Improving Recommendation via Contrastive Learning

Yupeng Hou

04/13/2022

Outline

- [5 min] Background - Contrastive Learning
- [10 min] Improving Collaborative Filtering via CL
- [8 min] Improving Session-based RecSys via CL
- [2 min] Conducting Research with 

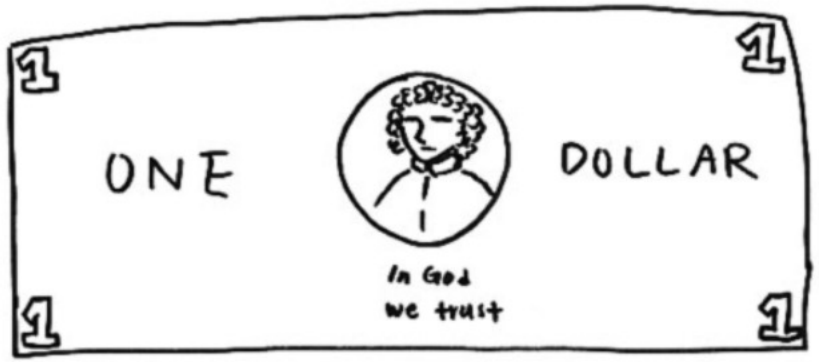
- [5 min] QA

Background

Contrastive Learning

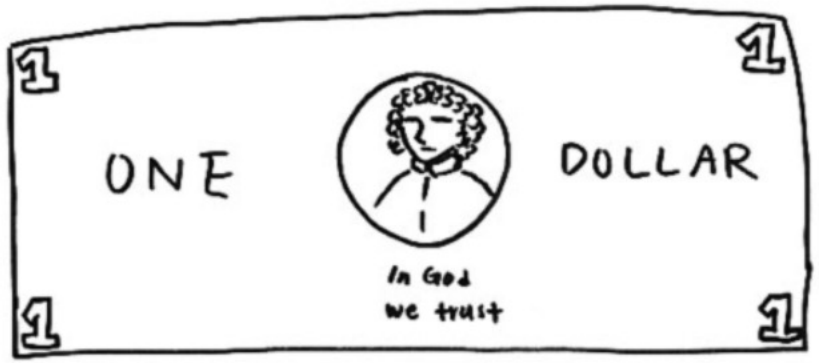
Background - Contrastive Learning

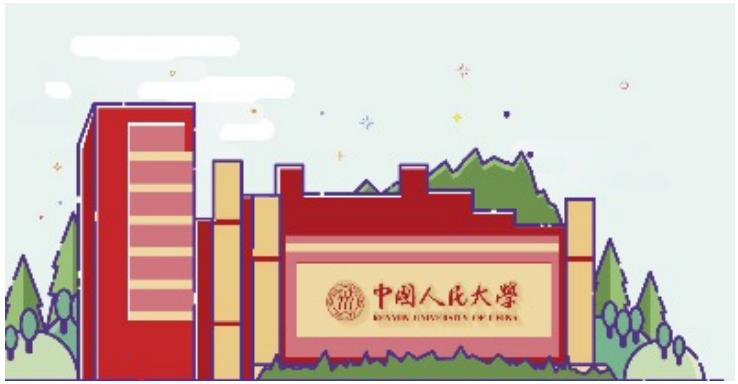
1



Which pair is more similar 🤔

2





Epstein, 2016. https://aeon.co/essays/your-brain-does-not-process-information-and-it-is-not-a-computer

Background - Contrastive Learning

- Arts of sample pairs and similarity

$$\text{score}(f(x), f(x^+)) >> \text{score}(f(x), f(x^-))$$

- Target: better encoder f
  - In other words, better distribution of encoded representations

Anand, 2020. https://ankeshanand.com/blog/2020/01/26/contrative-self-supervised-learning.html

Background - Contrastive Learning

- Arts of sample pairs and similarity (more mathematically)

$$\mathcal{L}_N = -\mathbb{E}_X \left[\log \frac{\exp\left(f(x)^T f\left(x^+\right)\right)}{\exp\left(f(x)^T f\left(x^+\right)\right) + \sum_{j=1}^{N-1} \exp\left(f(x)^T f\left(x_j\right)\right)} \right]$$

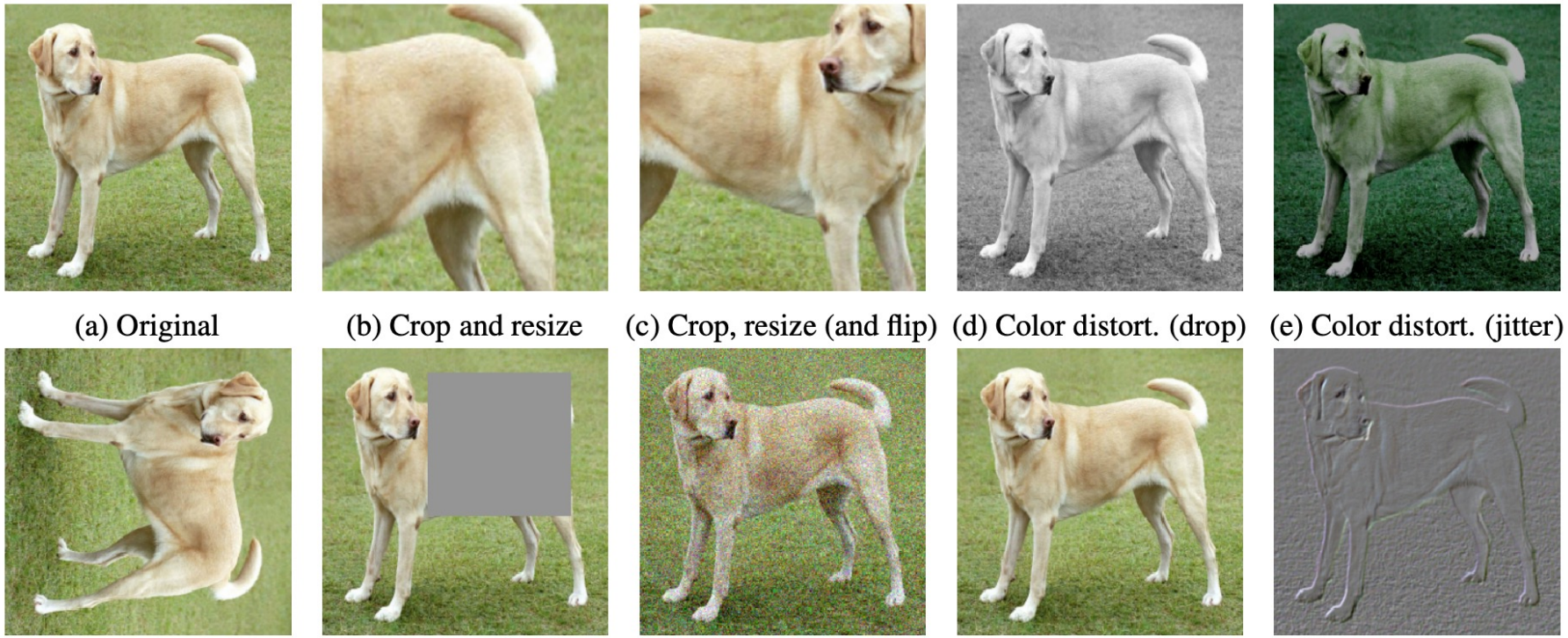
Anand, 2020. https://ankeshanand.com/blog/2020/01/26/contrative-self-supervised-learning.html

Background - Contrastive Learning

- Key points of CL
  - (Very) similar and easily-obtained positive pairs (mostly self-supervised, via data augmentation);
  - (Very) large amount of negative pairs;

Background - Contrastive Learning

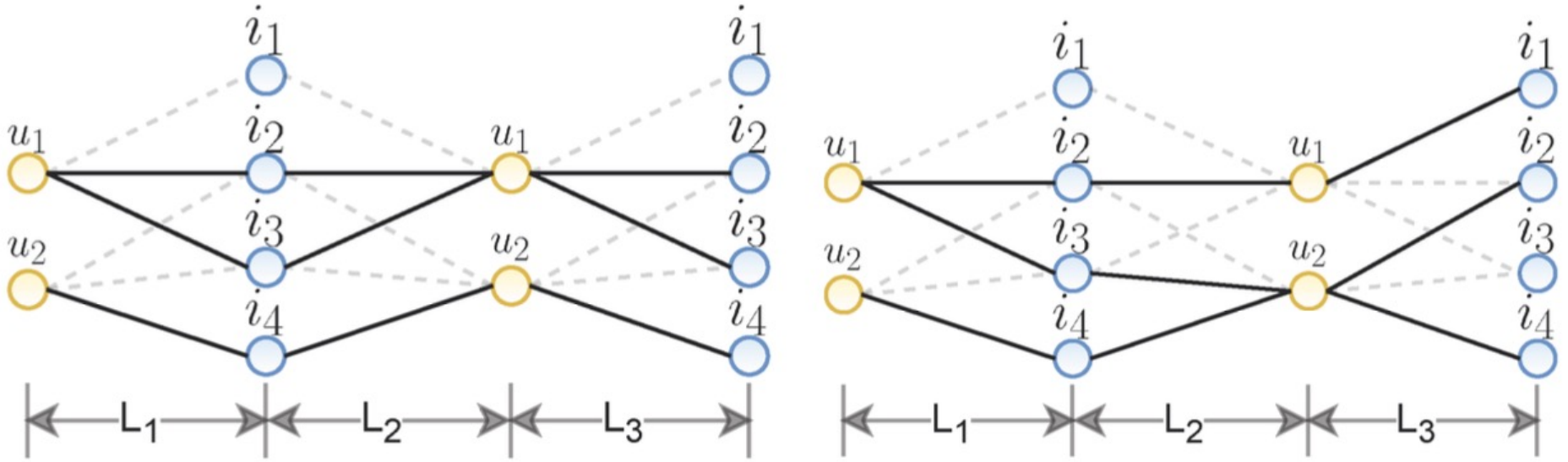
- Examples for self-supervised signal in CV



Chen et al. A Simple Framework for Contrastive Learning of Visual Representations. ICML 2020.

Background - Contrastive Learning

- Examples for self-supervised signal for graphs

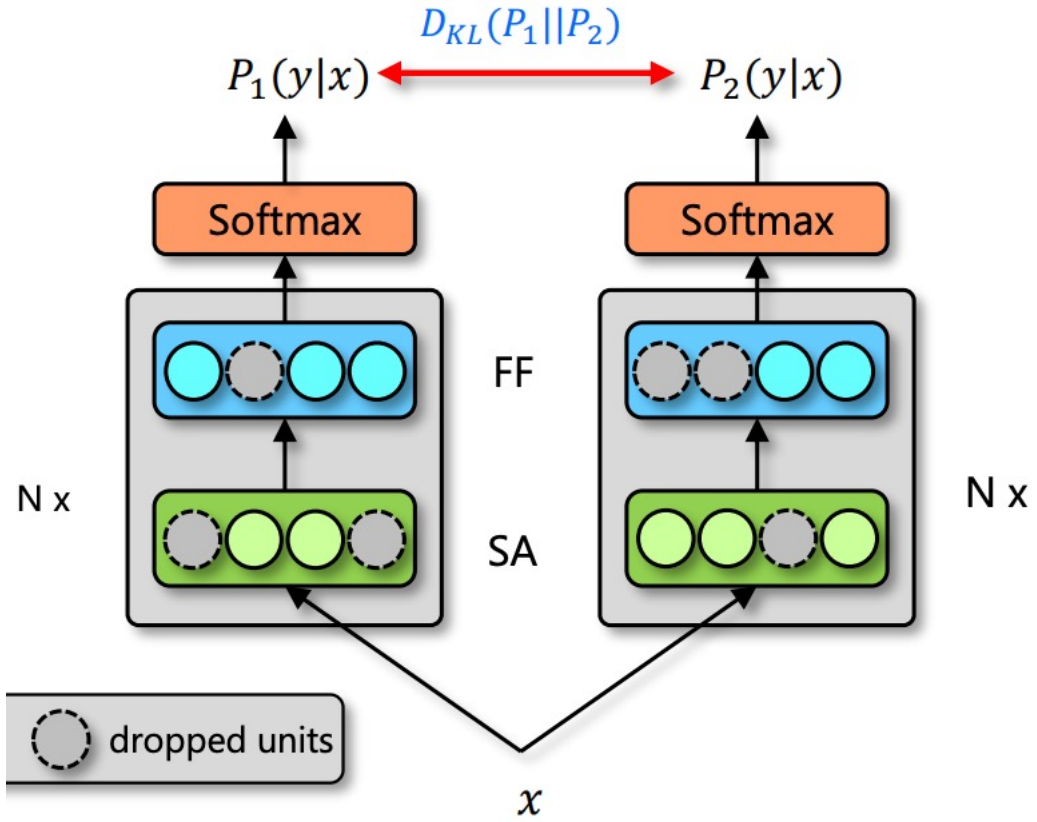


(a) Edge Dropout **(b) Random Walk**

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

Background - Contrastive Learning

- Examples for self-supervised signal via dropout



Liang et al. R-Drop: Regularized Dropout for Neural Networks. NeurIPS 2021.

