Criteria Tell you More than Ratings: Criteria Preference-Aware Light Graph Convolution for Effective Multi-Criteria Recommendation

Jin-Duk Park
Computational Science
and Engineering,
Yonsei Univ, Seoul, Korea

Siqing Li
School of Computer Science
and Engineering,
The University of New South
Wales, Australia

Xin Cao
School of Computer Science
and Engineering,
The University of New South
Wales, Australia

Won-Yong Shin
Computational Science
and Engineering,
Yonsei Univ, Seoul, Korea

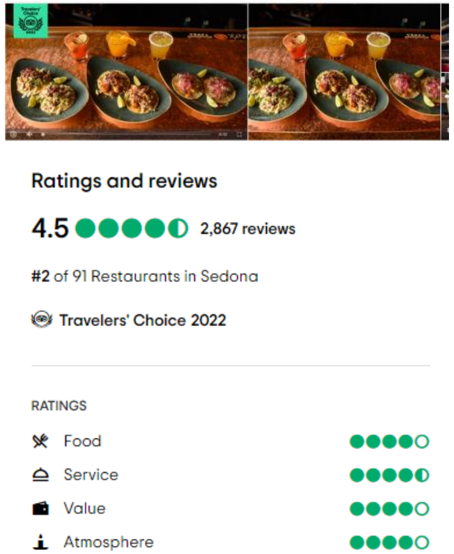
Presenter:
Jin-Duk Park

Date: 2023. 08. 09
Room 103C, Recommendation with Graph session

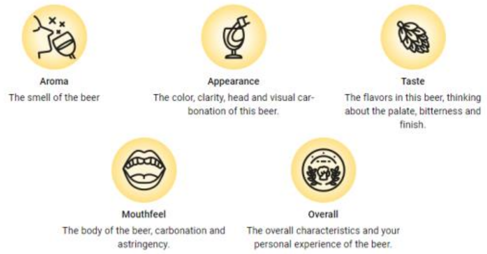
Multi-Criteria Recommender System (MCRS)

What is the Multi-criteria Recommender System (MCRS)?

Tripadvisor.com



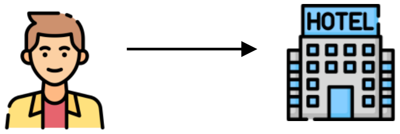
Agoda.com



RateBeer.com

Single-criterion RS

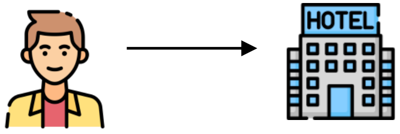
User a → Item i



R:

MCRS

User a → Item i



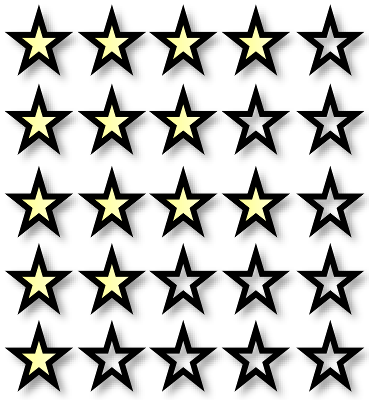
Overall R_0 :

Cleanliness R_1 :

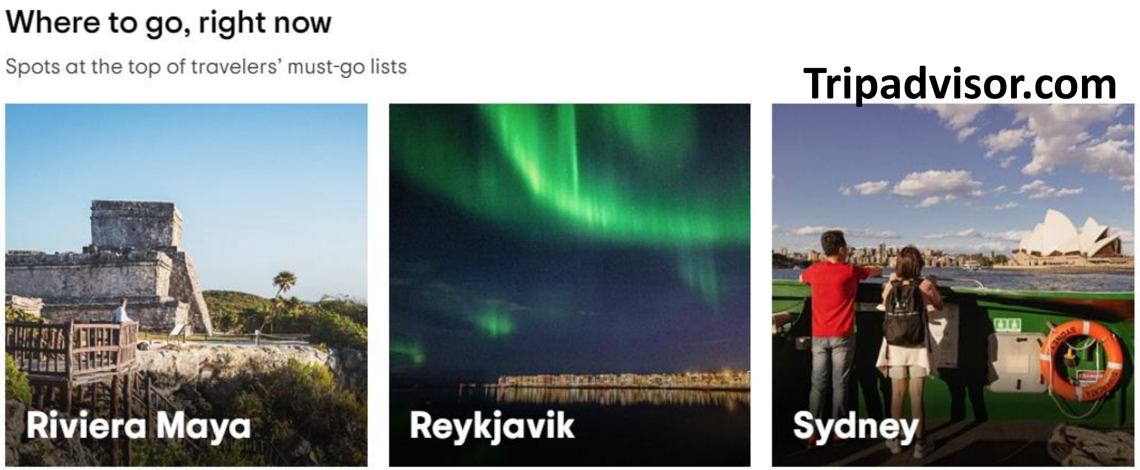
Business R_2 :

Price R_3 :

Kindness R_4 :



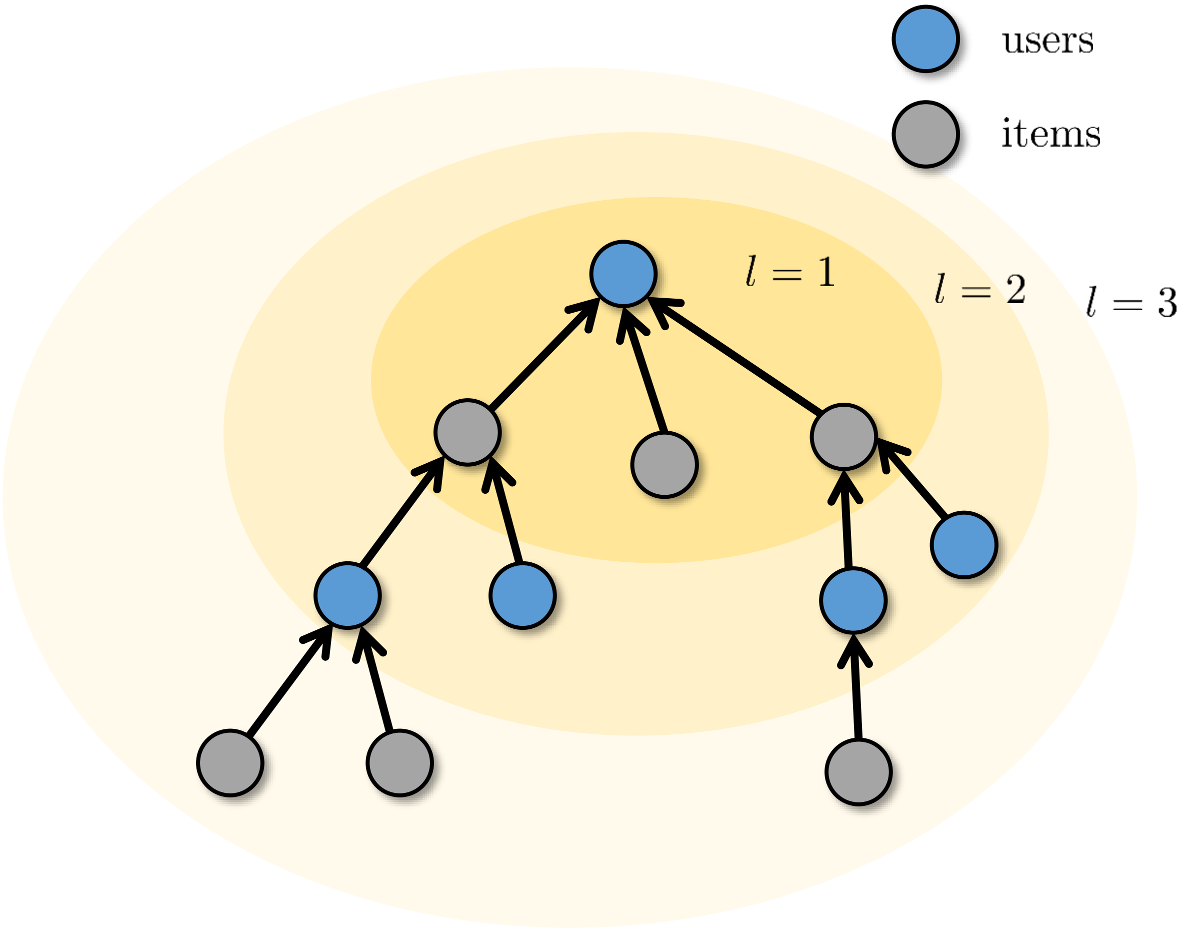
Formal Definition of MCRS



Definition 1: (Top-K MC recommendation) Given $u \in \mathcal{U}$ and $i \in \mathcal{I}$, and $C+1$ user–item ratings $\boxed{\mathcal{R}_0 \times \mathcal{R}_1 \times \ldots \times \mathcal{R}_C}$ including an overall rating $\mathcal{R}_0$, the top-K MC recommendation aims to recommend top-K items that user $u \in \mathcal{U}$ is most likely to prefer among his/her non-interacted items in $\mathcal{I} \setminus \mathcal{N}_u$ w.r.t. the *overall* rating by using all C+1 user-item MC ratings.

Recommend top-k relevant unseen items based on **overall** ratings

Motivation 1: GNN-based RS



GNN becomes the state-of-the-art for collaborative filtering,

because of its capacity to capture **collaborative signals** in **high-order connectivity** in user-item interactions
[He et al., SIGIR 2020]

However, there is **no prior attempt** to solve **MCRS based on GNNs**

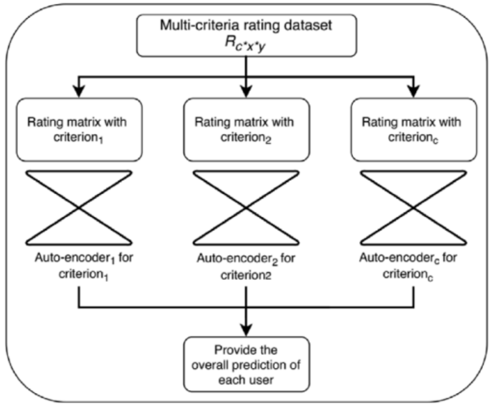
He et al. "Lightgcn: Simplifying and powering graph convolution network for recommendation." SIGIR (2020).

