neighbors

- The **structural neighbor contrastive** learning loss on the ***user side***

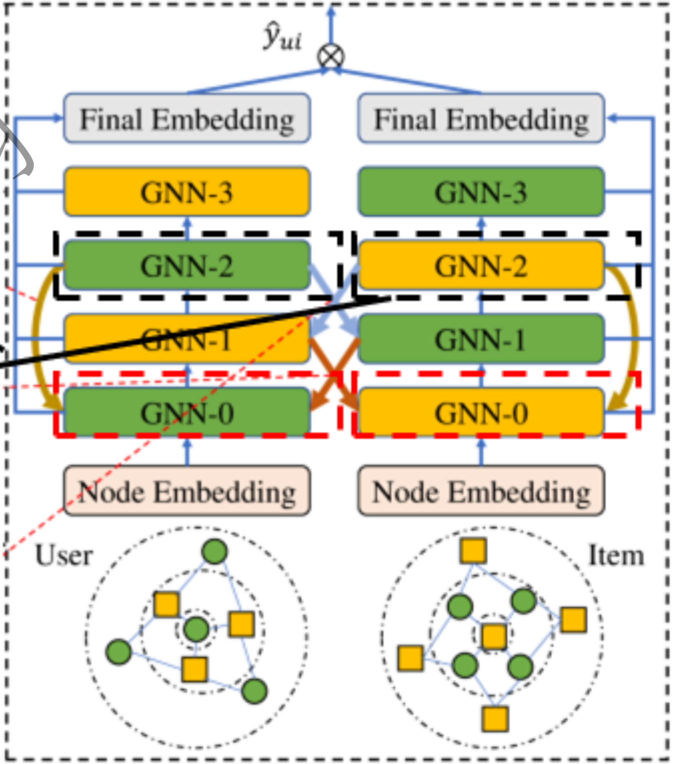
$$\mathcal{L}_S^U = \sum_{u\in\mathcal{U}} -\log \frac{\exp\left(sim\left(\boxed{e_u^{(k)}}, e_u^{(0)}\right)/\tau\right)}{\sum_{v\in\mathcal{U}} \exp\left(sim\left(e_u^{(k)}, \boxed{e_v^{(0)}}\right)/\tau\right)}.$$

- The **structural neighbor contrastive** learning loss on the ***item side***

$$\mathcal{L}_S^I = \sum_{i\in\mathcal{I}} -\log \frac{\exp\left(sim\left(\boxed{e_i^{(k)}}, e_i^{(0)}\right)/\tau\right)}{\sum_{j\in\mathcal{I}} \exp\left(sim\left(e_i^{(k)}, \boxed{e_j^{(0)}}\right)/\tau\right)}.$$

- The ***objective function*** of **structural neighbor contrastive** learning loss