Background - Contrastive Learning

- Examples for self-supervised signal in recommendation?
  - Directly transfer existing methods 🙅‍♂️
  - Self-supervised signals specifically for RecSys 🤔
    - easily obtained;
    - valuable;

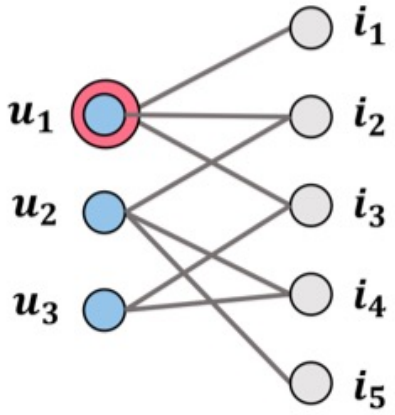
Improving Graph Collaborative Filtering with Neighborhood-enriched Contrastive Learning.

Zihan Lin*, Changxin Tian*, Yupeng Hou*, Wayne Xin Zhao✉.

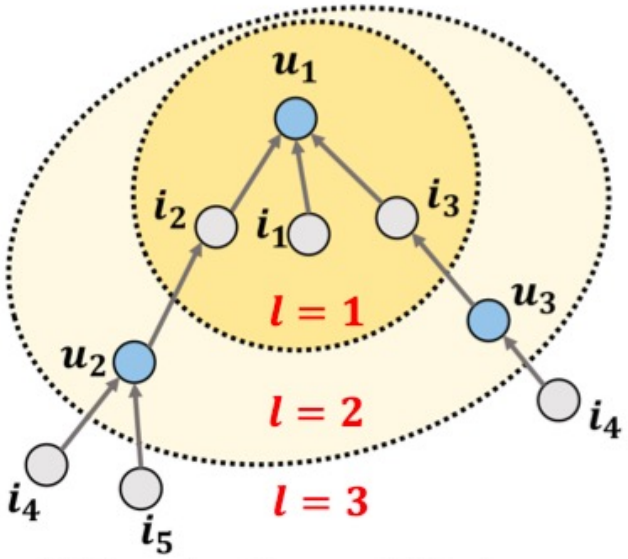
TheWebConf 2022. (*equal contribution)

Background - Graph Collaborative Filtering

(General Recommendation)



User-Item Interaction Graph



High-order Connectivity for u_1

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

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

LightGCN

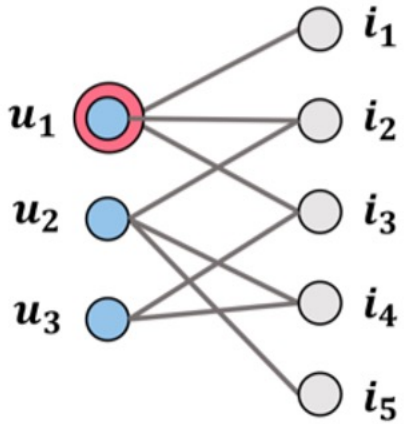
Wang et al. Neural Graph Collaborative Filtering. SIGIR 2019.

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

Challenge

- U-I graph is usually sparse or noisy;

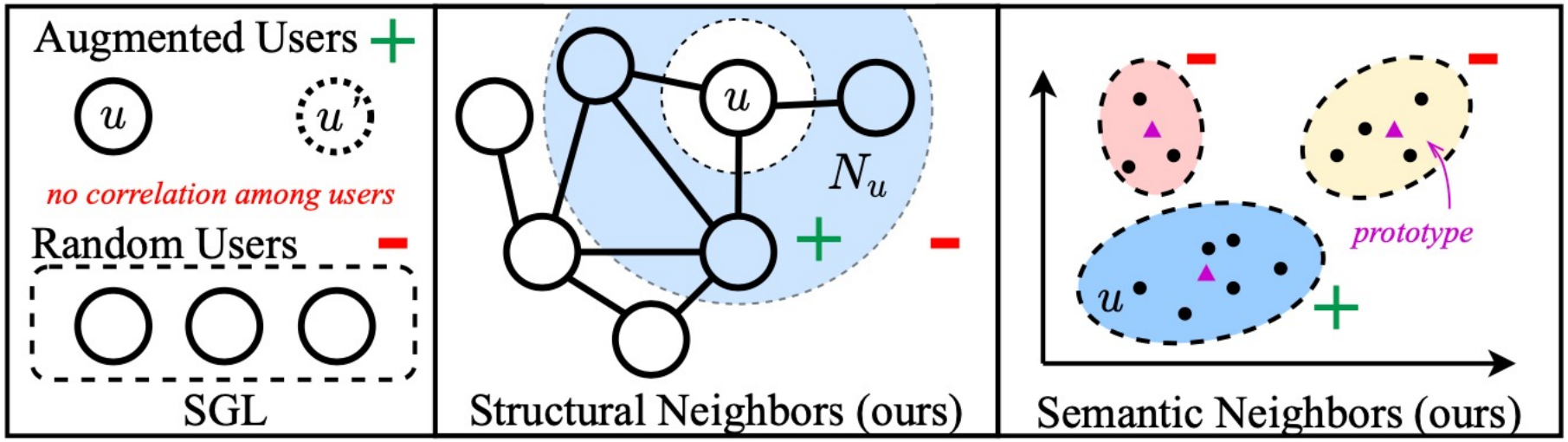
- Lack of explicitly modeling high-order constraints on U-I graph; (e.g. U-U, or I-I)



User-Item Interaction Graph

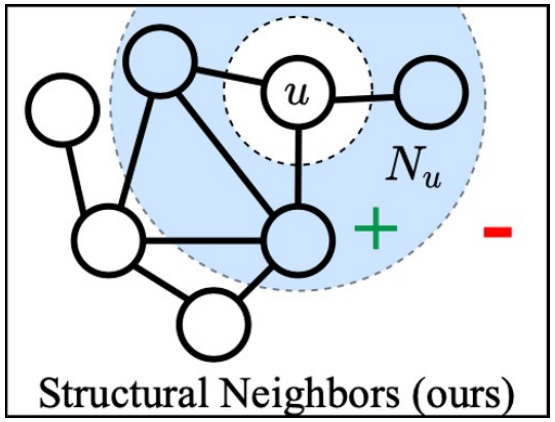
Idea

- high-order constraints (neighbors) as CL supervision signal
- -> alleviating sparsity and noisy issue;