Conventional Approaches

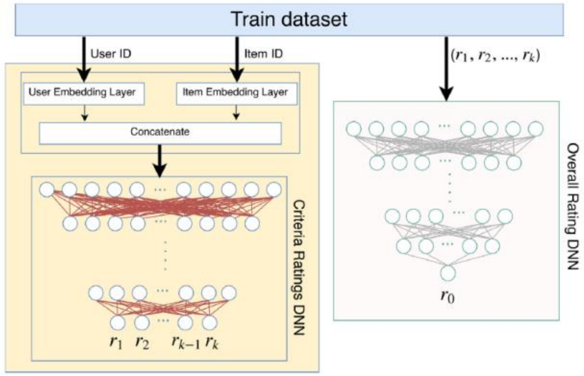
Prior work 1: AEMC [Shambour et al., KBS 2021]

- Reconstruct each rating matrix via **autoencoders**
- Overall prediction is calculated using arithmetic mean

$$r_0 = \frac{r_1 + r_2 .... + r_c}{c}$$



Prior work 2: DMCF [Nassar et al., KBS 2020]



- **Two-stage** DNN model
- Relation between overall rating and multi-criteria ratings is captured via DNN

Limitations of Prior Work

1. **High-order connectivity** of user-item interactions is **not explored**
2. **Information *across* criteria** is **implicitly** captured

Thus, **less** effective!

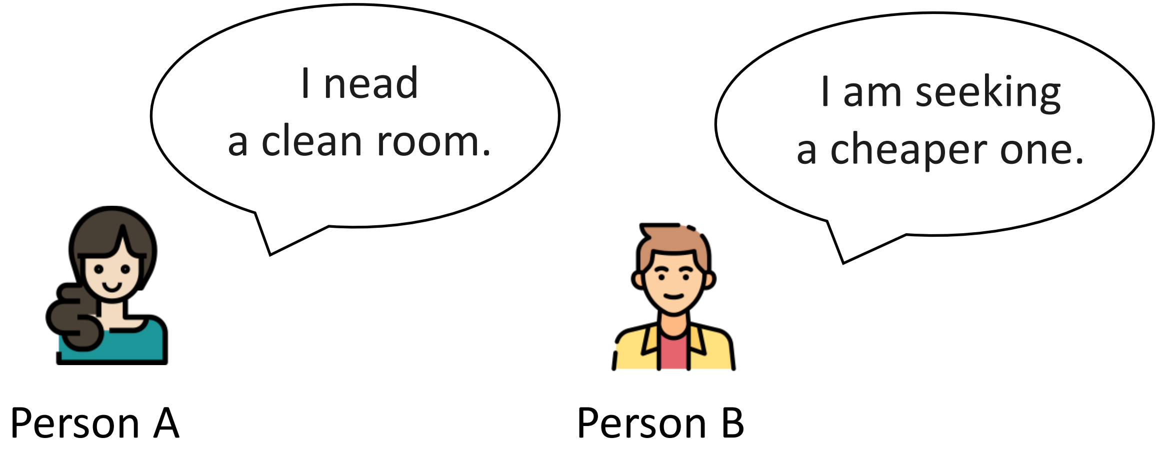
Shambour, Qusai. "A deep learning based algorithm for multi-criteria recommender systems." Knowledge-Based Systems (2021)

Nassar, Nour, Assef Jafar, and Yasser Rahhal. "A novel deep multi-criteria collaborative filtering model for recommendation system." Knowledge-Based Systems (2020)

Motivation 2: Criteria Preference

Criteria preference of users when consuming items

Each user has one's own ***criteria preference (bias)***

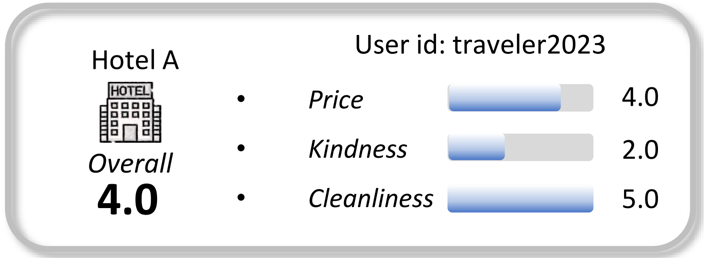


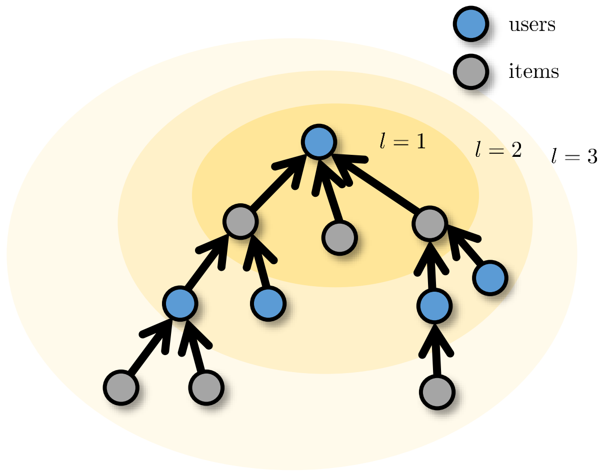
Person A tends to consider *cleanliness* of the hotel,
while person B tends to see *price* of the hotel.

Challenges in Designing MCRS

Challenge 1. Graph construction

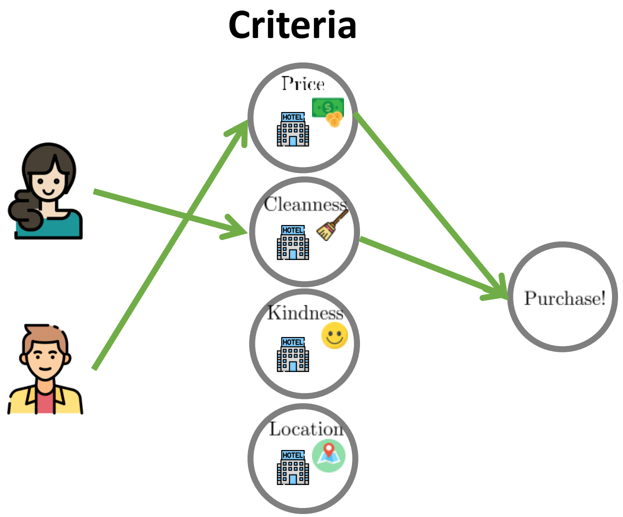
Which **graph type** should be considered to explore ***high-order connectivity patterns in MC ratings***?





Challenge 2. Criteria preference awareness

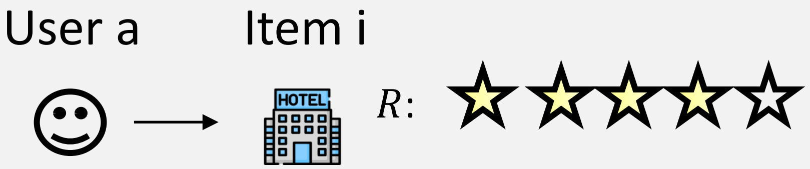
How to maximally grasp the ***criteria preference of users*** through graph convolution?

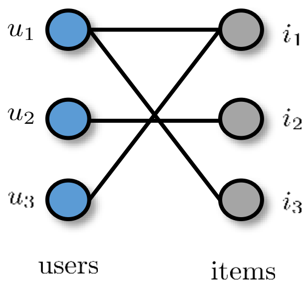


Methodology

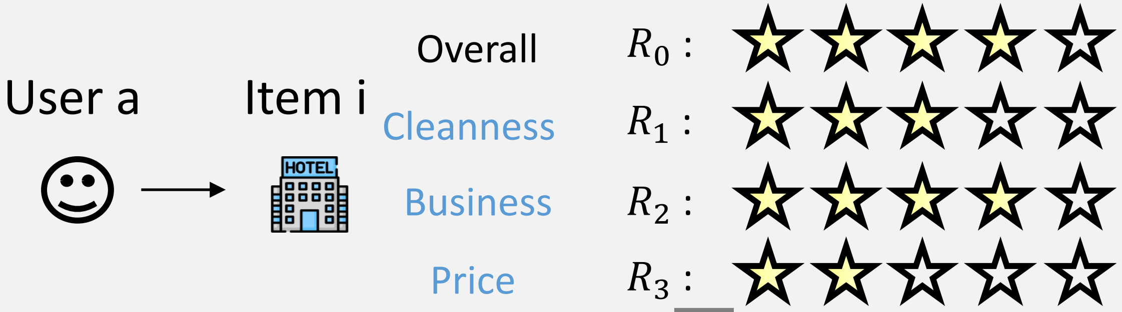
Single-Criterion RS vs MCRS

(Single-Criterion) RS





MCRS



1. How to construct **graph(s)**?

Motivation of Graph Construction

Complex semantics in the MC ratings

User	Item	Overall	Cleanliness	Price
u_1	i_1	4	4	1
u_2	i_1	4	4	5

"u_1 and u_2 both like hotel i_1,
while having the same opinion on the cleanliness aspect,
but revealing opposite opinions on the price aspect"

Motivation of Graph Construction

Complex semantics in the MC ratings

User	Item	Overall	Cleanliness	Price
u_1	i_1	4	4	1
u_2	i_1	4	4	5

How to construct a graph that let the GNNs capture such complex semantics in MC ratings?

“u

while having the

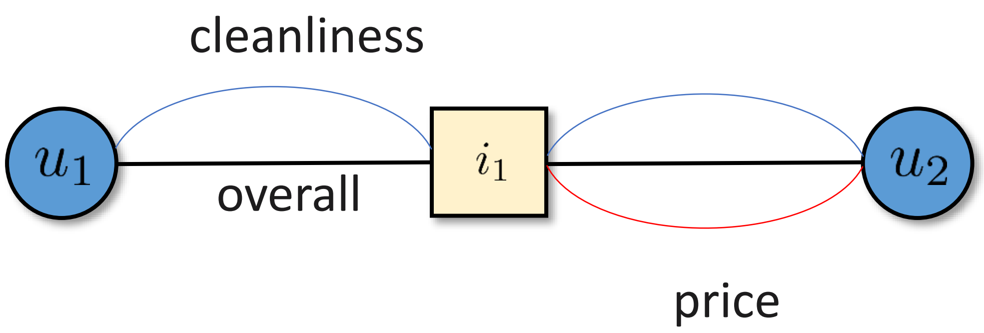
but revealing opposite opinions on the price aspect”

Possible Naïve Approaches

Option 1: Multi-graph construction

User	Item	Overall	Cleanliness	Price
u_1	i_1	4	4	1
u_2	i_1	4	4	5

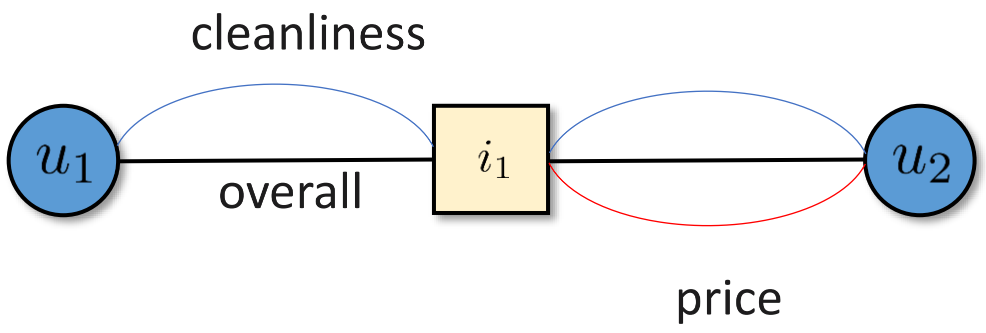
A rating instance



Multi-graph construction

Possible Naïve Approaches

Option 1: Multi-graph construction



Multi-graph

Limitations

1) Expensive computation costs

2) Handcrafted meta-paths

[Guo et al., TKDE 2020, Lv et al., KDD 2021]