$$\boxed{\mathcal{L}_S = \alpha\mathcal{L}_S^U + (1-\alpha)\mathcal{L}_S^I.}$$



Contrastive Learning with Structural Neighbors

- The **candidate contrastive** learning loss on the ***user side***

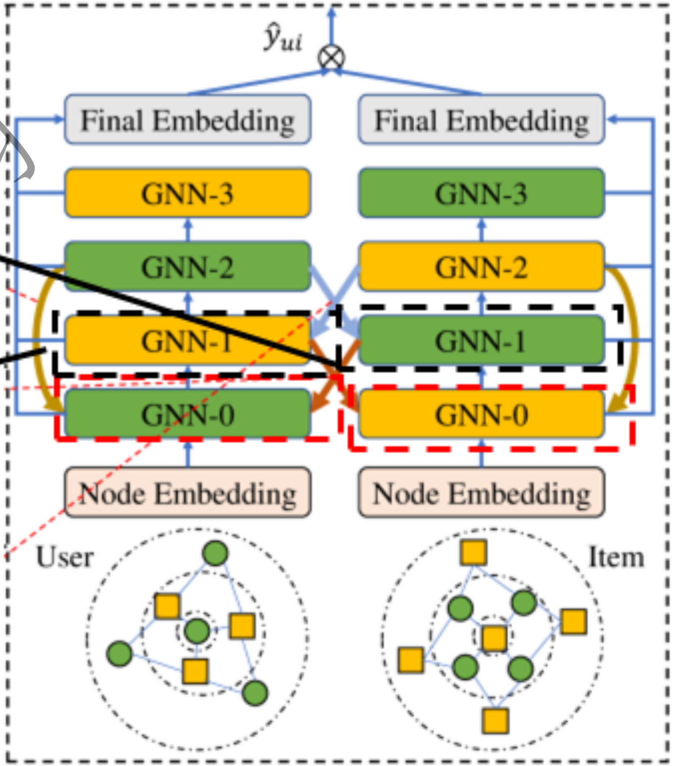
$$\mathcal{L}_C^U = \sum_{i\in\mathcal{I}} -\log \frac{\exp\left(sim\left(e_i^{(k)}, e_u^{(0)}\right)/\tau\right)}{\sum_{v\in\mathcal{U}} \exp\left(sim\left(e_i^{(k')}, e_v^{(0)}\right)/\tau\right)}.$$

- The **candidate contrastive** learning loss on the ***item side***

$$\mathcal{L}_C^I = \sum_{v\in\mathcal{U}} -\log \frac{\exp\left(sim\left(e_u^{(k)}, e_i^{(0)}\right)/\tau\right)}{\sum_{j\in\mathcal{I}} \exp\left(sim\left(e_u^{(k')}, e_j^{(0)}\right)/\tau\right)}.$$

- The ***objective function*** of **candidate contrastive** learning loss

$$\boxed{\mathcal{L}_C = \alpha\mathcal{L}_C^U + (1-\alpha)\mathcal{L}_C^I.}$$



Contrastive Learning with Candidate Structure Neighbors

- The **candidate structure neighbor** contrastive learning loss on the ***user side***

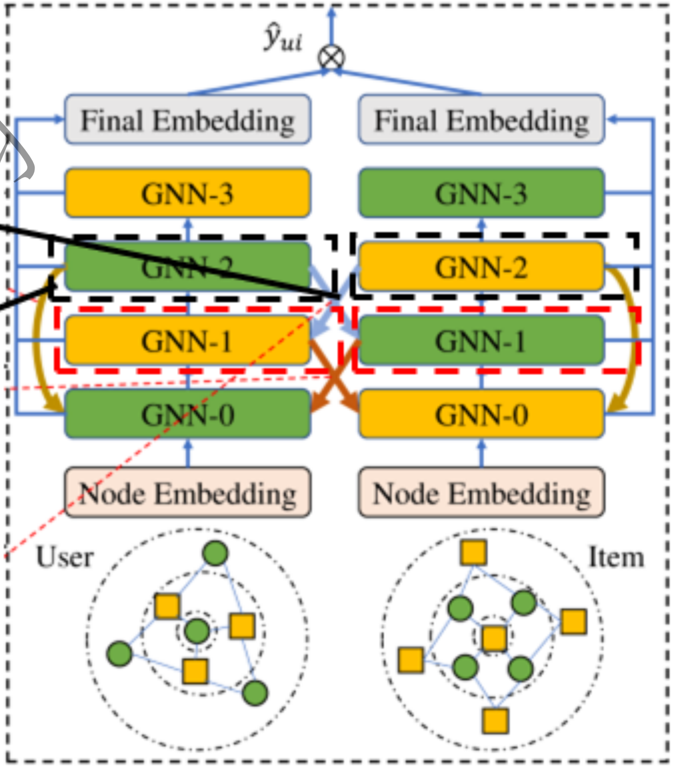
$$\mathcal{L}_{CS}^{U} = \sum_{i \in \mathcal{I}} -\log \frac{\exp\left(sim\left(\boxed{e_i^{(k)}}, e_u^{(k')}\right)/\tau\right)}{\sum_{v \in \mathcal{U}} \exp\left(sim\left(e_i^{(k)}, \boxed{e_v^{(k')}}\right)/\tau\right)}$$