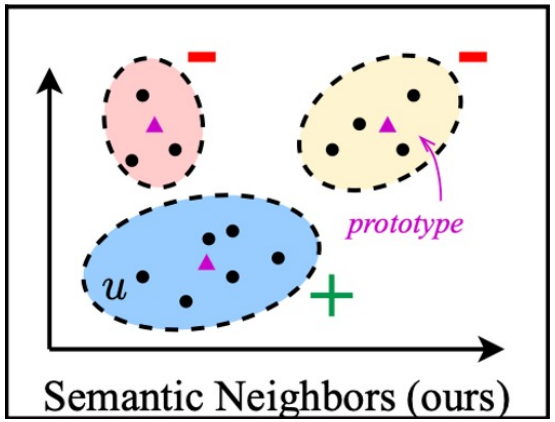


Neighborhood-enriched CL

- Structural Neighbors

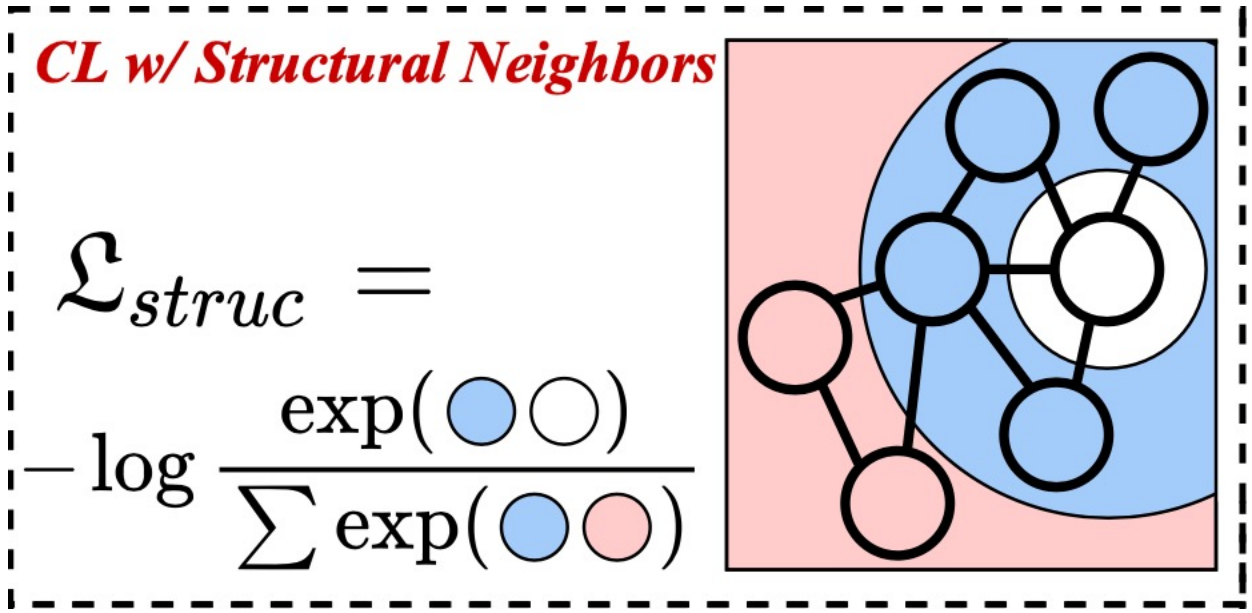


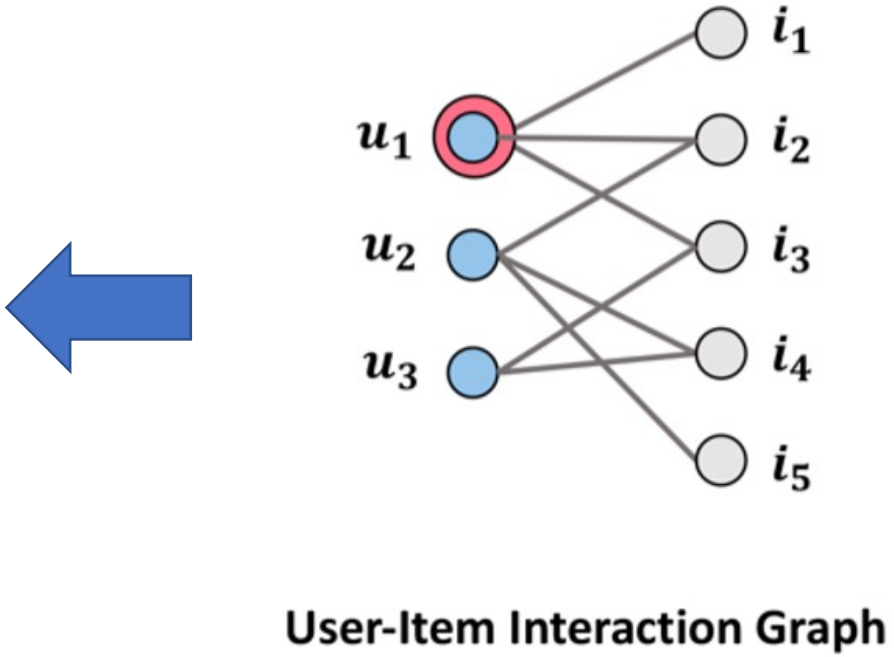
- Semantic Neighbors



Structural Neighbors

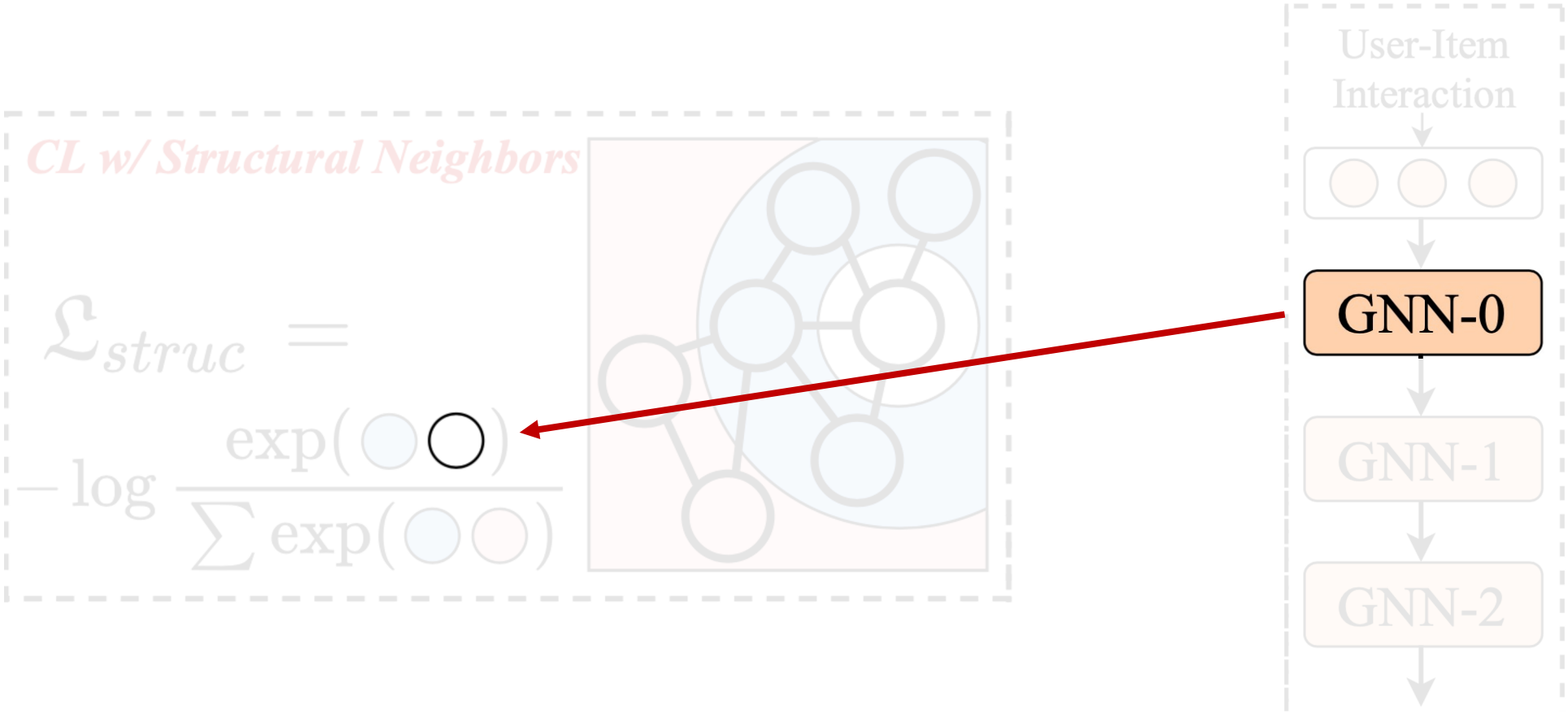
- Structurally connected nodes by high-order paths





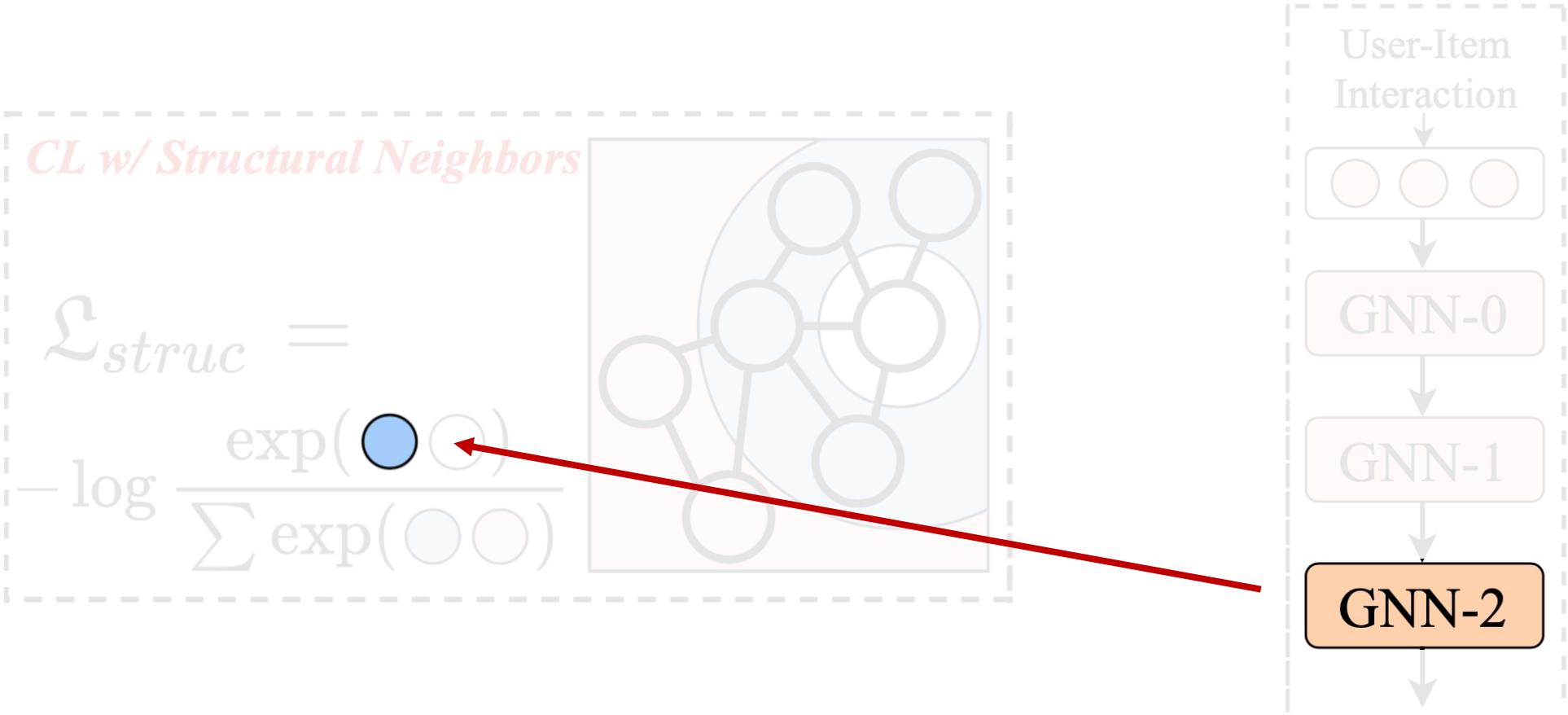
Structural Neighbors

- How to efficiently obtain high-order neighbors' representations?



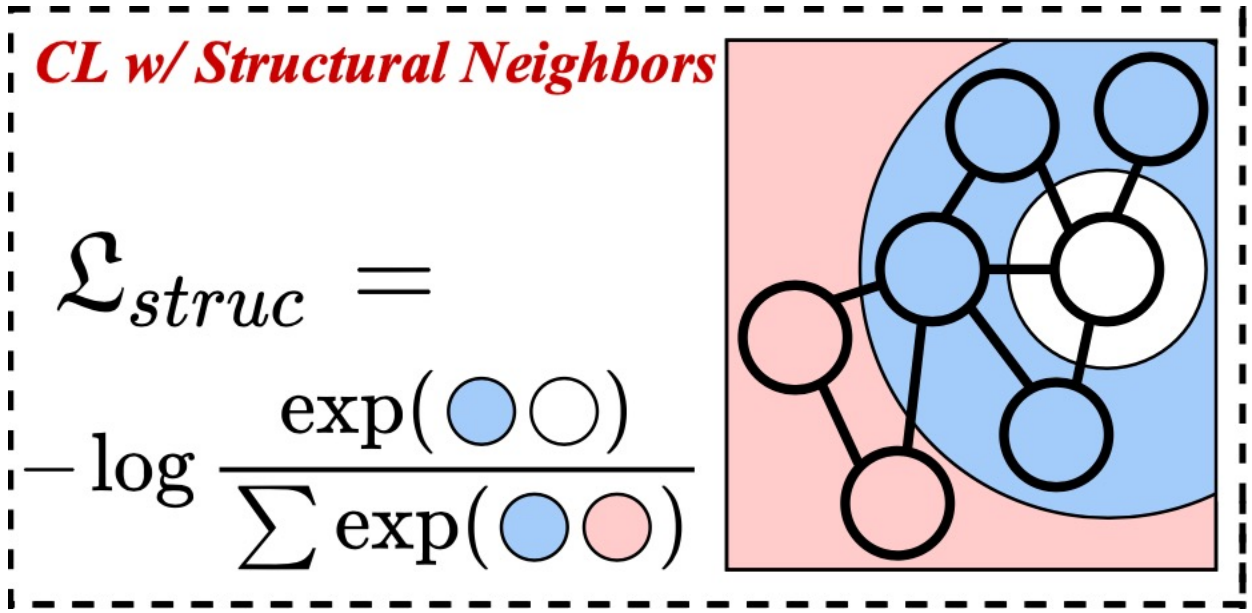
Structural Neighbors

- How to efficiently obtain high-order neighbors' representations?



Structural Neighbors

- How to efficiently obtain high-order neighbors' representations?

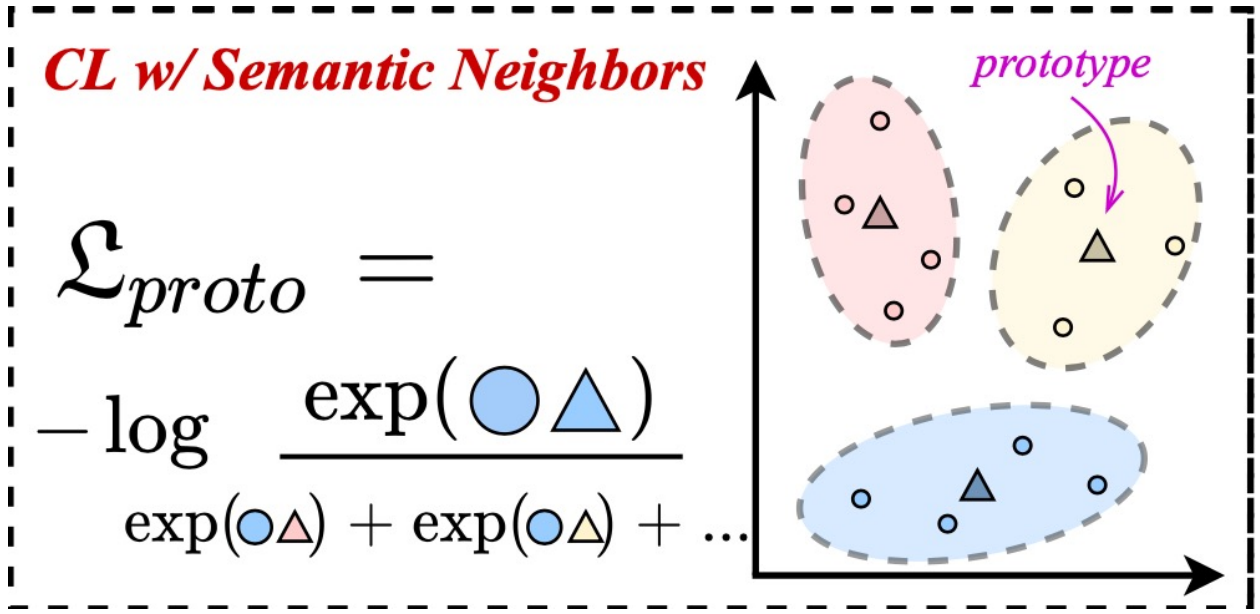


$$\mathcal{L}_S^U = \sum_{u \in \mathcal{U}} -\log \frac{\exp((\mathbf{z}_u^{(k)} \cdot \mathbf{z}_u^{(0)} / \tau))}{\sum_{v \in \mathcal{U}} \exp((\mathbf{z}_u^{(k)} \cdot \mathbf{z}_v^{(0)} / \tau))},$$

$$\mathcal{L}_S = \mathcal{L}_S^U + \alpha \mathcal{L}_S^I.$$

Semantic Neighbors

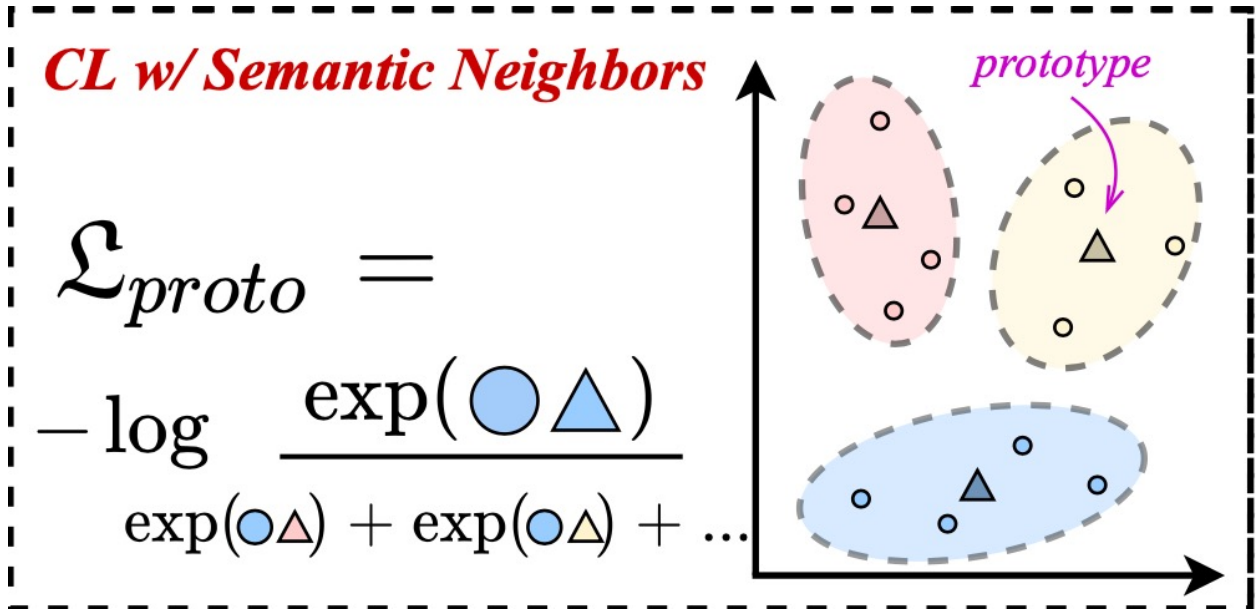
- Semantically similar neighbors which may not be directly reachable on graphs