Not desirable for scalable MCRS

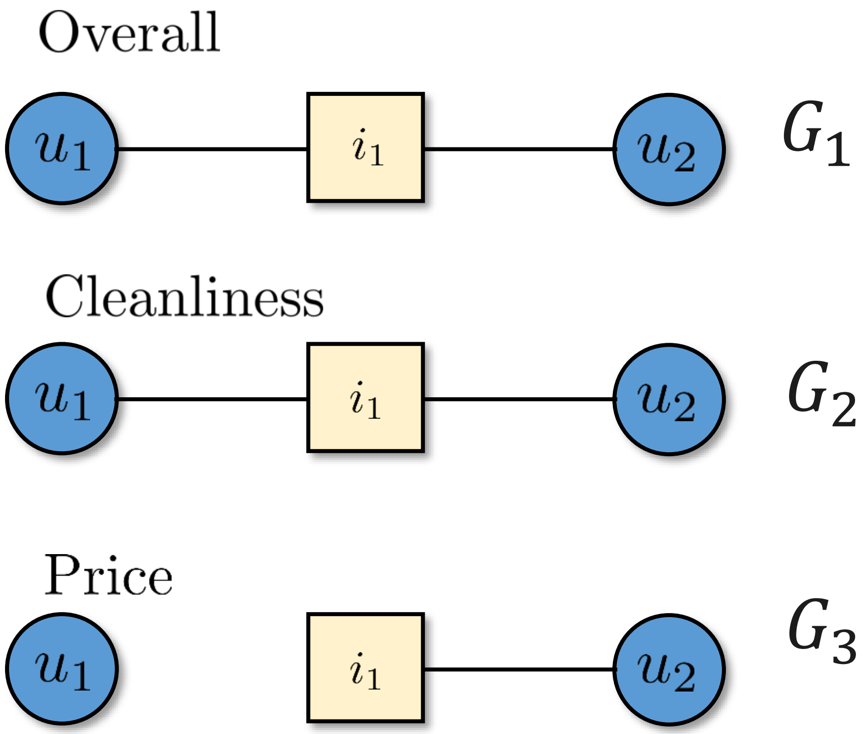
Guo et al, A survey on knowledge graph-based recommender systems, TKDE (2020)
Lv et al, Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks, KDD (2021)

Possible Naïve Approaches

Option 2: Separate Graph Construction for Each Criterion

		G_1	G_2	G_3
User	Item	Overall	Cleanliness	Price
u_1	i_1	4	4	1
u_2	i_1	4	4	5

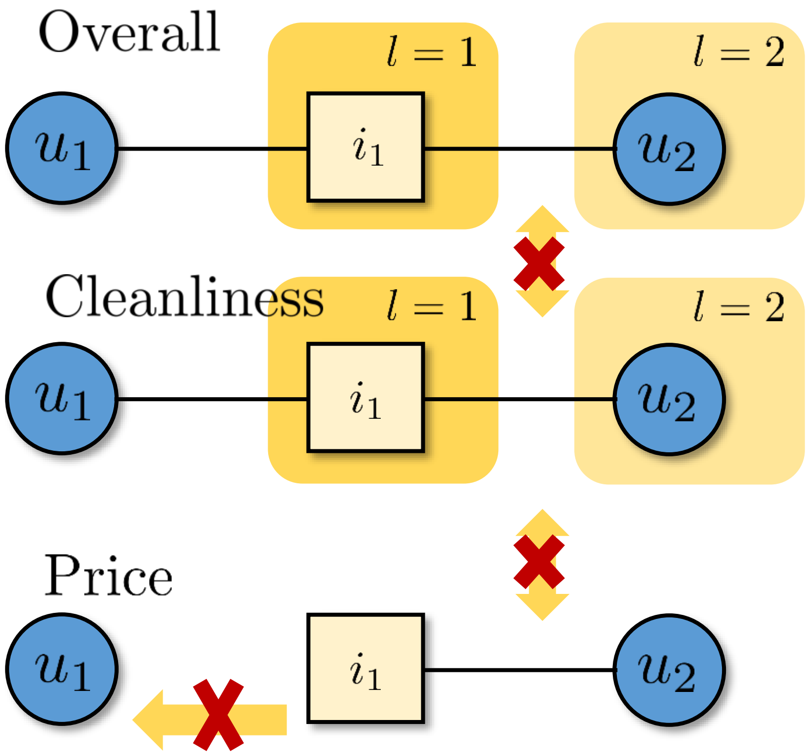
Rating instance



Graph construction

Possible Naïve Approaches

Option 2: Separate Graph Construction for Each Criterion



Root node of the GNN: u_1

Limitations

1) Complex semantics **across** MC ratings cannot be captured

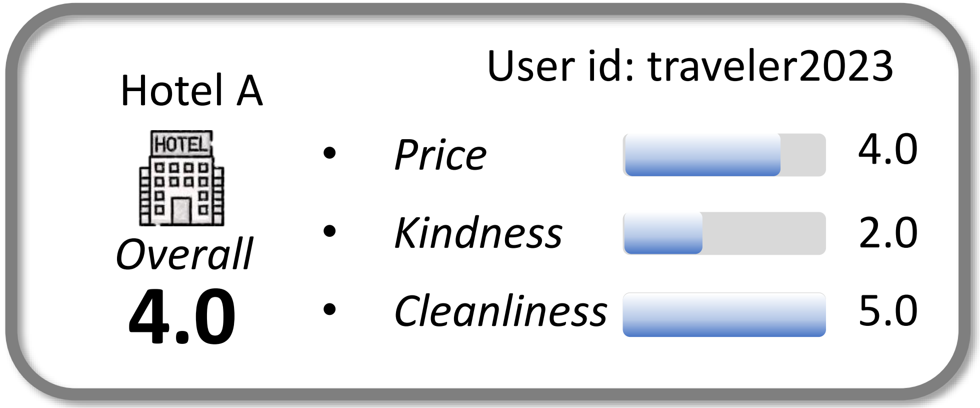
2) Needs a large-size model (w/ many parameters) to deal with multiple graphs