- The **candidate structure neighbor** contrastive learning loss on the ***item side***

$$\mathcal{L}_{CS}^{I} = \sum_{v \in \mathcal{U}} -\log \frac{\exp\left(sim\left(\boxed{e_u^{(k)}}, e_i^{(k')}\right)/\tau\right)}{\sum_{j \in \mathcal{I}} \exp\left(sim\left(e_u^{(k)}, \boxed{e_j^{(k')}}\right)/\tau\right)}$$

- The ***objective function*** of **candidate structure neighbor** contrastive learning loss

$$\boxed{\mathcal{L}_{CS} = \alpha \mathcal{L}_{CS}^{U} + (1-\alpha)\, \mathcal{L}_{CS}^{I}}$$



Optimization with CGCL

- The total loss function

$$\mathcal{L}_{CGCL} = \mathcal{L}_{Rec} + \lambda_1 \mathcal{L}_S + \lambda_2 \mathcal{L}_C + \lambda_3 \mathcal{L}_{CS} + \lambda_4 \|\Theta\|_2^2 .$$

Algorithm 1 Candidate-aware Graph Contrastive Learning

Input: User-Item bipartite graph $\mathcal{G} = \{\mathcal{U} \cup \mathcal{I}, \mathcal{E}\}$, training dataset $\mathcal{X}$;

Parameter: Trainable parameters $\{e\}_{u \in \mathcal{U}}, \{e\}_{i \in \mathcal{I}}$, Hyperparameter $\alpha, \lambda_1, \lambda_2, \lambda_3, \lambda_4$;

Output: Trained Model $\mathcal{F}(e_u, e_i | \Theta, \mathcal{G})$;

```
while CGCL Not Convergence do
    for x in Dataloader(X) do
        // Forward propagation
        e_u, e_i = GNN(G, e_u^(0), e_i^(0));
        // Calculate Loss
        Calculate BPR loss L_Rec;
        Calculate contrastive learning with structural neighbors loss L_S;
        Calculate contrastive learning with candidate loss L_C;
        Calculate contrastive learning with candidate structure neighbors loss L_CS;
        Calculate total loss L_CGCL;
        // Back propagation
        e_u = e_u - α ∂L/∂e_u;
        e_i = e_i - α ∂L/∂e_i;
    end for
end while
return F(e_u, e_i | Θ, G);
```

Experimental Setup

- **Dataset**
  - Yelp, Gowalla, and Amazon Books.
  - The training, validation, and test sets, 8:1:1
- **Metrics**
  - Recall@N and NDCG@N, N=20, 50
- **Baselines**
  - DNN
    - NeuMF [www 2017], DMF [IJCAI 2017]
  - GNN
    - GCMC [KDD 2018], NGCF [SIGIR 2019], LightGCN [SIGIR 2020]
  - GCL
    - SGL [SIGIR 2021], Simplex [CIKM 2021], NCL [WWW 2022]
- **Implementation Details**
  - All models are implemented using Recbole

DataSets	#Users	#Items	#Interactions	#Density
Yelp	45,478	30,709	1,777,765	0.00127
Gowalla	29,859	40,989	1,027,464	0.00084
Books	58,145	58,052	2,517,437	0.00075