Neighbors in embedding space

Semantic Neighbors

- Semantically similar neighbors which may not be directly reachable on graphs

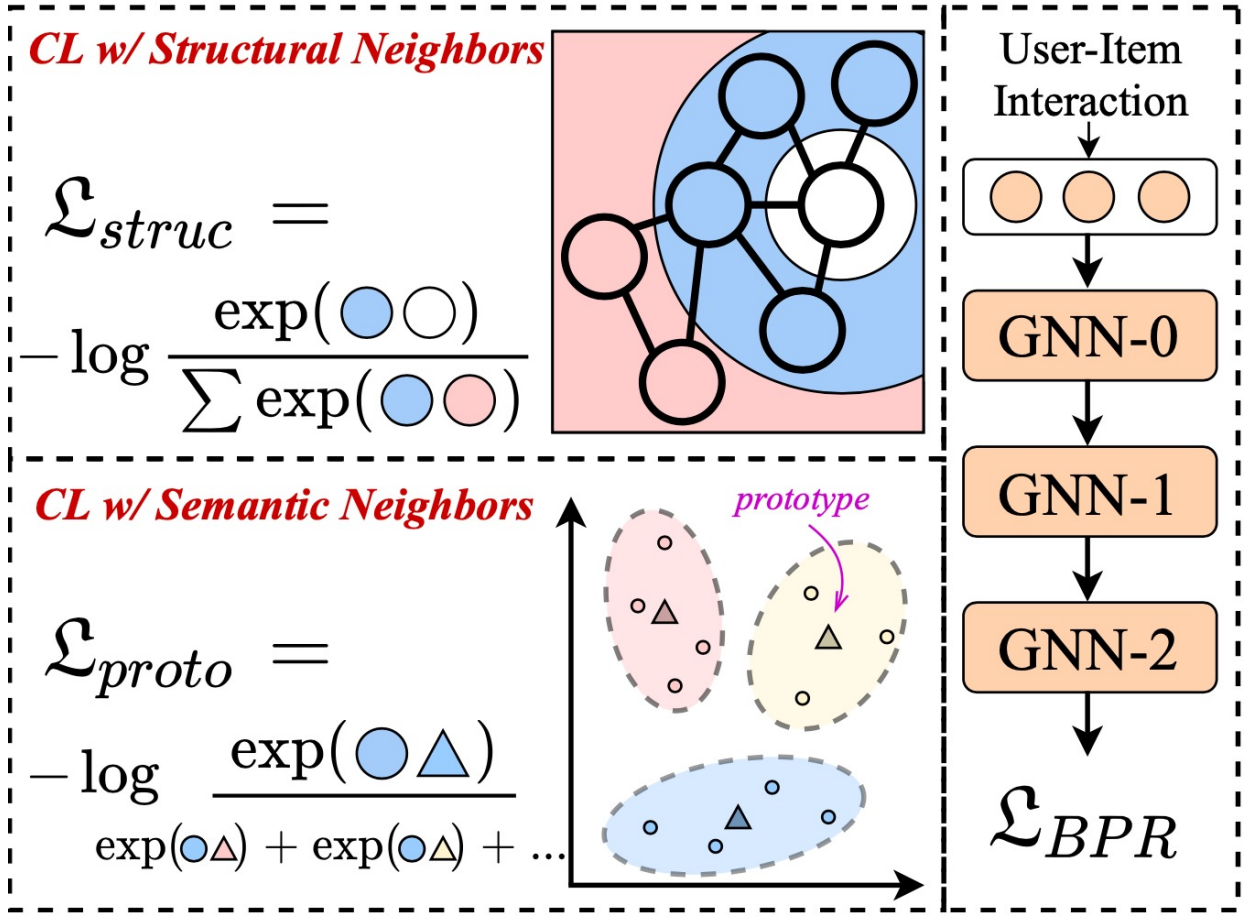


$$\mathcal{L}_P^U = \sum_{u \in \mathcal{U}} -\log \frac{\exp(\mathbf{e}_u \cdot \mathbf{c}_i / \tau)}{\sum_{\mathbf{c}_j \in C} \exp(\mathbf{e}_u \cdot \mathbf{c}_j / \tau)}.$$

$$\mathcal{L}_P = \mathcal{L}_P^U + \alpha \mathcal{L}_P^I.$$

K-means
(optimized via EM)

NCL overall



$$\mathcal{L} = \mathcal{L}_{BPR} + \lambda_1 \mathcal{L}_S + \lambda_2 \mathcal{L}_P + \lambda_3 ||\Theta||_2,$$

arbitrary GCF algorithm

NCL

NCL experiments

- 5 widely-used public datasets

Datasets	#Users	#Items	#Interactions	Density
ML-1M	6,040	3,629	836,478	0.03816
Yelp	45,478	30,709	1,777,765	0.00127
Books	58,145	58,052	2,517,437	0.00075
Gowalla	29,859	40,989	1,027,464	0.00084
Alibaba	300,000	81,614	1,607,813	0.00007

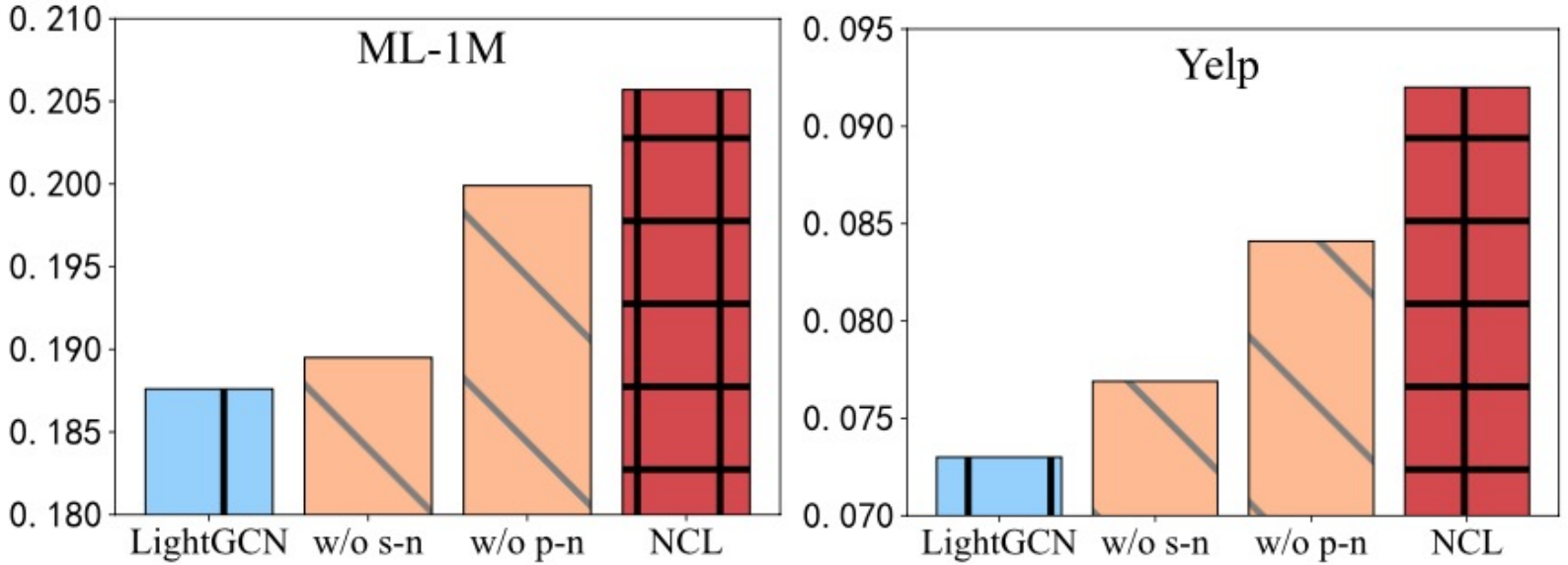
- Carefully hyper-parameter tuning for all baselines
- https://github.com/RUCAIBox/NCL

NCL experiments (1)

Dataset	Metric	BPRMF	NeuMF	FISM	NGCF	MultiGCCF	DGCF	LightGCN	SGL	NCL	Improv.
MovieLens-1M	Recall@10	0.1804	0.1657	0.1887	0.1846	0.1830	0.1881	0.1876	0.1888	**0.2057***	+8.95%
	NDCG@10	0.2463	0.2295	0.2494	0.2528	0.2510	0.2520	0.2514	0.2526	**0.2732***	+8.07%
	Recall@20	0.2714	0.2520	0.2798	0.2741	0.2759	0.2779	0.2796	0.2848	**0.3037***	+6.63%
	NDCG@20	0.2569	0.2400	0.2607	0.2614	0.2617	0.2615	0.2620	0.2649	**0.2843***	+7.32%
	Recall@50	0.4300	0.4122	0.4421	0.4341	0.4364	0.4424	0.4469	0.4487	**0.4686***	+4.44%
	NDCG@50	0.3014	0.2851	0.3078	0.3055	0.3056	0.3078	0.3091	0.3111	**0.3300***	+6.08%
Yelp	Recall@10	0.0643	0.0531	0.0714	0.0630	0.0646	0.0723	0.0730	0.0833	**0.0920***	+10.44%
	NDCG@10	0.0458	0.0377	0.0510	0.0446	0.0450	0.0514	0.0520	0.0601	**0.0678***	+12.81%
	Recall@20	0.1043	0.0885	0.1119	0.1026	0.1053	0.1135	0.1163	0.1288	**0.1377***	+6.91%
	NDCG@20	0.0580	0.0486	0.0636	0.0567	0.0575	0.0641	0.0652	0.0739	**0.0817***	+10.55%
	Recall@50	0.1862	0.1654	0.1963	0.1864	0.1882	0.1989	0.2016	0.2140	**0.2247***	+5.00%
	NDCG@50	0.0793	0.0685	0.0856	0.0784	0.0790	0.0862	0.0875	0.0964	**0.1046***	+8.51%
Amazon-Books	Recall@10	0.0607	0.0507	0.0721	0.0617	0.0625	0.0737	0.0797	0.0898	**0.0933***	+3.90%
	NDCG@10	0.043	0.0351	0.0504	0.0427	0.0433	0.0521	0.0565	0.0645	**0.0679***	+5.27%
	Recall@20	0.0956	0.0823	0.1099	0.0978	0.0991	0.1128	0.1206	0.1331	**0.1381***	+3.76%
	NDCG@20	0.0537	0.0447	0.0622	0.0537	0.0545	0.064	0.0689	0.0777	**0.0815***	+4.89%
	Recall@50	0.1681	0.1447	0.183	0.1699	0.1688	0.1908	0.2012	0.2157	**0.2175***	+0.83%
	NDCG@50	0.0726	0.061	0.0815	0.0725	0.0727	0.0843	0.0899	0.0992	**0.1024***	+3.23%
Gowalla	Recall@10	0.1158	0.1039	0.1081	0.1192	0.1108	0.1252	0.1362	0.1465	**0.1500***	+2.39%
	NDCG@10	0.0833	0.0731	0.0755	0.0852	0.0791	0.0902	0.0876	0.1048	**0.1082***	+3.24%
	Recall@20	0.1695	0.1535	0.1620	0.1755	0.1626	0.1829	0.1976	0.2084	**0.2133***	+2.35%
	NDCG@20	0.0988	0.0873	0.0913	0.1013	0.0940	0.1066	0.1152	0.1225	**0.1265***	+3.27%
	Recall@50	0.2756	0.2510	0.2673	0.2811	0.2631	0.2877	0.3044	0.3197	**0.3259***	+1.94%
	NDCG@50	0.1450	0.1110	0.1169	0.1270	0.1184	0.1322	0.1414	0.1497	**0.1542***	+3.01%
Alibaba-iFashion	Recall@10	0.303	0.182	0.0357	0.0382	0.0401	0.0447	0.0457	0.0461	**0.0477***	+3.47%
	NDCG@10	0.0161	0.0092	0.0190	0.0198	0.0207	0.0241	0.0246	0.0248	**0.0259***	+4.44%
	Recall@20	0.0467	0.0302	0.0553	0.0615	0.0634	0.0677	0.0692	0.0692	**0.0713***	+3.03%
	NDCG@20	0.0203	0.0123	0.0239	0.0257	0.0266	0.0299	0.0246	0.0307	**0.0319***	+3.01%
	Recall@50	0.0799	0.0576	0.0943	0.1081	0.1107	0.1120	0.1144	0.1141	**0.1165***	+1.84%
	NDCG@50	0.0269	0.0177	0.0317	0.0349	0.0360	0.0387	0.0396	0.0396	**0.0409***	+3.28%