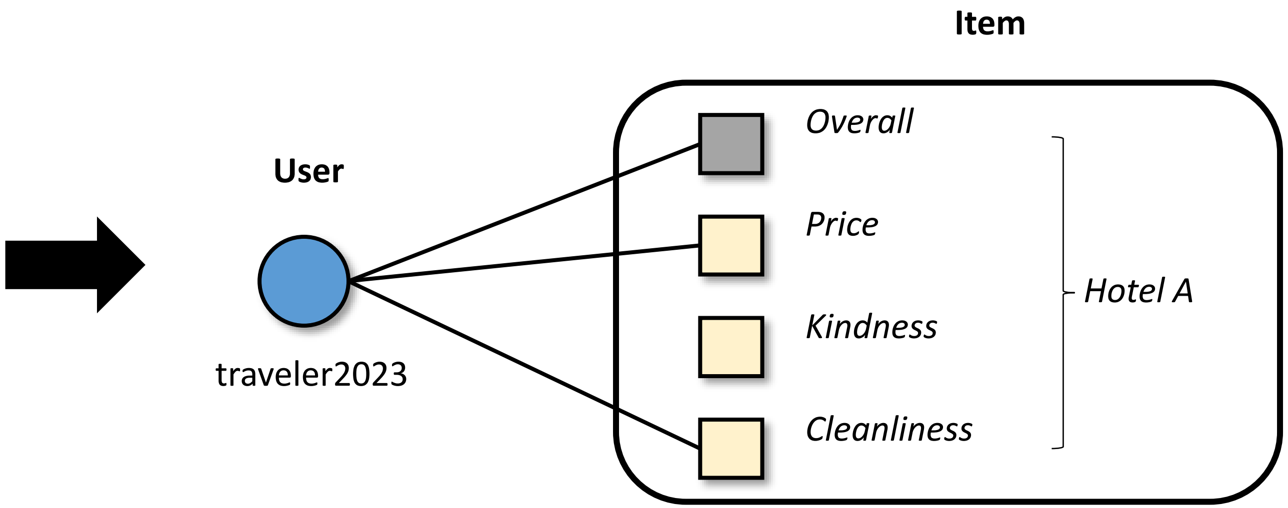
Proposition: Graph Construction in MCRS

MC Expansion Graph

Expand each “item” to (C+1)-criterion-item nodes

C multi-criteria + 1 overall ratings





MC expansion graph

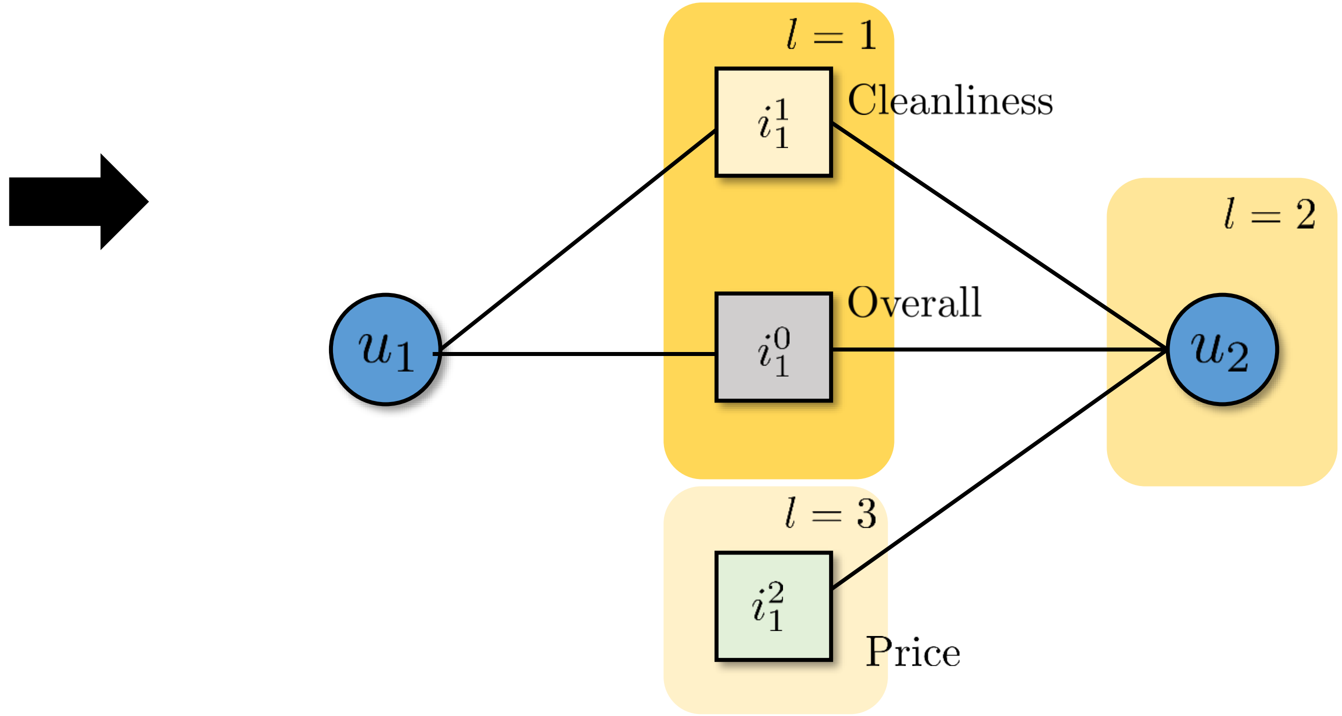
Proposition: Graph Construction in MCRS

MC Expansion Graph

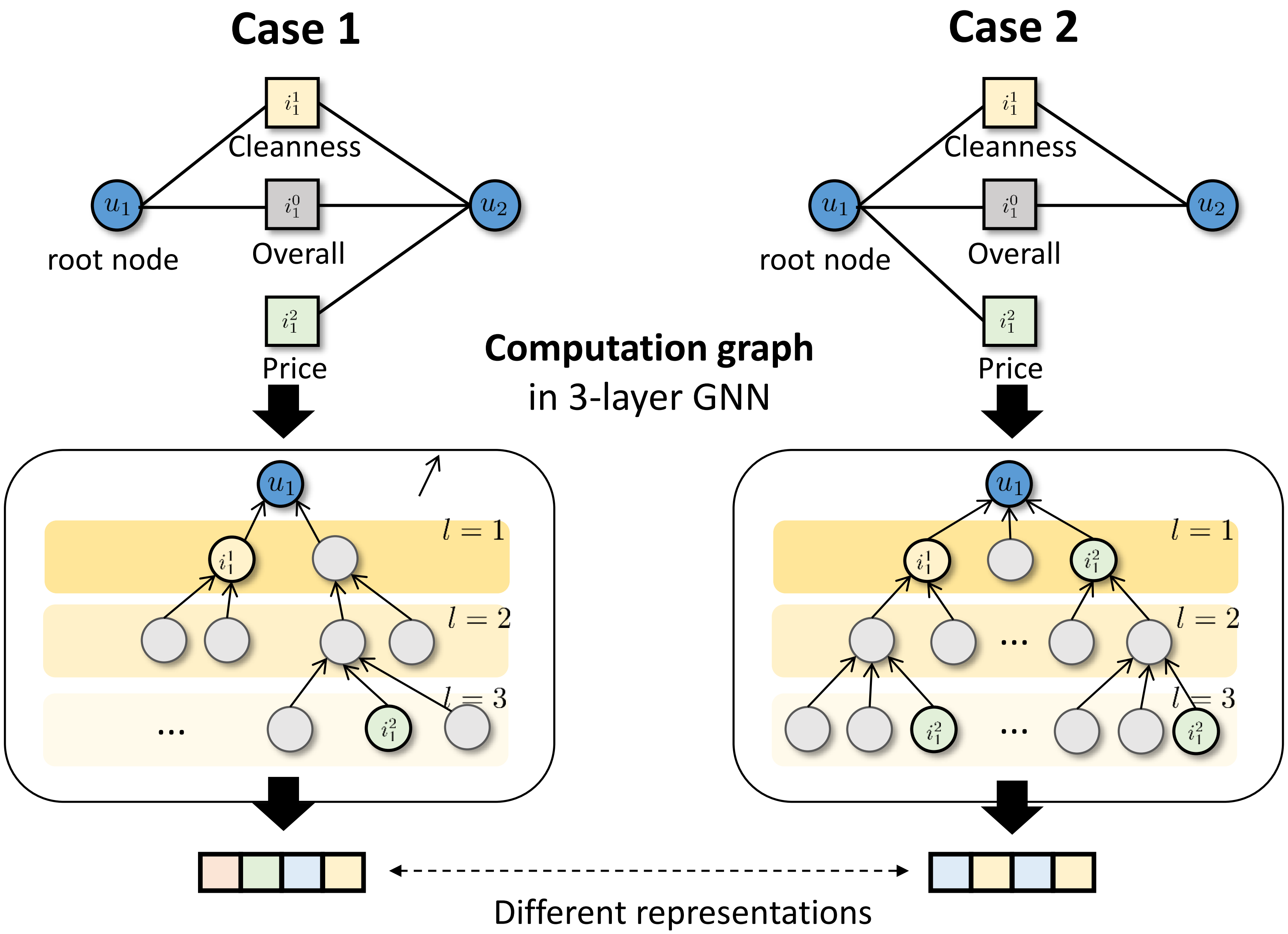
Our MC expansion graph let the multi-layer GNN to capture complex semantics in the MC ratings!

User	Item	Overall	Cleanliness	Price
u_1	i_1	4	4	1
u_2	i_1	4	4	5

A rating instance



Capability of the MC Expansion Graph



Case 1:

User	Item	Overall	Cleanliness	Price
u_1	i_1	4	4	1
u_2	i_1	4	4	5

*"Price of item 1" appears first at the **3rd layer***

Distinguishable!

Case 2:

User	Item	Overall	Cleanliness	Price
u_1	i_1	4	4	5
u_2	i_1	4	4	1

*"Price of item 1" appears first at the **1st layer***

Single-Criterion RS vs MCRS

(Single-Criterion) RS

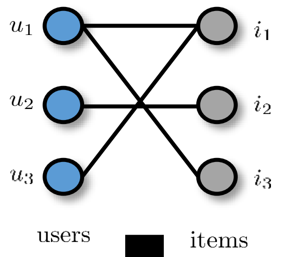
User a → Item i

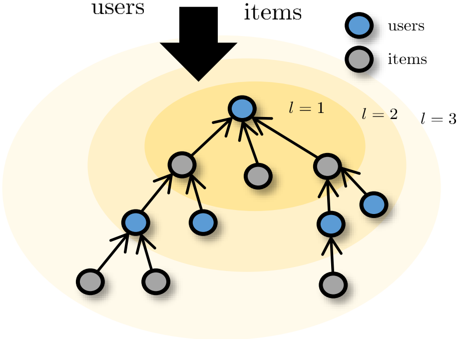
R: ★★★★★

MCRS

User a → Item i

Overall	R_0 :	★★★★★
Cleanness	R_1 :	★★★★★
Business	R_2 :	★★★★★
Price	R_3 :	★★★★★

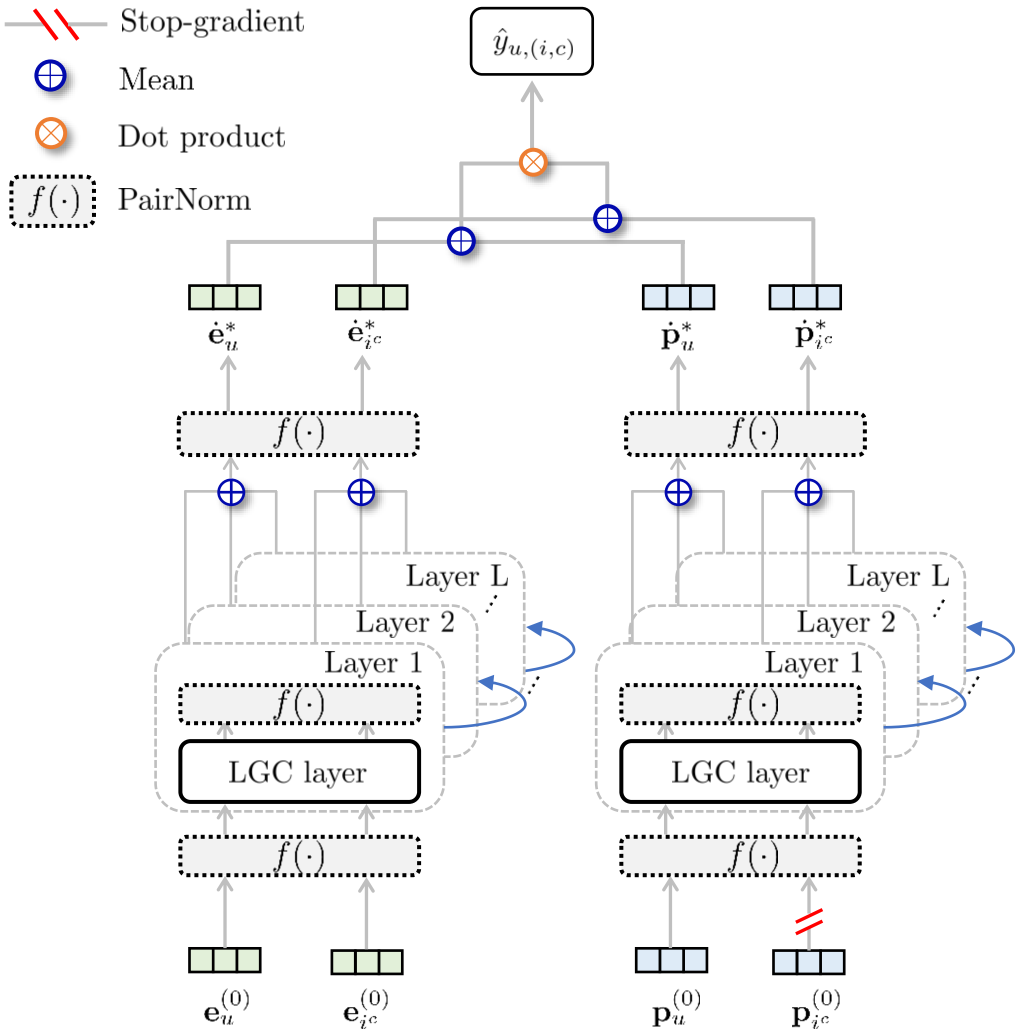




1. How to construct a **graph**?

2. How to design an effective **GNN architecture for MCRS**?

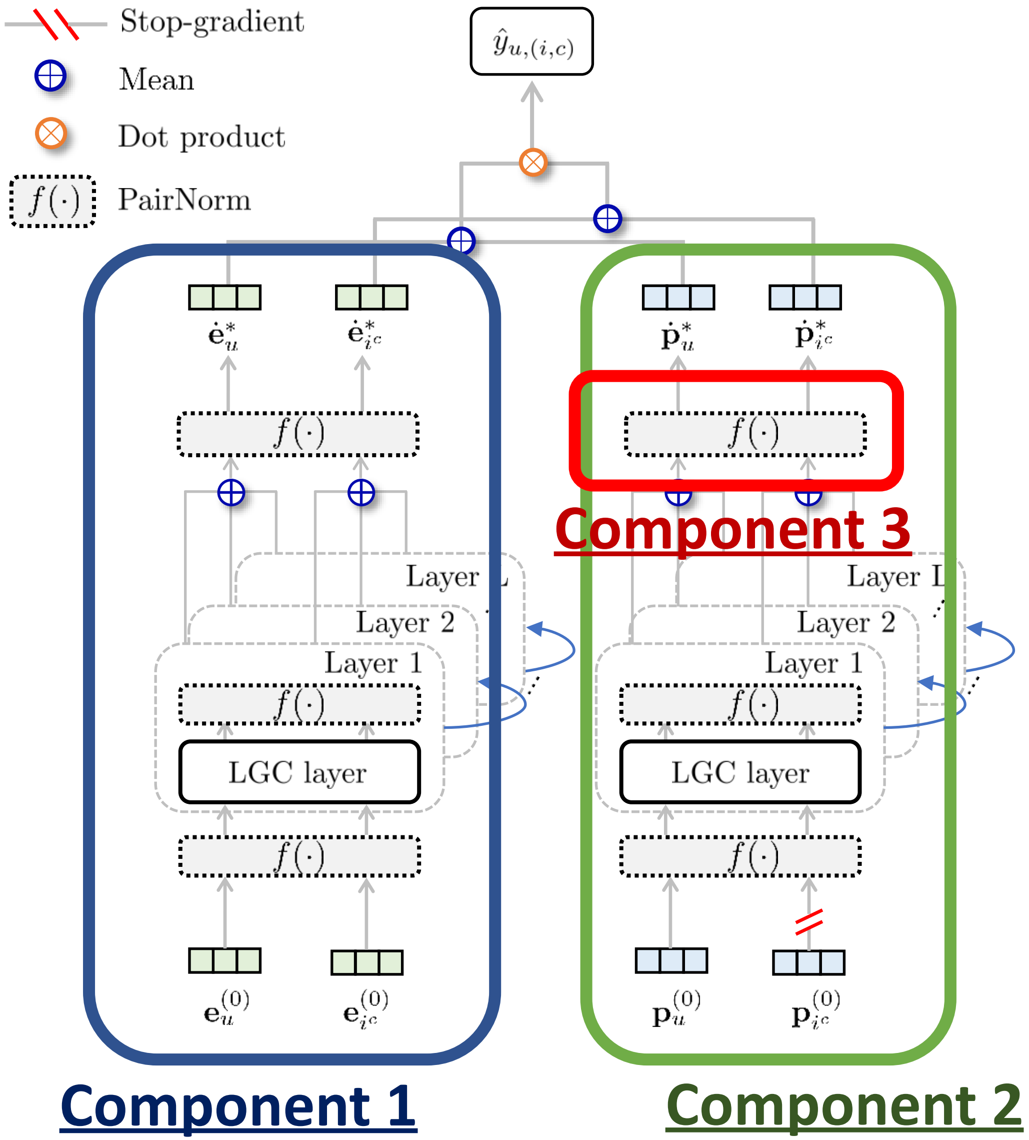
Overview of CPA-LGC



CPA-LGC

Criteria Preference-Aware Light Graph Convolution (CPA-LGC)

Overview of CPA-LGC



Criteria Preference-Aware Light Graph Convolution (CPA-LGC)

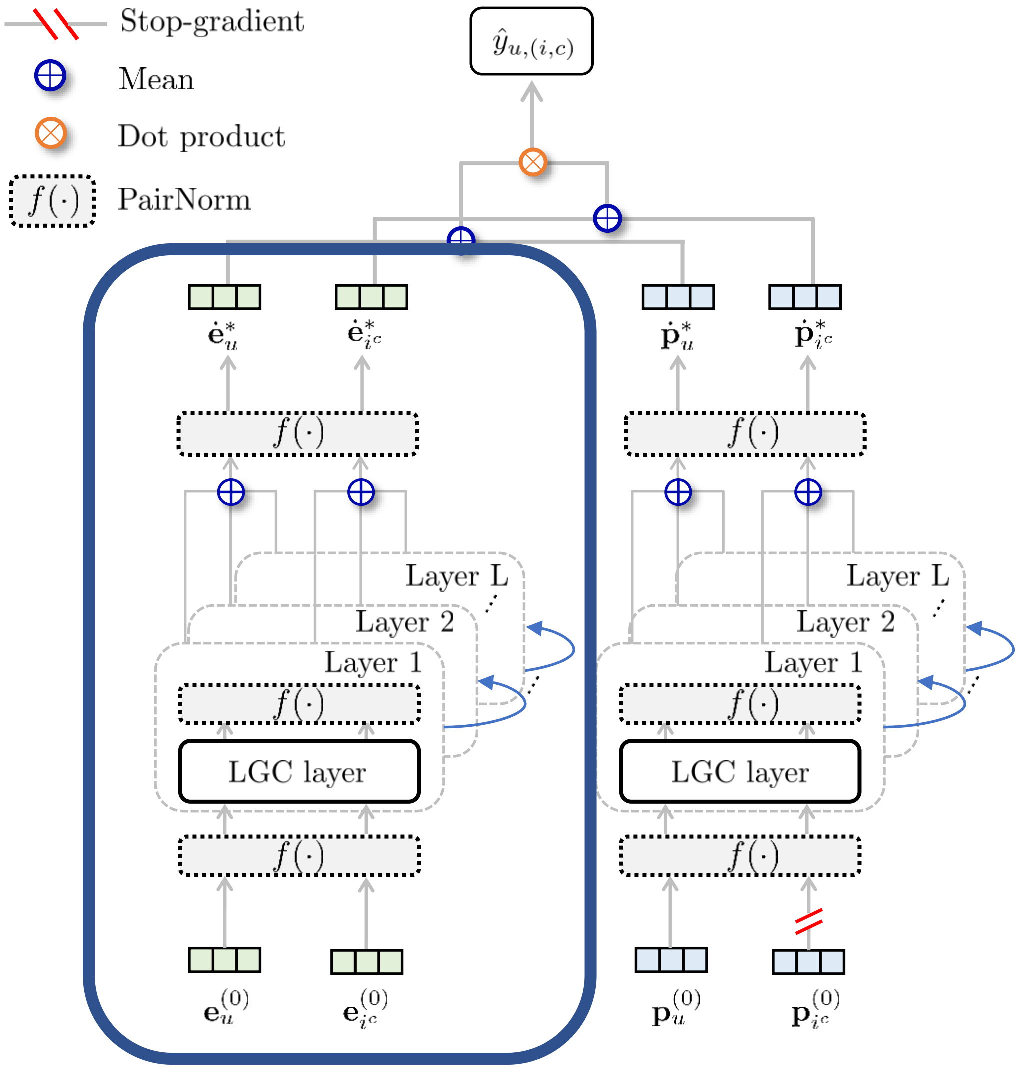
Three key components

Component 1: Light graph convolution (LGC) on user/criterion-item embeddings

Component 2: LGC on user-specific criteria-preference (UCP) / item-specific criterion (IC) embeddings

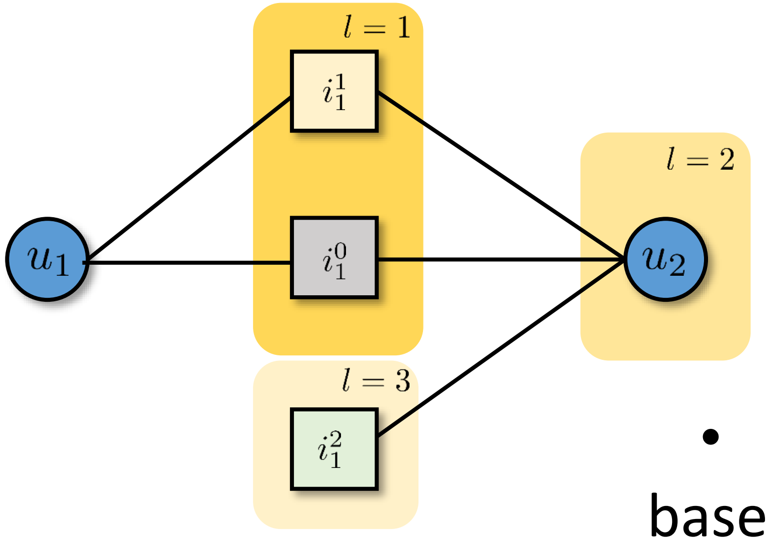
Component 3: Over-smoothing alleviation

Implementation Details: Component 1



CPA-LGC

LGC on User/Criterion-Item Embeddings

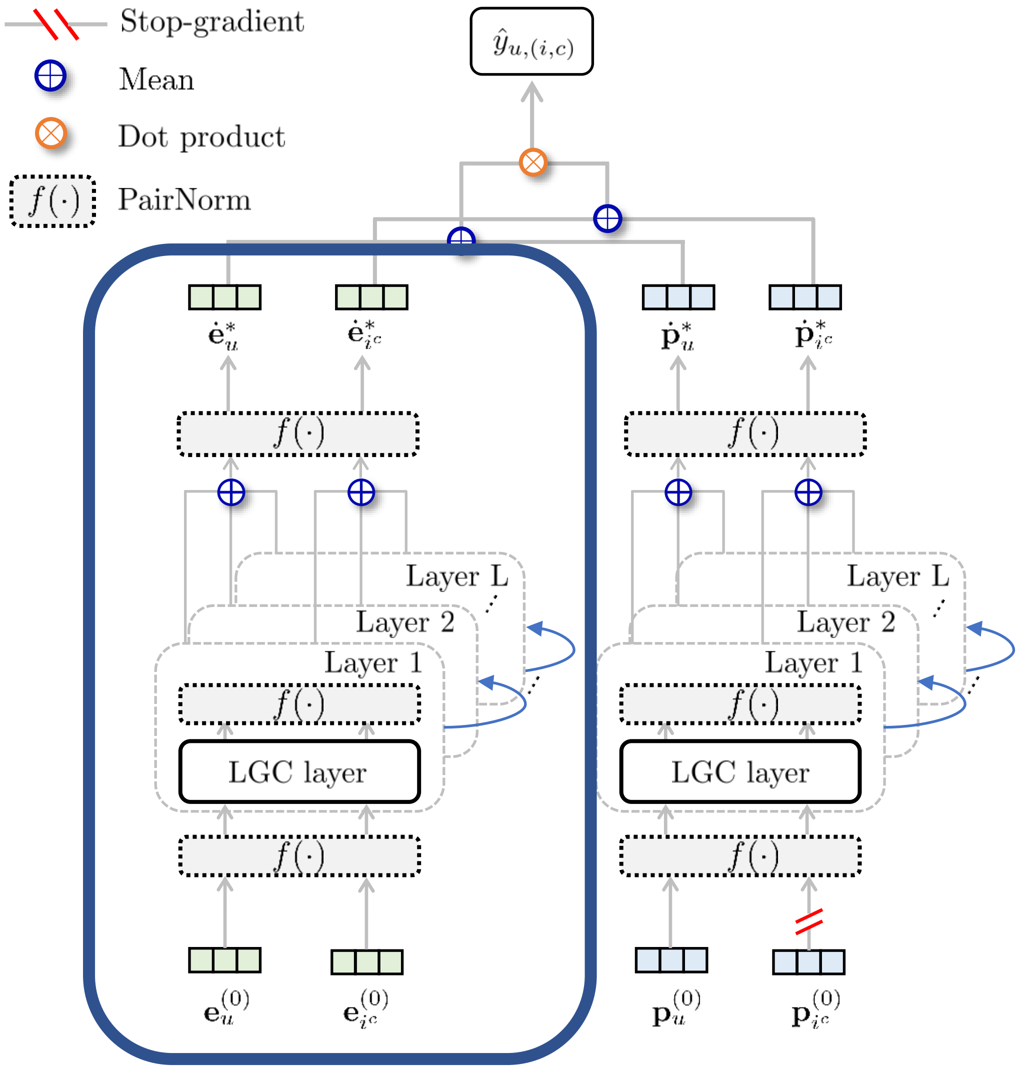


- Graph convolution based on the structure of the MC expansion graph

$$\mathbf{e}_u^{(l)} = \sum_{\boxed{i^c \in \mathcal{N}_u}} \frac{w_{u,i^c}}{\sqrt{\sum_{i^c \in \mathcal{N}_u} w_{u,i^c}}\sqrt{\sum_{v \in \mathcal{N}_{i^c}} w_{v,i^c}}} \dot{\mathbf{e}}_{i^c}^{(l-1)}$$