Table 1: Statistics of the datasets

Performance Comparison

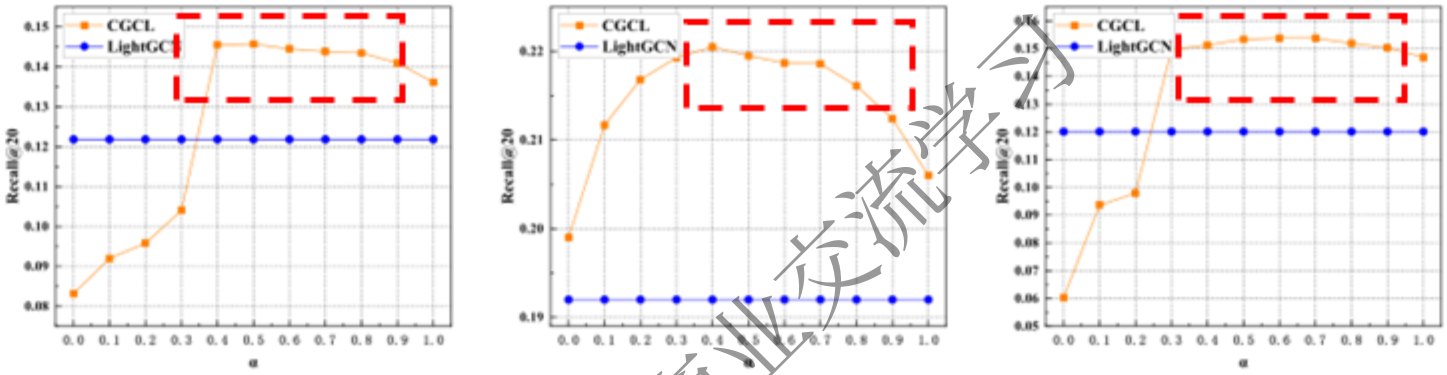
- Q1: How does the proposed CGCL perform compared to state-of-the-art ***DNN-***, ***GNN-*** and ***GCL-*** based methods?

DataSet	Yelp				Gowalla				Amazon-Books			
Metric	Recall		NDCG		Recall		NDCG		Recall		NDCG	
Method	20	50	20	50	20	50	20	50	20	50	20	50
NeuMF	0.0572	0.1457	0.0328	0.0510	0.1167	0.2255	0.0753	0.0969	0.0530	0.1216	0.0339	0.0486
DMF	0.0660	0.1253	0.0353	0.0507	0.1025	0.1706	0.0591	0.0759	0.0542	0.1007	0.0293	0.0414
GCMC	0.0908	0.1690	0.0494	0.0696	0.1472	0.2410	0.0810	0.1038	0.0868	0.1553	0.0481	0.0658
NGCF	0.0953	0.1764	0.0519	0.0729	0.1576	0.2546	0.0893	0.1130	0.0902	0.1598	0.0495	0.0676
LightGCN	0.1218	0.2123	0.0690	0.0741	0.1920	0.2988	0.1133	0.1192	0.1201	0.2002	0.0691	0.0737
SGL-ED	0.1335	0.2177	0.0808	0.1028	0.2153	0.3277	0.1281	0.1558	0.1418	0.2172	0.0866	0.1065
SimpleX	0.1221	0.2044	0.0728	0.0941	0.1555	0.2601	0.0784	0.1040	0.1339	0.2143	0.0790	0.1001
NCL	0.1394	0.2258	0.0816	0.1044	0.2147	0.3299	0.1260	0.1543	0.1389	0.2209	0.0818	0.1034
CGCL	**0.1457**	**0.2404**	**0.0849**	**0.1097**	**0.2205**	**0.3391**	**0.1292**	**0.1583**	**0.1538**	**0.2400**	**0.0920**	**0.1147**
Imp (%).	4.52	6.47	4.04	5.08	2.42	2.79	0.86	1.60	8.46	8.65	6.24	7.70

- CGCL achieves the best performance on all datasets, significantly exceeding state-of-the-art methods
  - CGCL can optimize the quality of node embeddings more reasonably by establishing the relationship between the embeddings of the user and candidate item at different layers

Impact of the Balance Coefficient α.

- Q2: How do **different parameter settings** affect the performance of CGCL?