NCL experiments (2)

- Ablation Study



NCL experiments (3)

- Impact of Data Sparsity Levels

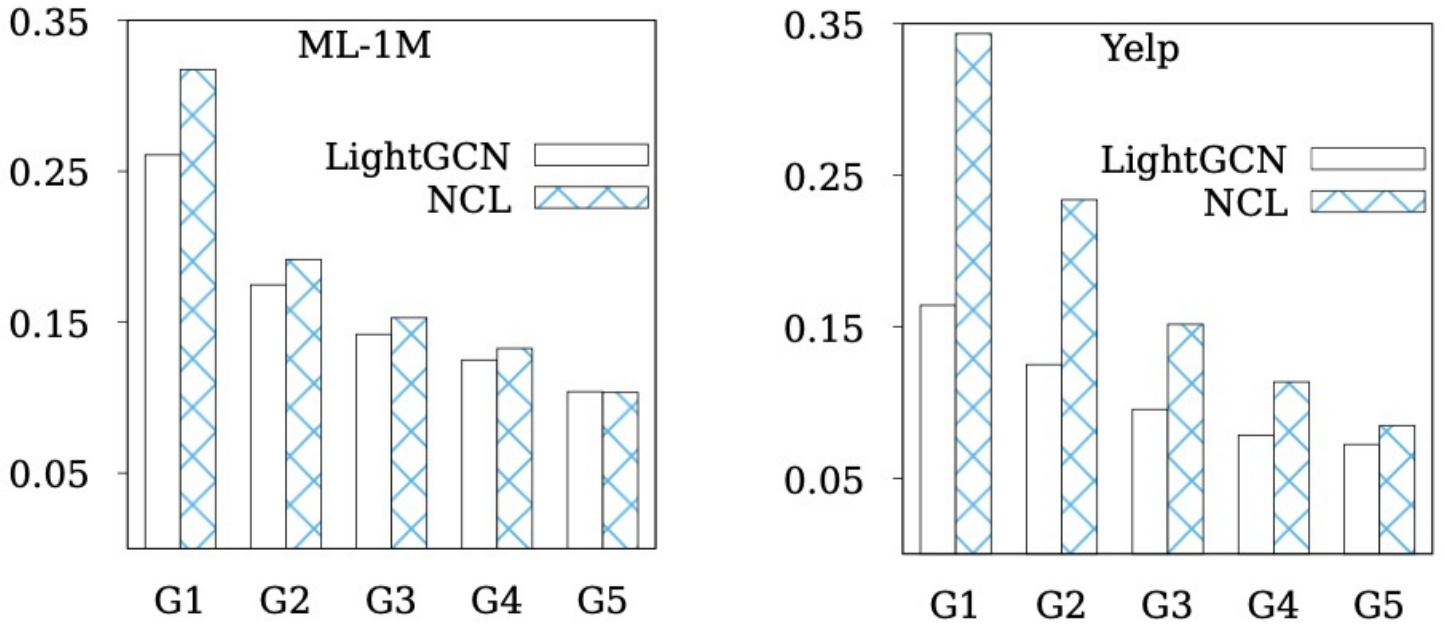


Figure 4: Performance analysis for different sparsity-level users (Recall@10). $G1$ denotes the group of users with the lowest average number of interactions.

NCL experiments (4)

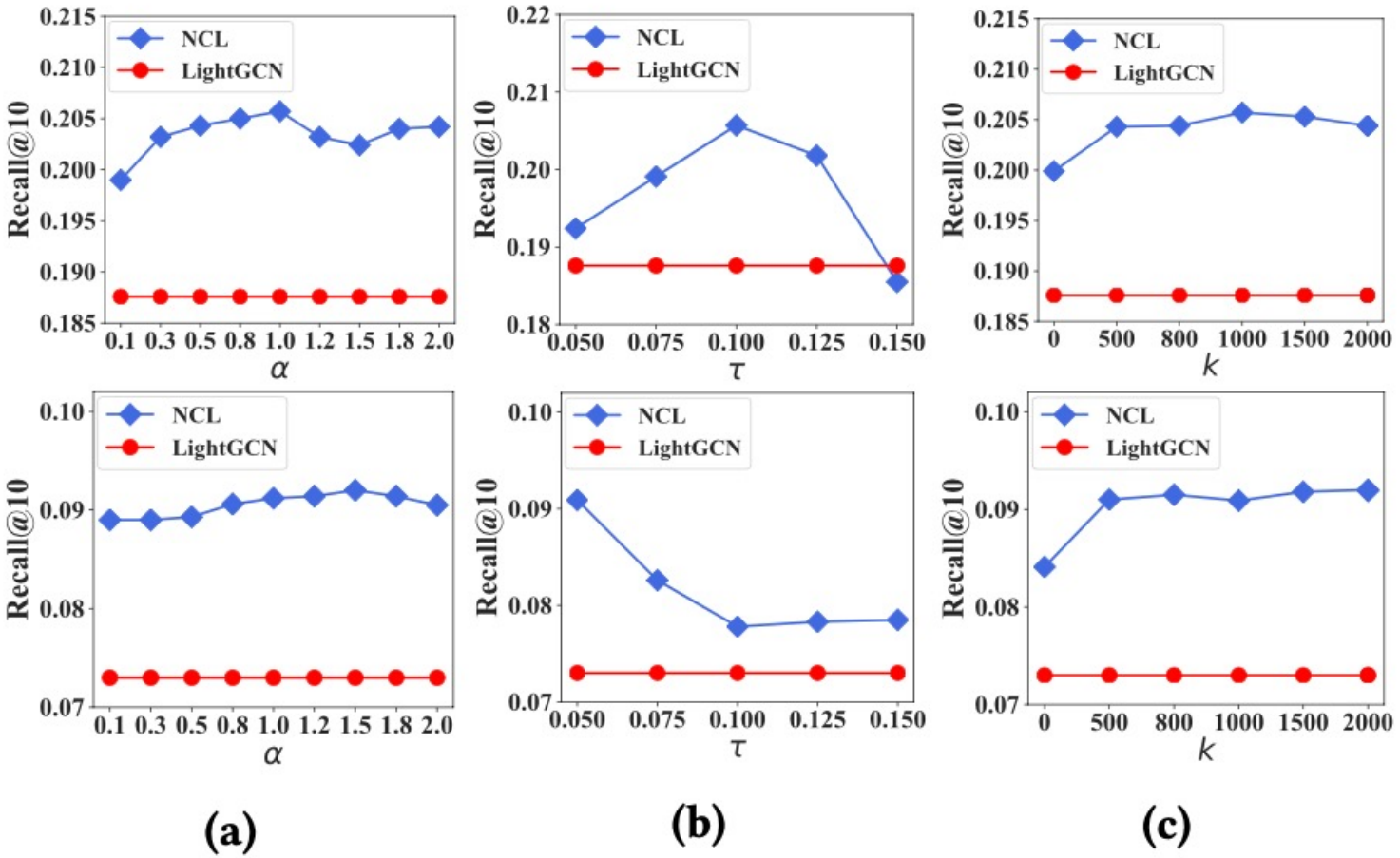
- Effect of Structural Neighbors

Table 3: Performance comparison w.r.t. different hop of structural neighbors.

Hop	MovieLens-1M		Yelp	
	Recall@10	NDCG@10	Recall@10	NDCG@10
w/o s-n	0.1876	0.2514	0.0730	0.0520
1	0.2057	0.2732	0.0920	0.0678
2	0.1838	0.2516	0.0837	0.0602
3	0.1839	0.2507	0.0787	0.0557

NCL experiments (5)

- Hyper-parameter Tuning



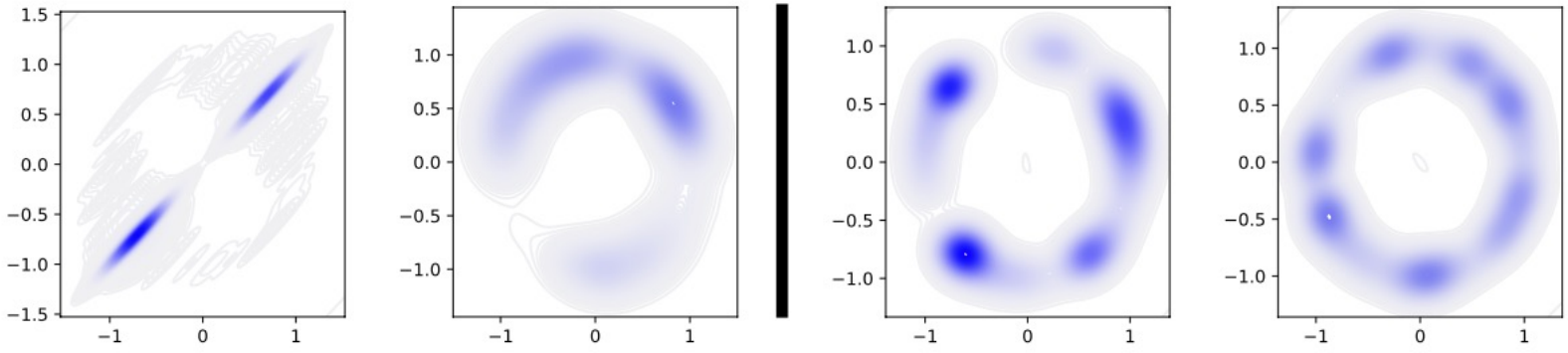
NCL experiments (6)

- Applying NCL on Other GNN Backbones

Method	MovieLens-1M		Yelp	
	Recall@10	NDCG@10	Recall@10	NDCG@10
NGCF	0.1846	0.2528	0.0630	0.0446
+NCL	**0.1852**	**0.2542**	**0.0663**	**0.0465**
DGCF	0.1853	0.2500	0.0723	0.0514
+NCL	**0.1877**	**0.2522**	**0.0739**	**0.0528**
LightGCN	0.1888	0.2526	0.0833	0.0601
+NCL	**0.2057**	**0.2732**	**0.0920**	**0.0678**

NCL experiments (7)

- Visualizing the Distribution of Representations



(a) LightGCN (b) NCL (c) LightGCN (d) NCL

Figure 6: Visualization of item embeddings. Items from ML-1M and Yelp are illustrated in (a), (b) and (c), (d), respectively.