$$\mathbf{e}_{i^c}^{(l)} = \sum_{\boxed{u \in \mathcal{N}_{i^c}}} \frac{w_{u,i^c}}{\sqrt{\sum_{u \in \mathcal{N}_{i^c}} w_{u,i^c}}\sqrt{\sum_{j^r \in \mathcal{N}_u} w_{u,j^r}}} \dot{\mathbf{e}}_u^{(l-1)}$$

Implementation Details: Component 1



CPA-LGC

LGC on User/Criterion-Item Embeddings

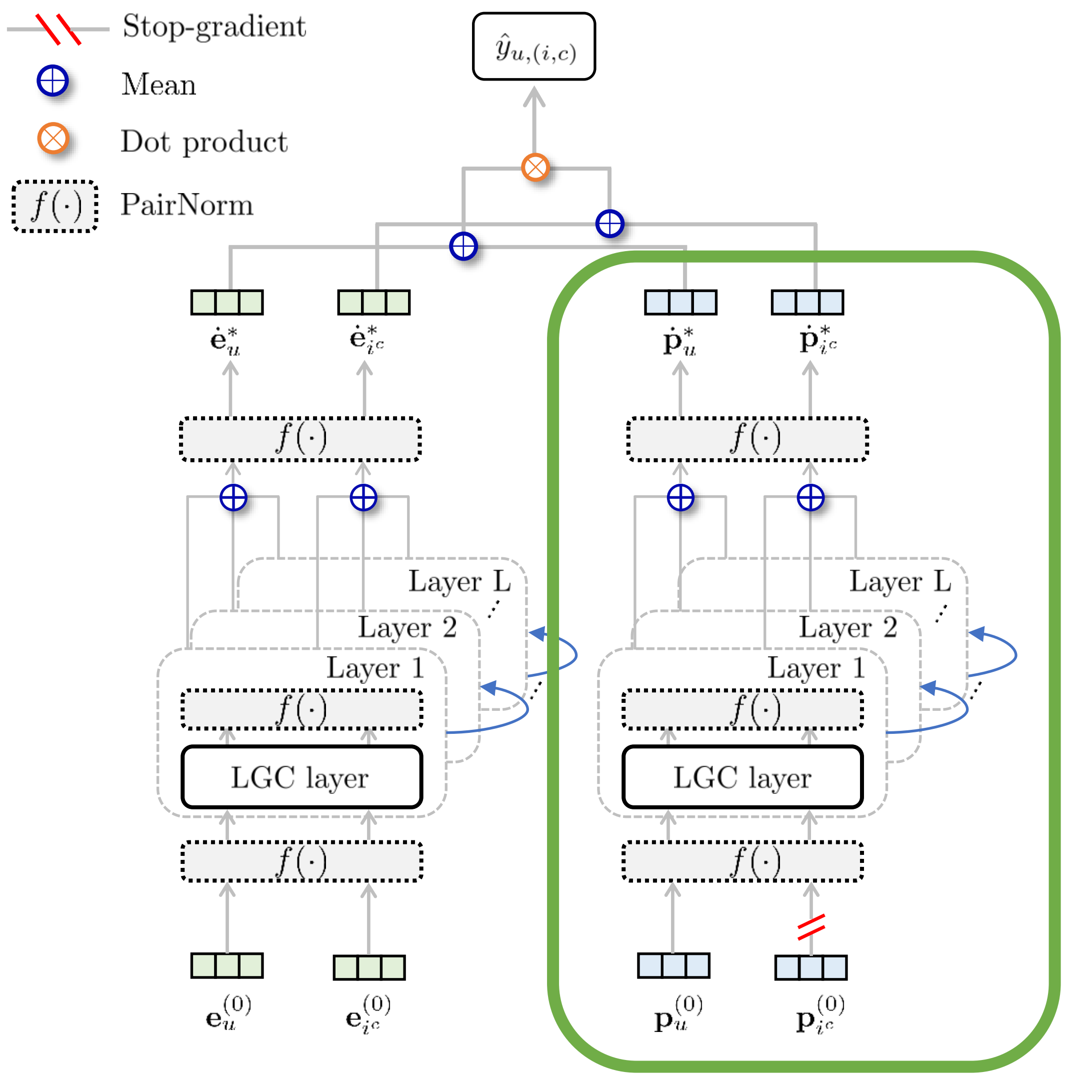
- Light-weight architecture: Neither feature transformation nor non-linearity

$$\mathbf{e}_u^{(l)} = \sum_{i^c \in \mathcal{N}_u} \frac{w_{u,i^c}}{\sqrt{\sum_{i^c \in \mathcal{N}_u} w_{u,i^c}} \sqrt{\sum_{v \in \mathcal{N}_{i^c}} w_{v,i^c}}} \dot{\mathbf{e}}_{i^c}^{(l-1)}$$

$$\mathbf{e}_{i^c}^{(l)} = \sum_{u \in \mathcal{N}_{i^c}} \frac{w_{u,i^c}}{\sqrt{\sum_{u \in \mathcal{N}_{i^c}} w_{u,i^c}} \sqrt{\sum_{j^r \in \mathcal{N}_u} w_{u,j^r}}} \dot{\mathbf{e}}_u^{(l-1)}$$

- Weighted propagation: Importance of information from each criterion may differ

Implementation Details: Component 2

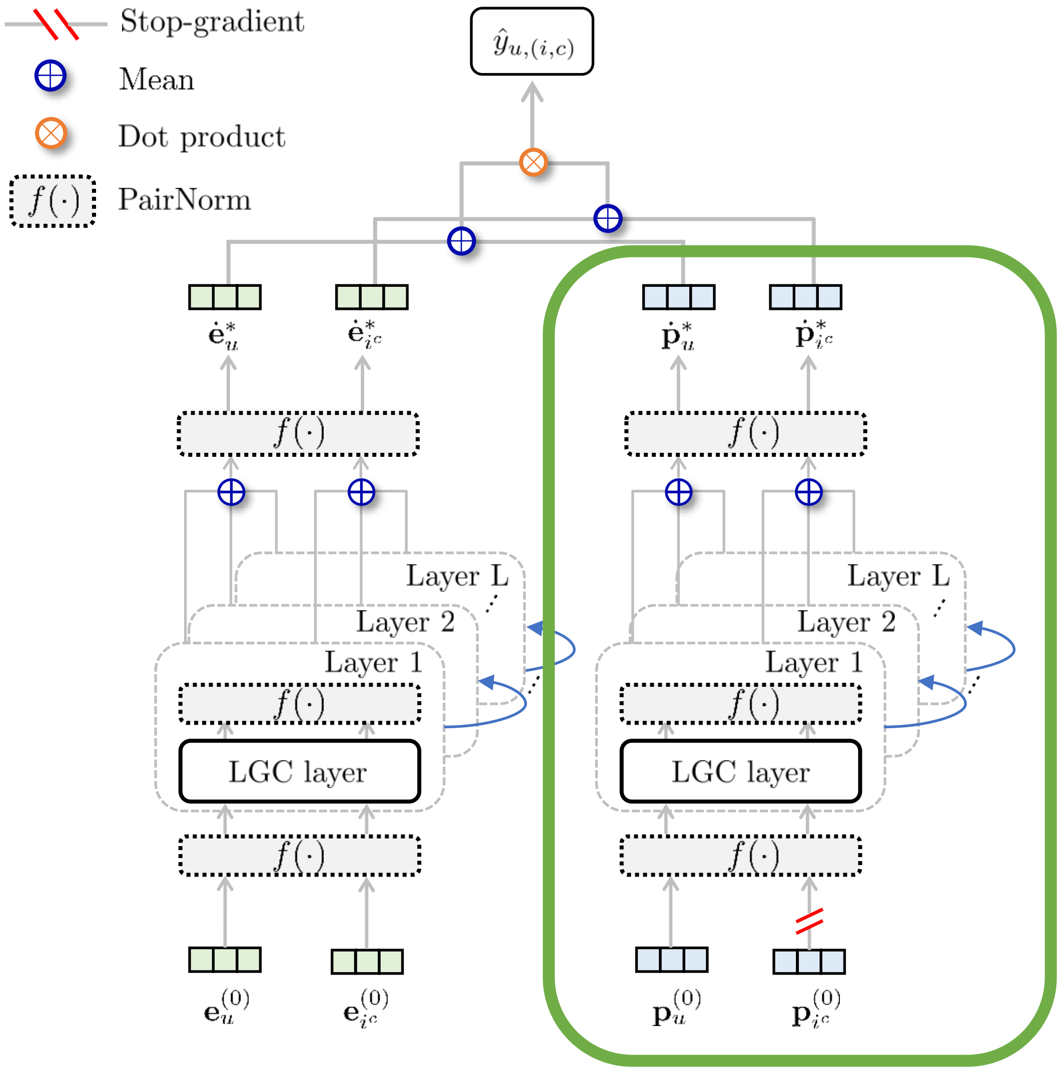


CPA-LGC

LGC on UCP/IC Embeddings

- Newly characterized embeddings
  - ✓ **User-specific criteria-preference (UCP)** embedding $\mathbf{p}_u$
  - ✓ **Item-specific criteria (IC)** embedding $\mathbf{p}_{i^c}$
    - <u>Initialized specific to the criterion</u> of given criterion-item node