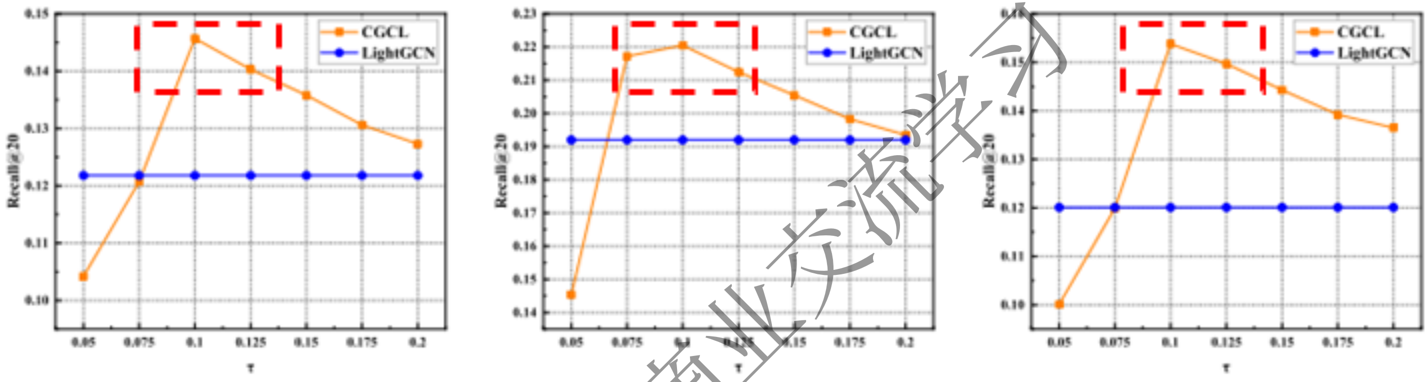


(a) Yelp. (b) Gowalla. (c) Amazon-Books.

- When the balance coefficient α is around 0.5, CGCL achieves the best performance
  - The appropriate balance coefficient can help the user and the item learn better embeddings
- Decreasing the importance of user-side contrastive learning loss, the performance decreased significantly
  - The user-side contrastive learning loss is more important than the item-side contrastive

Impact of the Temperature τ

- Q2: How do **different parameter settings** affect the performance of CGCL?

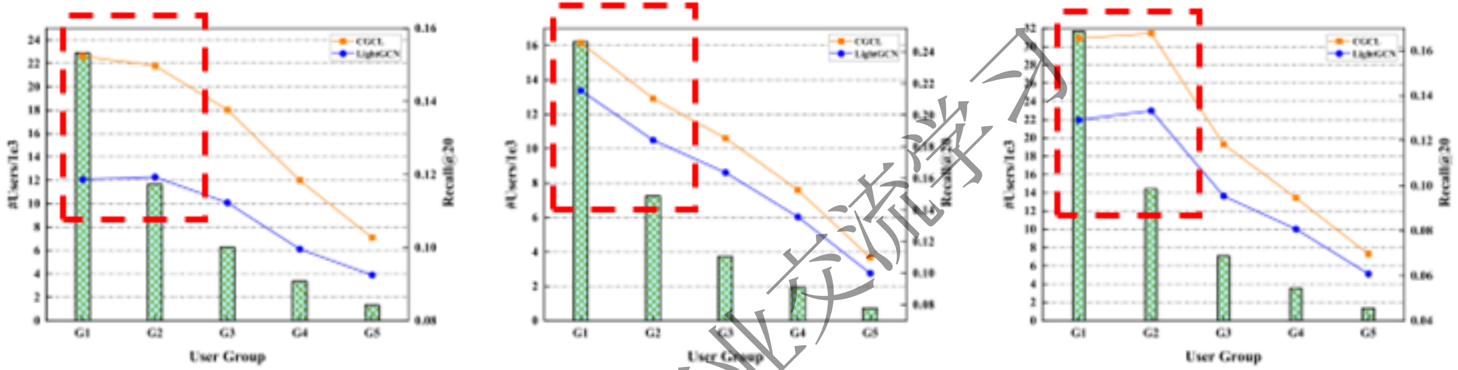


(a) Yelp. (b) Gowalla. (c) Amazon-Books.

- When the temperature τ is between 0.075 and 0.125, CGCL achieves the best results
  - Small temperature coefficient models can better focus on difficult sample learning so that similar user (item) node embeddings are separated from each other in the embedding space.
  - The high temperature will make it difficult to separate hard negative instances from positive instances.

Impact of Data Sparsity Levels

- Q3: How do data of **different sparsity** affect the performance of CGCL?