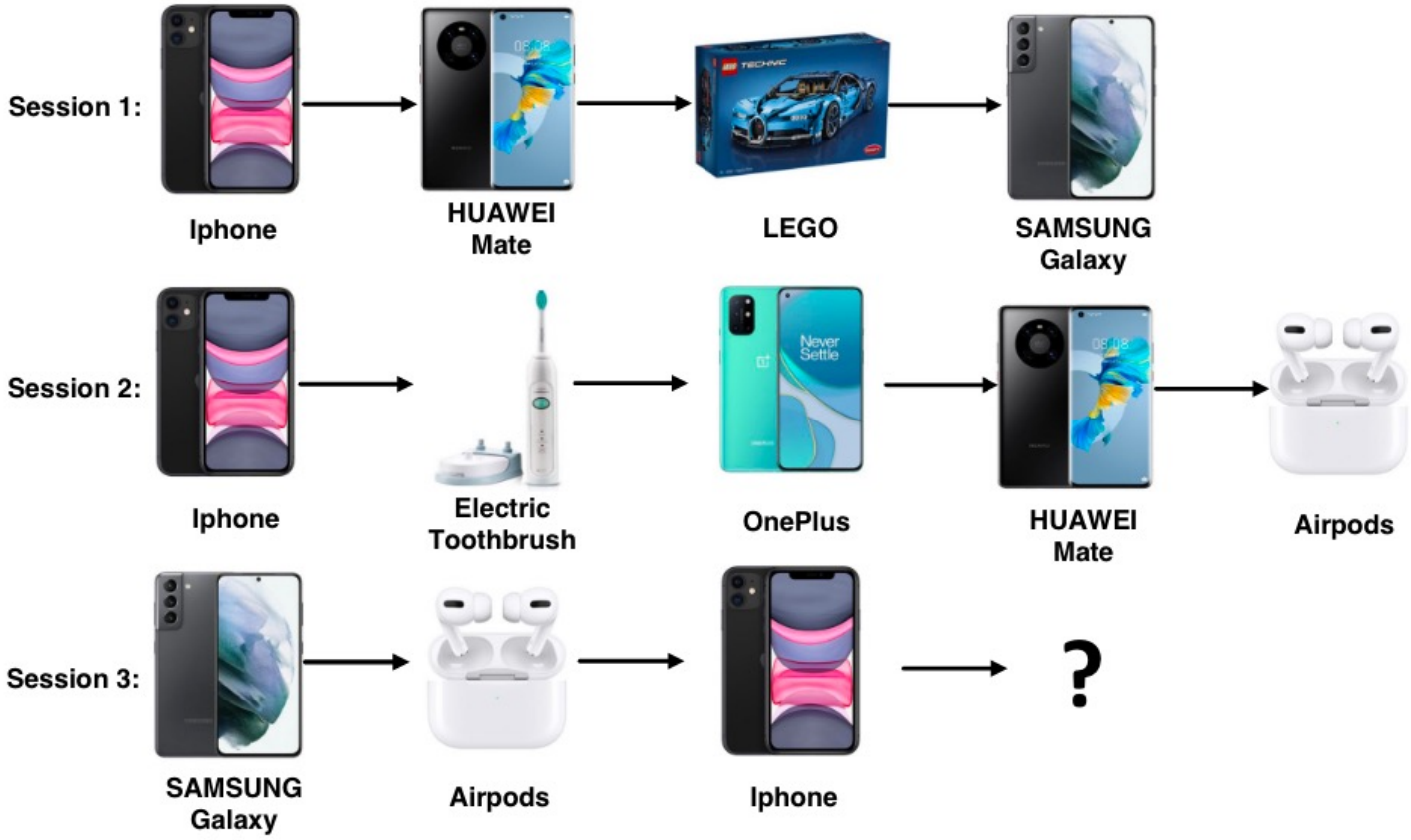
CORE:
Simple and Effective Session-based Recommendation within Consistent Representation Space

Yupeng Hou, Binbin Hu, Zhiqiang Zhang, Wayne Xin Zhao✉.

SIGIR 2022, short paper.

Background - Session-based Rec

- Next-item prediction;
- Anonymous sessions;
- Short-term Interest;



Background - Session-based Rec

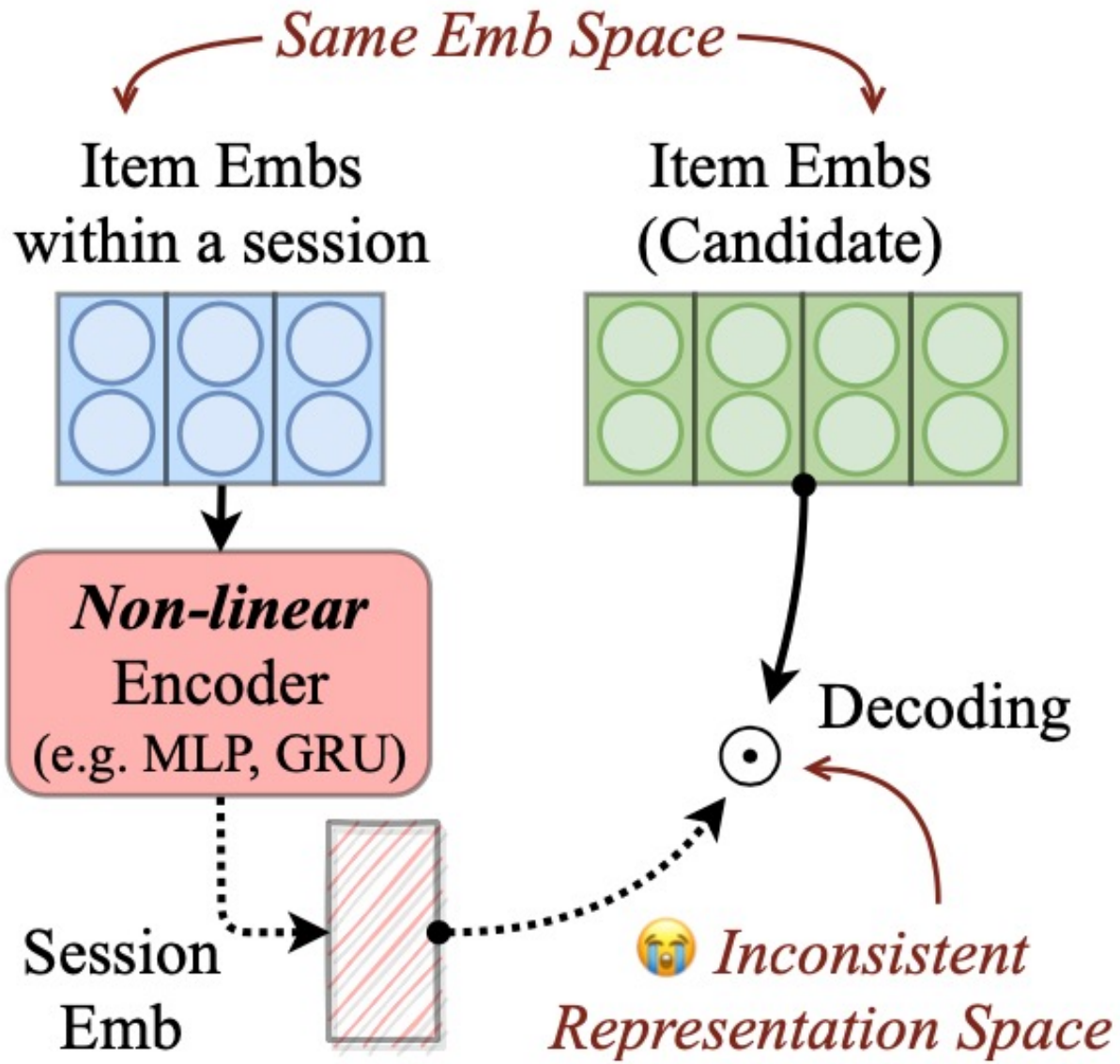
- Existing studys - Fancy and complex session encoders

- RNN, CNN, GNN
- RNN + Attention, CNN + Attention, GNN + Attention
- Transformers
- GNN + Transformers

... ... 🥲

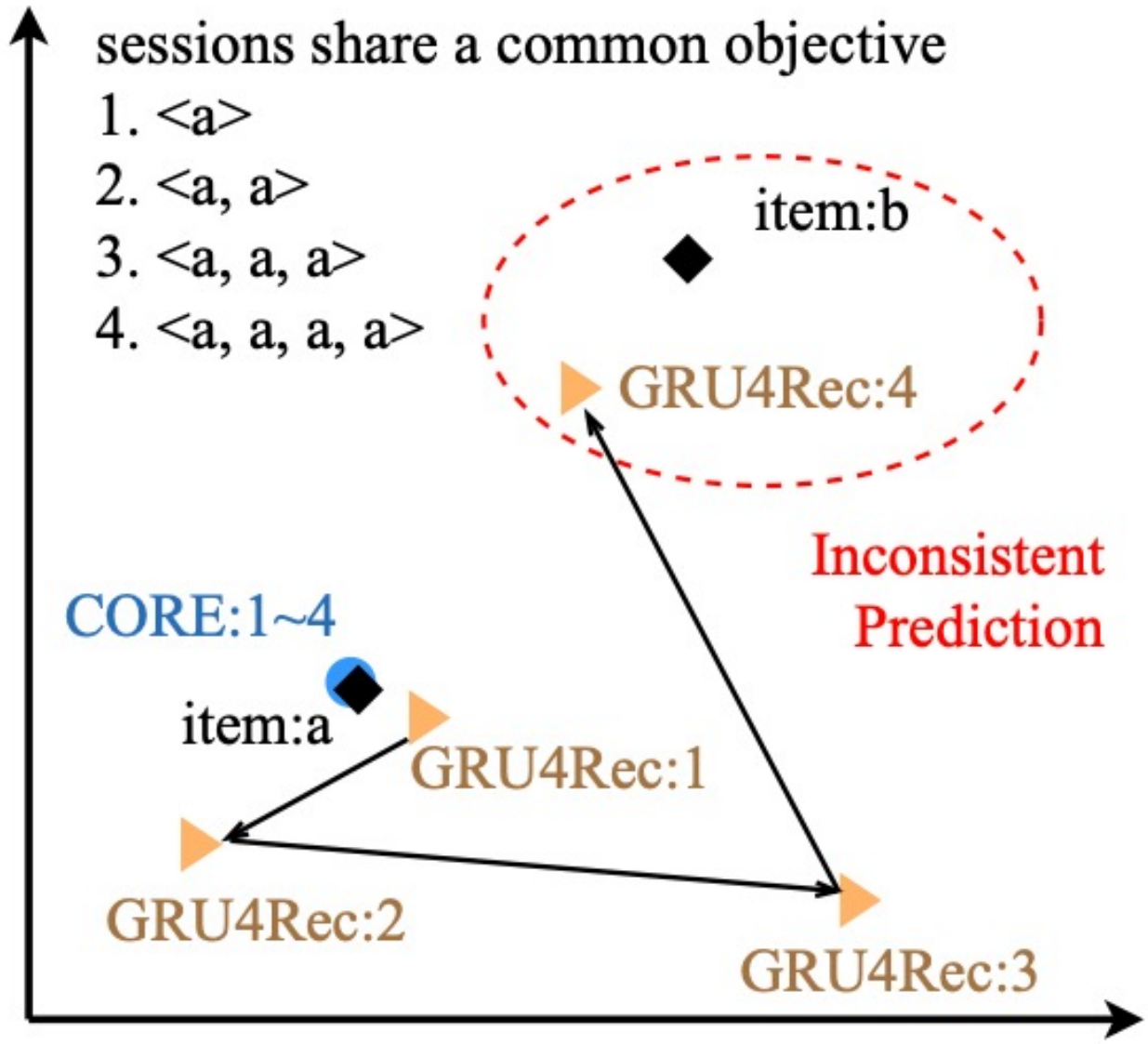
Observation

- Encoder-Decoder



Issue

- Inconsistent Prediction
- (a toy example)



Idea

- What if encoding-decoding in consistent representation space mandatorily? 🤔
- Basically, linear combination as encoder 💡

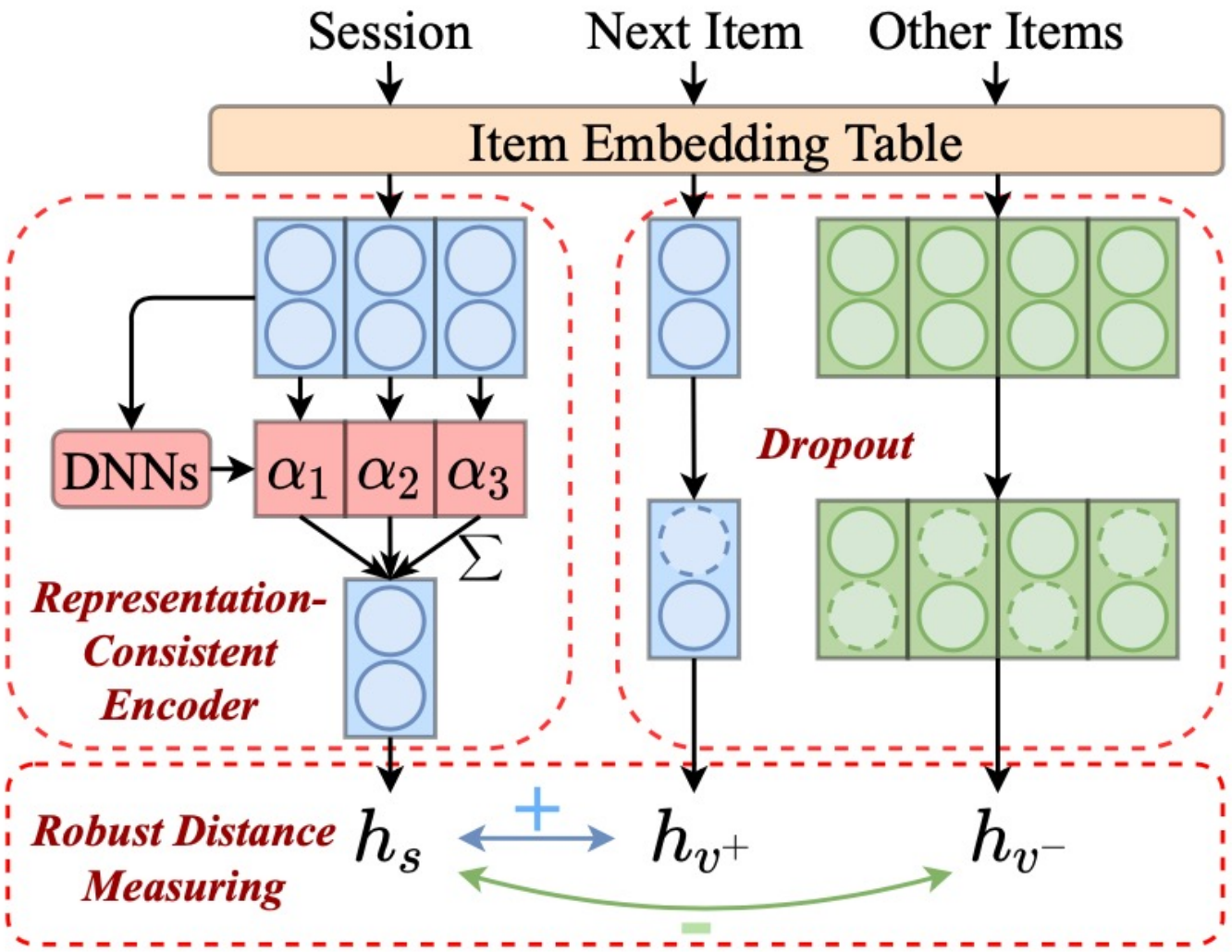
Challenge

- Strong power of DNNs + consistent representation space;