Implementation Details: Component 2



CPA-LGC

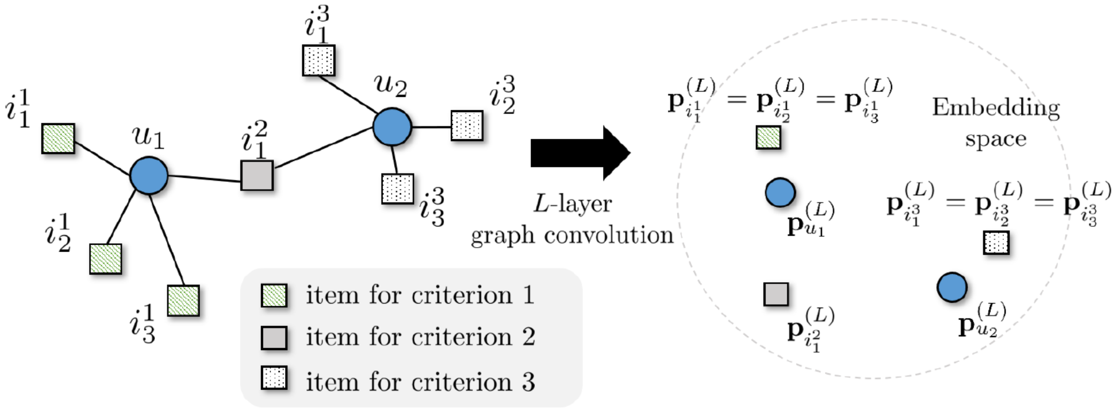
LGC on UCP/IC Embeddings

- Graph convolution

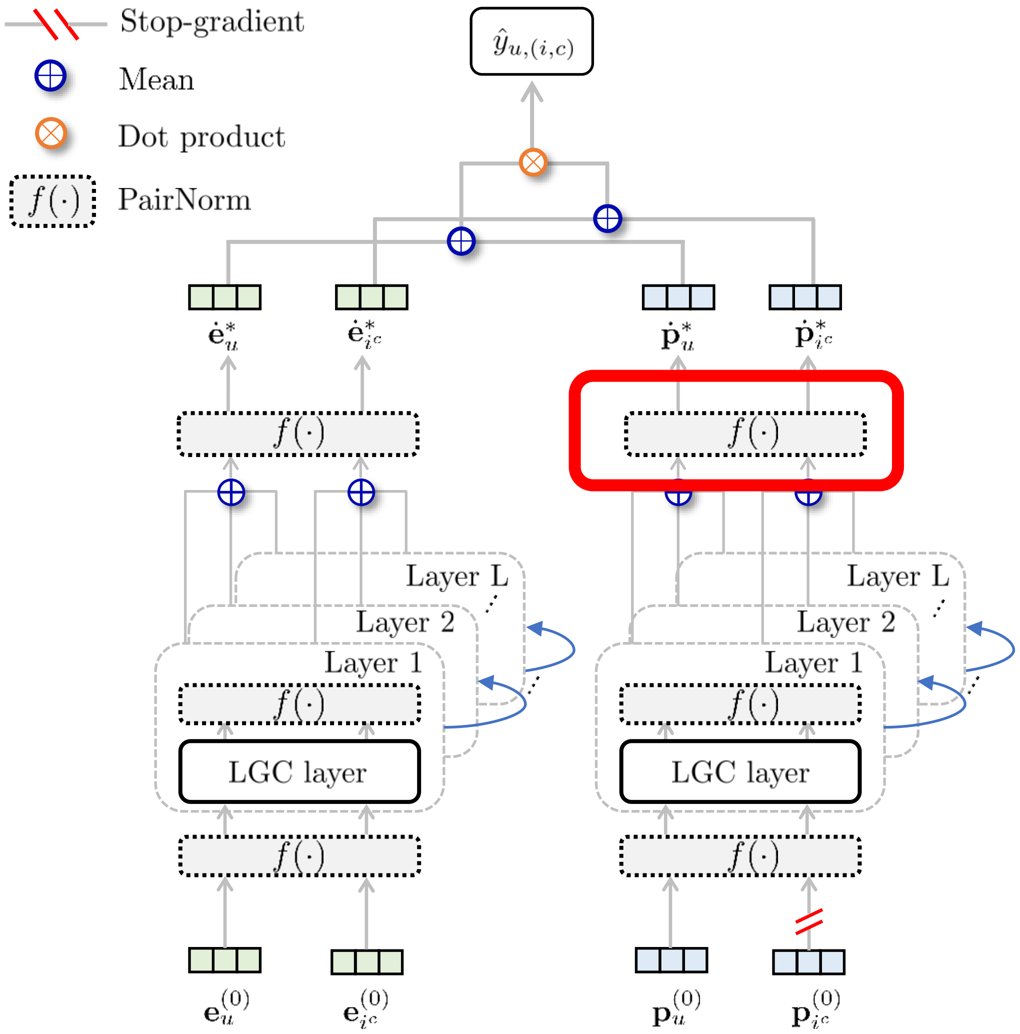
$$\mathbf{p}_u^{(l)} = \sum_{i^c \in \mathcal{N}_u} \frac{w_{u,i^c}}{\sqrt{\sum_{i^c \in \mathcal{N}_u} w_{u,i^c}}\sqrt{\sum_{v \in \mathcal{N}_{i^c}} w_{v,i^c}}} \dot{\mathbf{p}}_{i^c}^{(l-1)}$$

$$\mathbf{p}_{i^c}^{(l)} = \sum_{u \in \mathcal{N}_{i^c}} \frac{w_{u,i^c}}{\sqrt{\sum_{u \in \mathcal{N}_{i^c}} w_{u,i^c}}\sqrt{\sum_{j^r \in \mathcal{N}_u} w_{u,j^r}}} \dot{\mathbf{p}}_u^{(l-1)},$$

- Stop-gradient on item criterion embedding



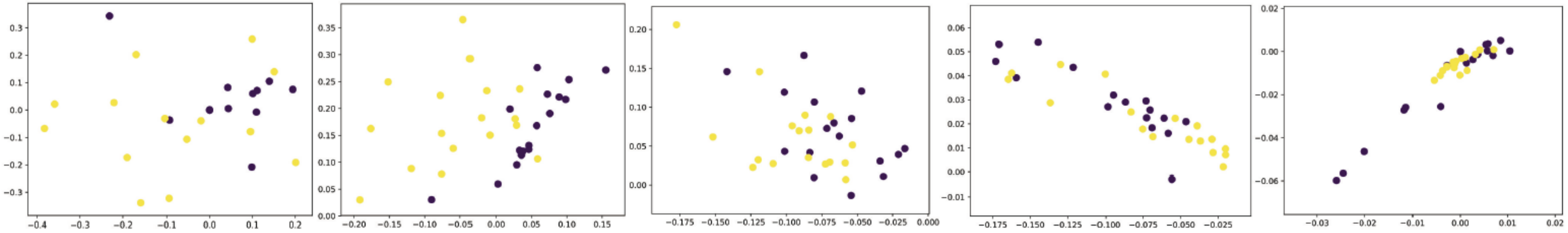
Implementation Details: Component 3



CPA-LGC

Preliminary: Over-Smoothing in GNNs

[Chen et al., ICML 2020]



(a) 1-layer (b) 2-layer (c) 3-layer (d) 4-layer (e) 5-layer

Figure 2: Vertex embeddings of Zachary's karate club network with GCNs with 1,2,3,4,5 layers.

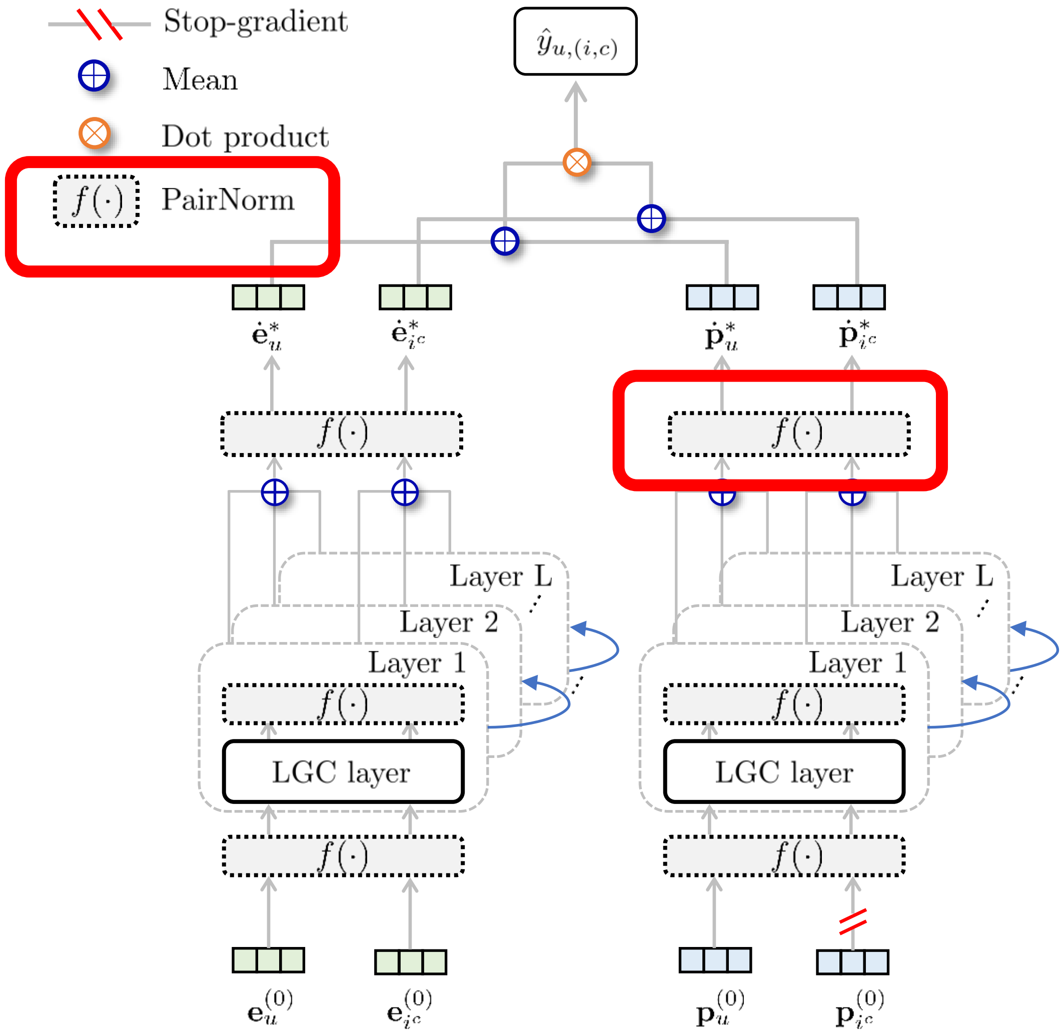
- **Stacking many layers** in GNN may suffer over-smoothing, where node representations become similar [Li et al., AAAI 2018]
- Nodes with high-degree is more vulnerable
- Convergence rate of a node is faster for high-degree nodes