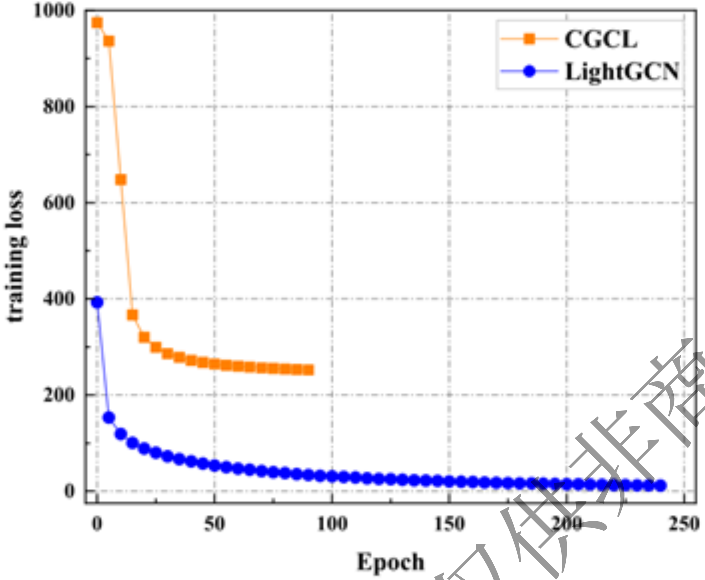


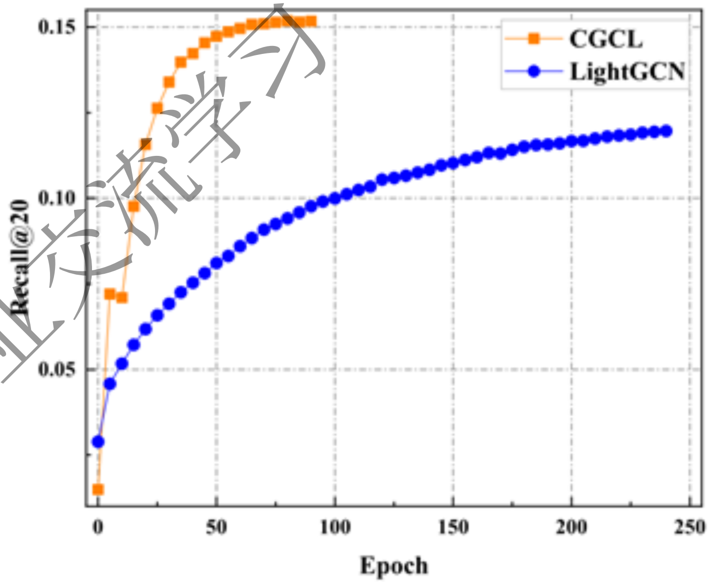
(a) Yelp. (b) Gowalla. (c) Amazon-Books.

- CGCL has achieved better results in all user groups
  - Exploring the relationship between the user and candidate item can effectively alleviate the data sparsity problem and enrich the semantic information of nodes.
- The number of interactions decreasing, made a greater performance gain
  - CGCL is more suitable for sparse interacting data

Impact of Data Sparsity Levels

- Q3: How do data of **different sparsity** affect the performance of CGCL?





- CGCL took fewer epochs to reach convergence than LighGCN
  - Explore hard negative samples, providing a larger gradient during training

Ablation Experimental

- Q4: How do **different parts** of CGCL contribute to the final performance?

DataSet	Yelp		Gowalla		Amazon-Books	
Metric	R@20	N@20	R@20	N@20	R@20	N@20
LightGCN	0.1218	0.069	0.1942	0.1123	0.1201	0.0691
O s	0.1328	0.0761	0.2028	0.1191	0.1236	0.0712
O c	0.1342	0.0768	0.2061	0.1195	0.1343	0.0780
O cs	0.1331	0.0755	0.2058	0.1202	0.1262	0.0729
W/o s	0.1327	0.0755	0.2056	0.1177	0.1259	0.0727
W/o c	0.1412	0.0824	0.2025	0.1184	0.1521	0.0910
W/o cs	0.1447	0.0846	0.2197	0.1291	0.1534	0.0908
All	0.1457	0.0849	0.2205	0.1292	0.1538	0.0920