- Prevent overfitting of item embeddings;
  (in consistent representation space)

COnsistent REpresentation - RCE

(Representation-Consistent Encoder)



$$\boldsymbol{\alpha} = \mathrm{DNNs}([\boldsymbol{h}_{s,1}; \boldsymbol{h}_{s,2}; \ldots; \boldsymbol{h}_{s,n}])$$

$$\boldsymbol{h}_s = \sum_{i=1}^{n} \alpha_i \boldsymbol{h}_{s,i}.$$

DNNs can be:

- Pooling;
- Transformers;
- ... ...

COnsistent REpresentation - RDM

(Robust Distance Measuring)

- Traditional cross-entropy loss

$$\ell_{\text{ori}} = -\log \frac{\exp(\boldsymbol{h}_s \cdot \boldsymbol{h}_{v^+})}{\sum_{i=1}^{m} \exp(\boldsymbol{h}_s \cdot \boldsymbol{h}_{v_i})}$$

$$\propto \sum_{v^- \in \mathcal{V} \setminus \{v^+\}} \left(\|\boldsymbol{h}_s - \boldsymbol{h}_{v^+}\|^2 - \|\boldsymbol{h}_s - \boldsymbol{h}_{v^-}\|^2 + 2 \right).$$

$(N-1)$ −tuplet loss with L2-distance & fixed margin 2

COnsistent REpresentation - RDM

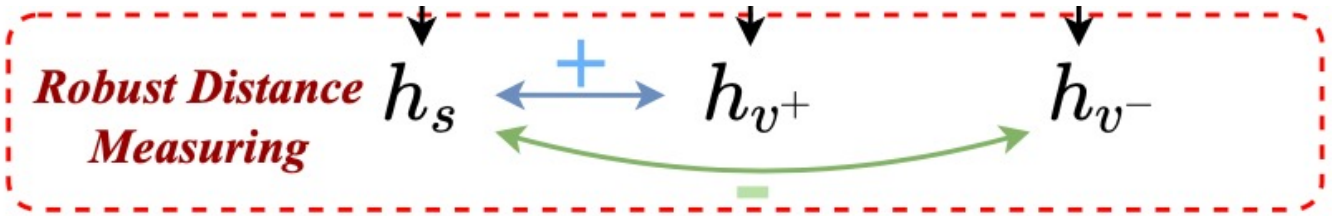
(Robust Distance Measuring)

$(N-1)$ −tuplet loss with L2-distance & fixed margin 2

Dropout & cosine distance & tunable margin τ

$$\ell = -\log \frac{\exp\left(\cos(\boldsymbol{h}_s, \boldsymbol{h}'_{v^+})/\tau\right)}{\sum_{i=1}^{m} \exp\left(\cos(\boldsymbol{h}_s, \boldsymbol{h}'_{v_i})/\tau\right)},$$

(contrastive learning)
Sessions <-> Next items



CORE experiments

- 5 widely-used public datasets

Dataset	# Interactions	# Items	# Sessions	Avg. Length
Diginetica	786,582	42,862	204,532	4.12
Nowplaying	1,085,410	59,593	145,612	9.21
RetailRocket	871,637	51,428	321,032	6.40
Tmall	427,797	37,367	66,909	10.62
Yoochoose	1,434,349	19,690	470,477	4.64

- Carefully hyper-parameter tuning for all baselines
- https://anonymous.4open.science/r/CORE (temporarily)

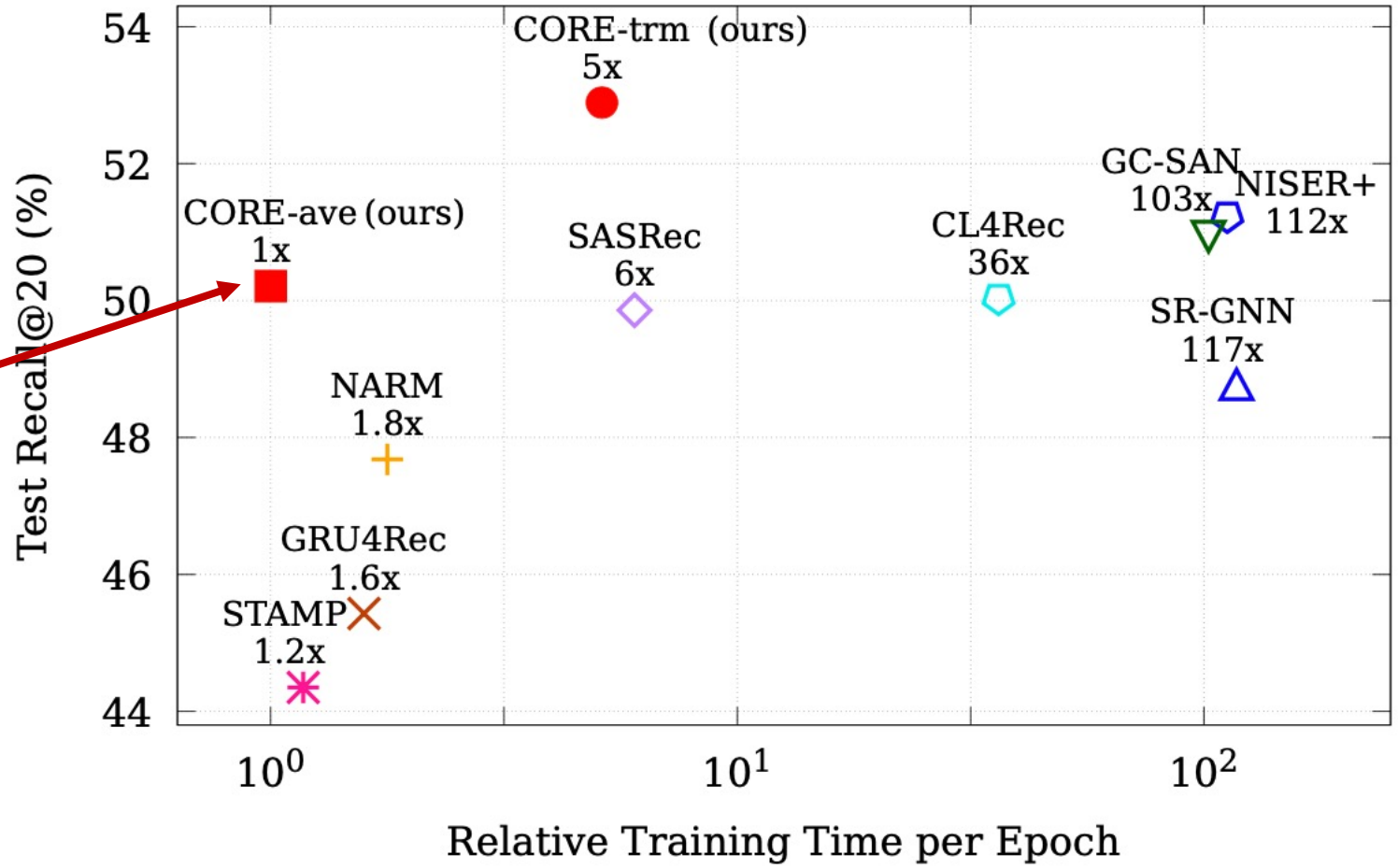
CORE experiments (1)

Dataset	Metric	FPMC	GRU4Rec	NARM	SR-GNN	NISER+	LESSR	SGNN-HN	SASRec	GC-SAN	CL4Rec	CORE-ave	CORE-trm	Improv.
Diginetica	R@20	31.83	45.43	47.68	48.76	51.23	48.80	50.89	49.86	50.95	50.03	50.21	**52.89***	+3.24%
	M@20	8.79	14.77	15.58	16.93	18.32	16.96	17.25	17.19	17.84	17.26	18.07	**18.58***	+1.42%
Nowplaying	R@20	10.18	13.80	14.17	15.28	16.55	17.60	16.75	20.69	18.30	20.59	20.31	**21.81***	+5.41%
	M@20	4.51	5.83	6.11	6.10	7.14	7.13	6.13	**8.14**	8.13	8.21	6.62	7.35	–
RetailRocket	R@20	46.04	55.32	58.65	58.71	60.36	56.22	58.82	59.81	60.18	59.69	59.18	**61.85***	+2.47%
	M@20	21.95	33.18	34.69	36.42	37.43	37.11	35.72	36.03	36.85	35.95	37.52*	**38.76***	+3.55%
Tmall	R@20	20.30	23.25	31.67	33.65	35.97	32.45	39.14	35.82	35.32	35.59	**44.67***	44.48*	+14.13%
	M@20	13.07	15.78	21.83	25.27	27.06	23.96	23.46	25.10	23.48	25.07	**31.85***	31.72*	+17.70%
Yoochoose	R@20	–	60.78	61.67	61.84	62.99	62.89	62.49	63.55	63.24	63.61	58.83	**64.61***	+1.57%
	M@20	–	27.27	27.82	28.15	28.98	28.59	28.24	28.63	**29.00**	28.73	25.05	28.24	–

CORE experiments (2)

- Efficiency

Variant with
only item embs



CORE experiments (3)

- Ablation Study

Method	Diginetica		RetailRocket	
	R@20	M@20	R@20	M@20
CORE	**52.89**	**18.58**	**61.85**	**38.76**
w/o RCE	49.82	17.41	59.59	36.27
w/o RDM	52.31	18.38	60.93	37.72
SASRec	49.86	17.19	59.81	36.03

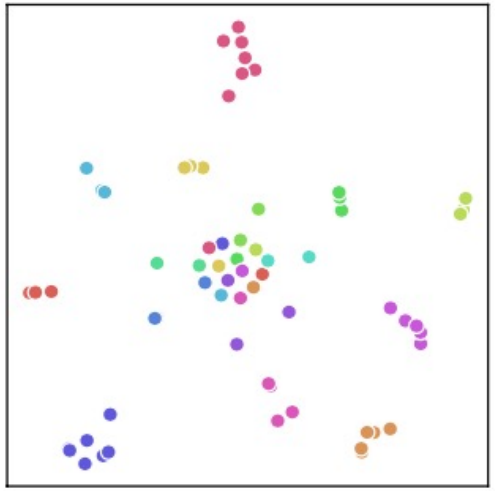
CORE experiments (4)

- Improving existing methods with RCE & RDM

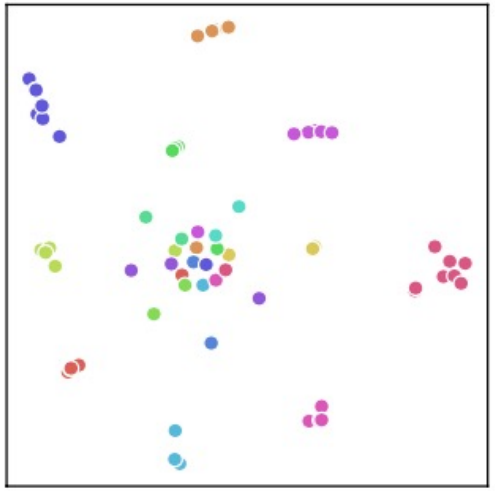
Method	Diginetica		RetailRocket	
	R@20	M@20	R@20	M@20
NARM	47.68	15.58	58.65	34.69
+ RCE	51.86	18.27	60.77	37.01
+ RDM	51.62	17.79	61.33	37.11
+ All	**52.51**	**18.58**	**62.19**	**38.84**
SR-GNN	48.76	16.93	58.71	36.42
+ RCE	49.51	17.53	57.05	35.70
+ RDM	51.36	18.57	61.41	38.27
+ All	**52.38**	**18.95**	**61.43**	**38.38**

CORE experiments (5)

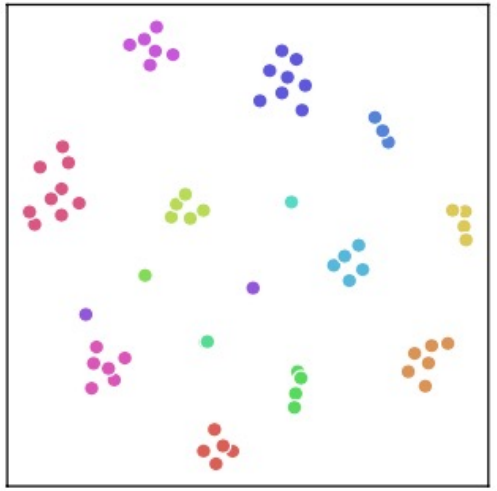
- Visualization of session embeddings
- (sessions with same next-item are in the same class)



(a) GRU4Rec



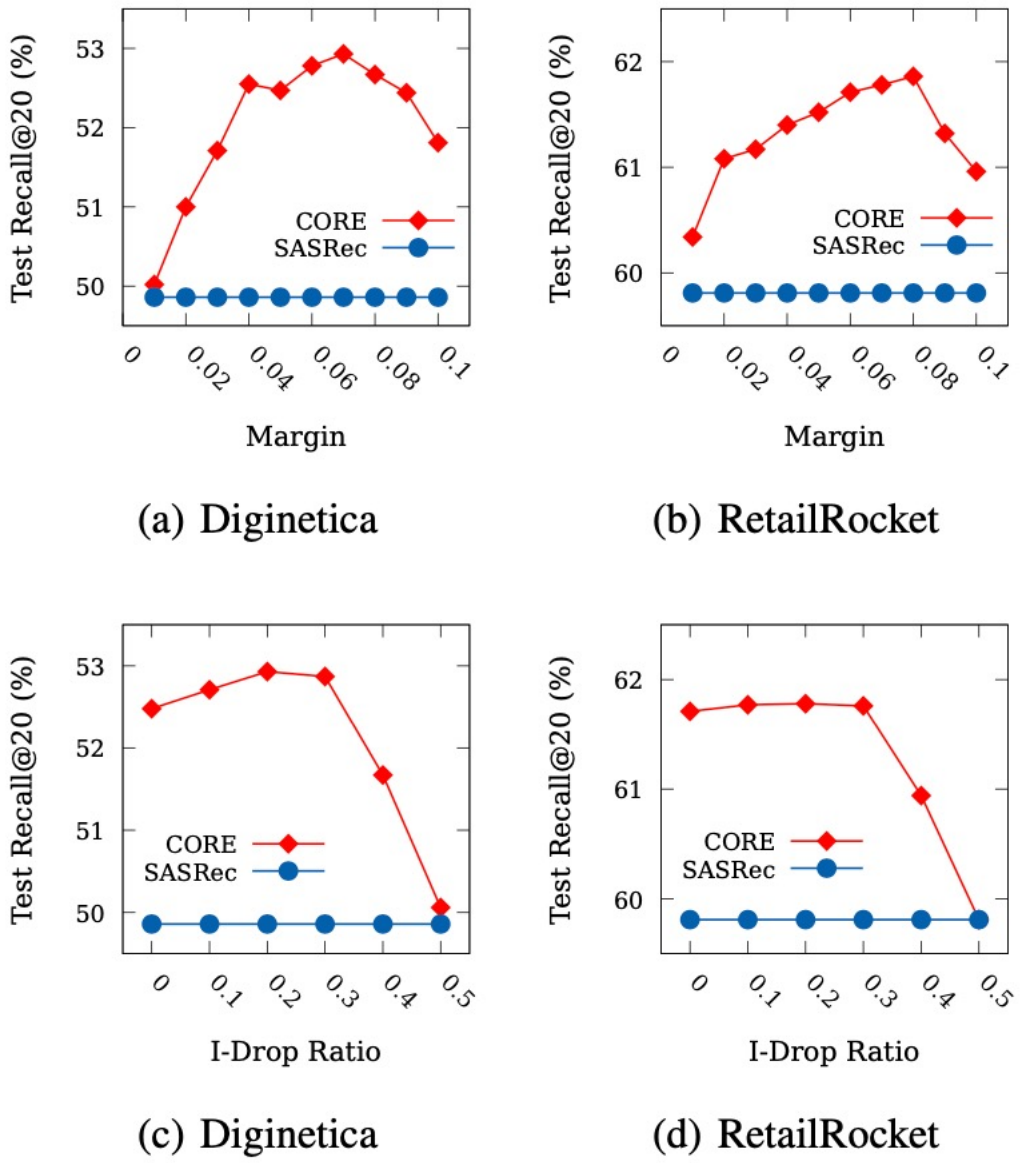
(b) SASRec



(c) CORE (ours)

CORE experiments (6)

- Hyper-parameter Tuning



Conducting RecSys Research w/

RecBole

- https://recbole.io/
- PyTorch, 78 models in 4 categories, 28 processed datasets

- One-stop solution for RecSys research 🤩

https://github.com/RUCAIBox/NCL

properties — config file

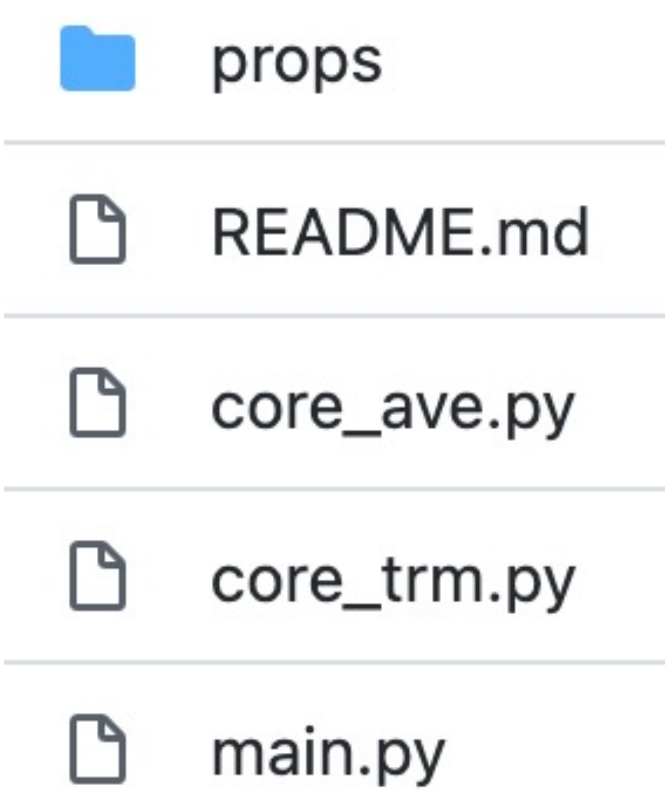
README.md

main.py

ncl.py — model with `forward`, `calculate_loss`, `predict`

trainer.py — trainer w/ EM

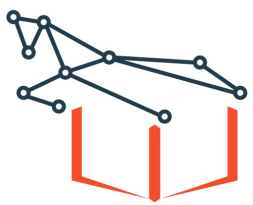
https://anonymous.4open.science/r/CORE



config file

model with
`forward`, `calculate_loss`, `predict`

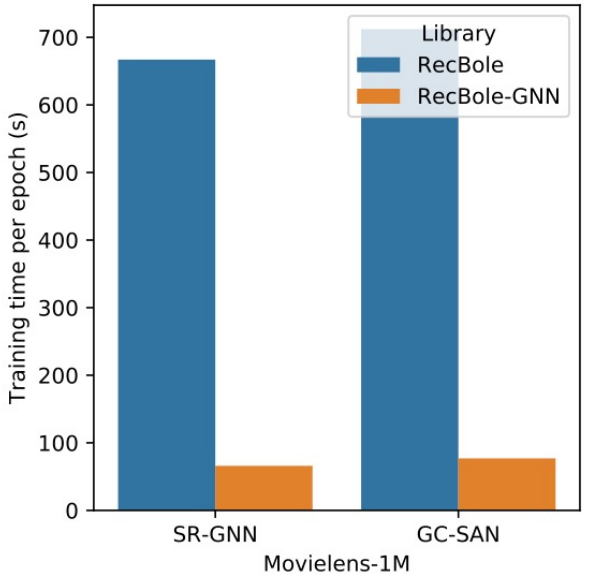
GNN-enhanced RecSys?

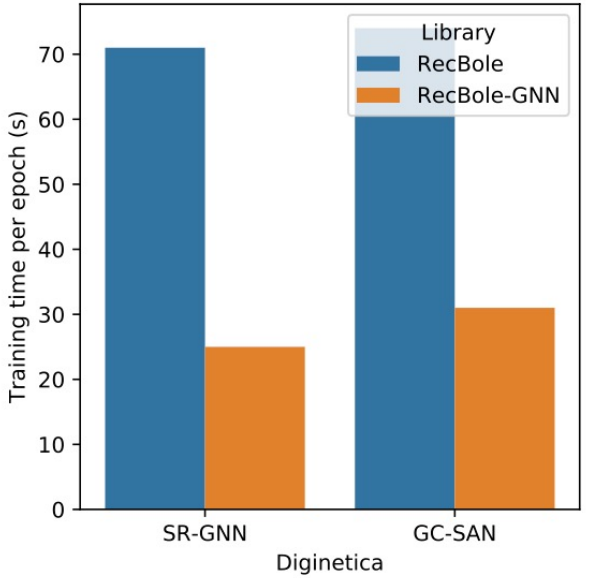


RecBole-GNN

- https://github.com/RUCAIBox/RecBole-GNN
- 15 new models!
- Leaderboards for 3 categories;
- Efficient and reusable graph processing;

- Credit to Lanling  and Changxin 



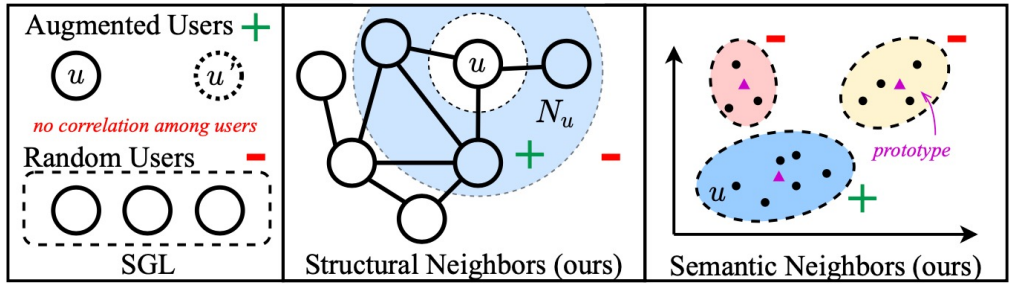


Conclusion

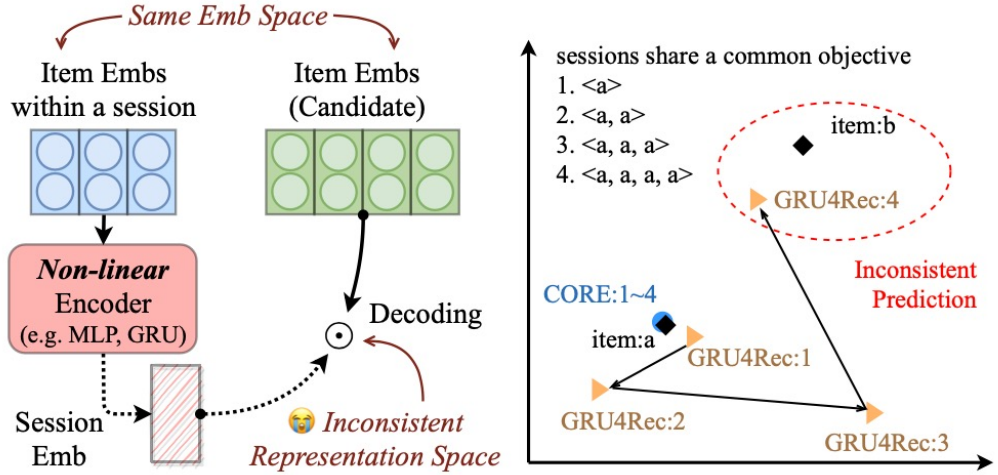
Improving recommendation via Contrastive Learning

Yupeng Hou

Conclusion & QA



- Improving RecSys via CL
  - NCL [TheWebConf 22] for graph collaborative filtering;
  - CORE [SIGIR 22 short] for session-based recommendation;



- Self-supervised signals in RecSys
  - High-order neighbors (structural & semantic);
  - Sessions' next-item;

- Conducting Research with RecBole

RecBole

RecBole-GNN