- Removing or only retaining part of the structure of the models can achieve better performance
  - The three contrastive learning objects proposed can improve the quality of node embeddings
- Keeping two or all parts of the model can achieve better results than using only one
  - Each part of CGCL makes an important contribution to the final performance

Different GNN Backbones

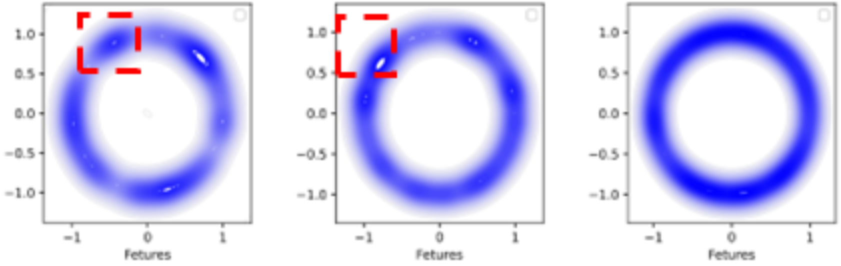
- Q5: How does CGCL perform when applied to **other GNN backbones**?

DataSet	Yelp		Gowalla		Amazon-Books	
Metric	R@20	N@20	R@20	N@20	R@20	N@20
NGCF	0.0953	0.0519	0.1576	0.0893	0.0902	0.0495
+CGCL	0.1084	0.0609	0.1812	0.1048	0.1259	0.0731
Imp(%).	13.75	17.34	14.97	17.36	39.58	47.68
LightGCN	0.1218	0.0690	0.1942	0.1123	0.1201	0.0691
+CGCL	0.1457	0.0849	0.2205	0.1292	0.1538	0.0920
Imp(%).	19.62	23.04	13.54	15.05	27.73	32.56

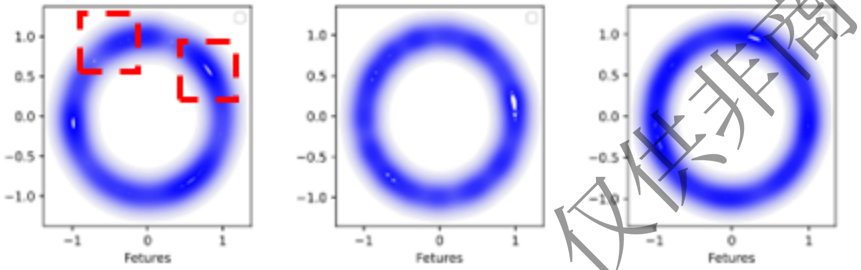
- CGCL achieves better results on the LightGCN backbone than on the NGCF backbone
  - LighGCN removes nonlinear activation and feature transformation from NGCF that are not conducive to model learning
  - NGCF with the self-loop makes embeddings contain the information of two different property nodes, which decreases the semantic similarity of embeddings

Visualizing Analysis

- Q6: What is the effect of CGCL on improving the **embedding distribution** of nodes?



(a) LightGCN in Yelp. (b) LightGCN in Gowalla. (c) LightGCN in Books.



(d) CGCL in Yelp. (e) CGCL in Gowalla. (f) CGCL in Books.

- LighGCN
  - several fixed regions
  - approximately evenly **distributed**
- CGCL
  - **uniformly distributed**
  - **prominently important**
- The CGCL can improve the quality of node embeddings and better model the preferences of different users and the characteristics of items

CONCLUSION

- Explore the relationship between the user and the candidate item and use similar semantic embeddings to construct contrastive pairs

- To establish the relationship between **nodes** and **structural neighbors**
  - Propose structural neighbor contrastive learning objects

- To establish the relationship between the **user** and **candidate item**
  - Propose candidate contrastive learning objects

- To establish the relationship between **users** and the **structural neighbors of the candidate item**
  - Propose candidate structural neighbor contrastive learning objects

Thanks for your attention.