*** Node representation after K-layer GCN**

$$\left(\sum_{i=1}^{n} \frac{\sqrt{d_i+1}}{2m+n} x_i \pm \frac{\sum_{i=1}^{n} x_i \left(1-\frac{\lambda_{\tilde{G}}^2}{2}\right)^K}{\sqrt{d_j+1}}\right)$$

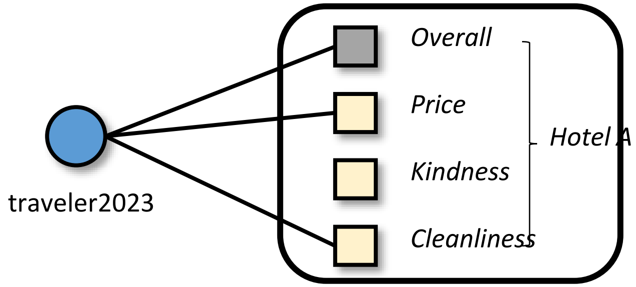
Chen et al. "Simple and deep graph convolutional networks." International conference on machine learning. PMLR (2020).
Li et al. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. AAAI (2018)

Implementation Details: Component 3



CPA-LGC

Layer-Wise Over-Smoothing Alleviation

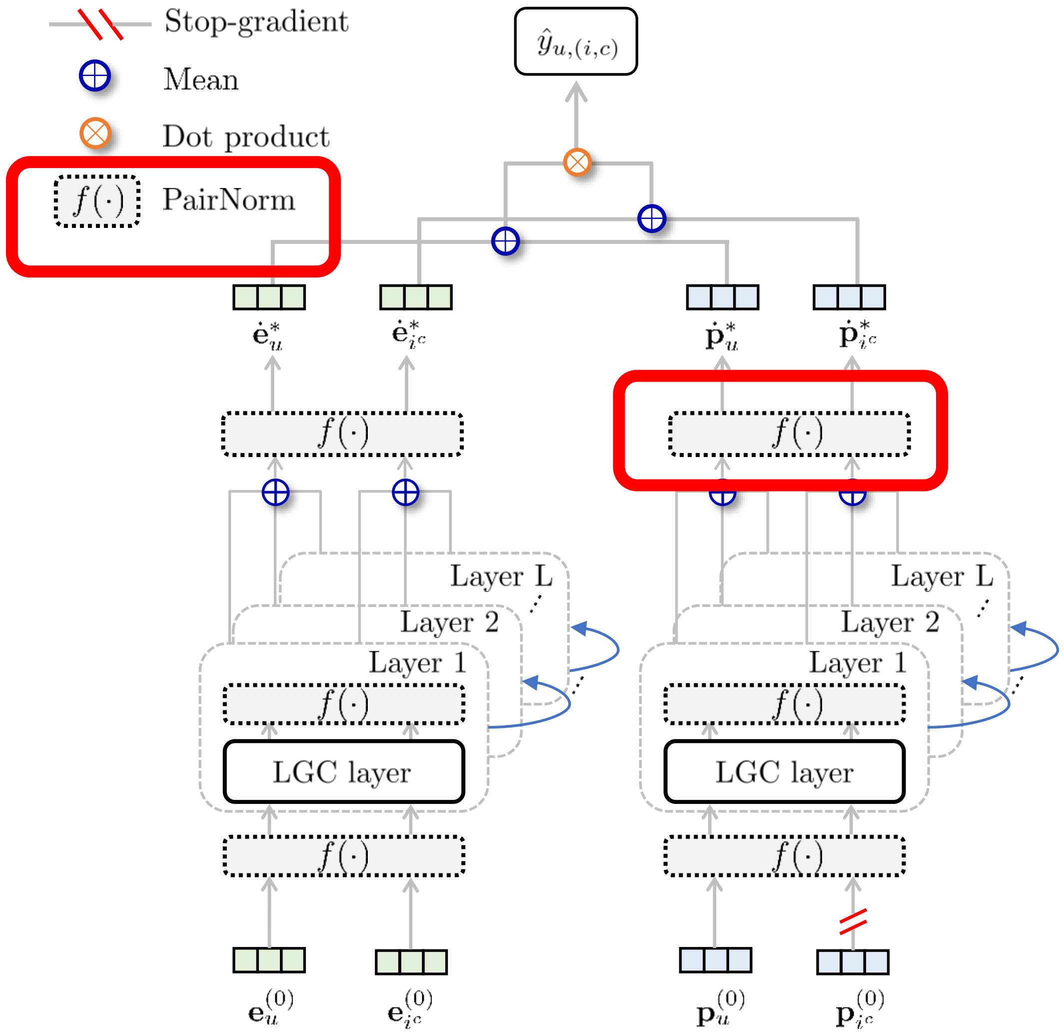


- User nodes in the **MC expansion graph** : **Largely connected (w/ high-degree)** by graph expansion

⬇

Potentially, more vulnerable to **over-smoothing**

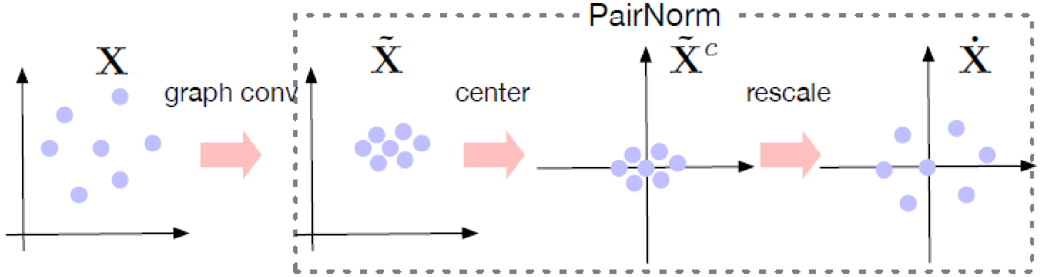
Implementation Details: Component 3



CPA-LGC

Layer-Wise Over-Smoothing Alleviation

- Alleviate potential over-smoothing via layer-wise PairNorm [Zhao et al., ICLR 2019]

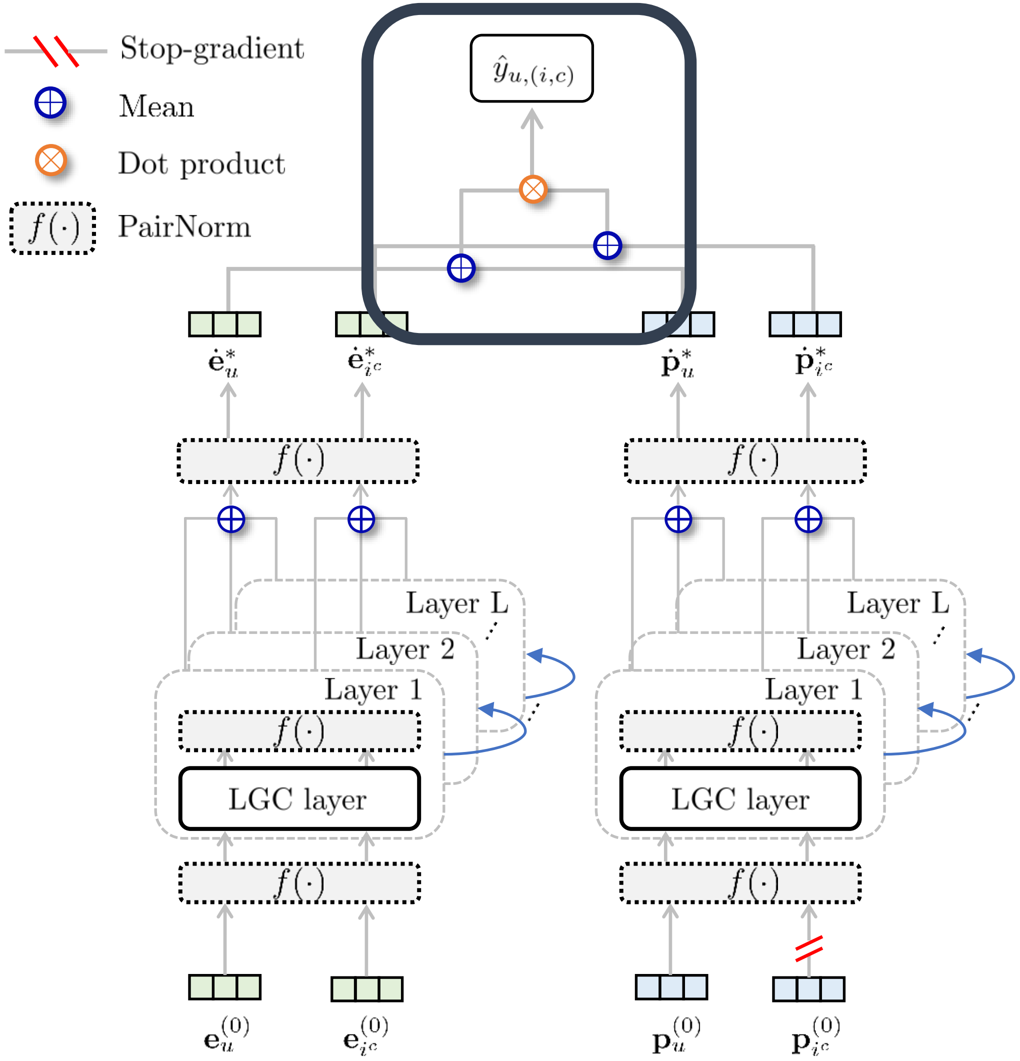


$$\mathbf{m}_v^{(l)} = \mathbf{e}_v^{(l)} - \frac{1}{|\mathcal{V}|}\sum_{i=1}^{|\mathcal{V}|} \mathbf{e}_i^{(l)}$$

$$\dot{\mathbf{e}}_v^{(l)} = s\sqrt{|\mathcal{V}|}\frac{\mathbf{m}_v^{(l)}}{\sqrt{\|\mathbf{E}^{(l)}\|_F^2}},$$

Zhao et al., "Pairnorm: Tackling oversmoothing in gnns." ICLR (2019).

Implementation Details: Component 3



CPA-LGC

Prediction

- Layer-wise combination

$$\mathbf{e}_u^* = \frac{1}{L}\sum_{l=0}^{L} \dot{\mathbf{e}}_u^{(l)}; \mathbf{e}_{i^c}^* = \frac{1}{L}\sum_{l=0}^{L} \dot{\mathbf{e}}_{i^c}^{(l)}; \mathbf{p}_u^* = \frac{1}{L}\sum_{l=0}^{L} \dot{\mathbf{p}}_u^{(l)}; \mathbf{p}_{i^c}^* = \frac{1}{L}\sum_{l=0}^{L} \dot{\mathbf{p}}_{i^c}^{(l)},$$

- If a user-item pair has similar representation, it has high score

$$\hat{y}_{u,i^c} = (\dot{\mathbf{e}}_u^* + \dot{\mathbf{p}}_u^*) \cdot (\dot{\mathbf{e}}_{i^c}^* + \dot{\mathbf{p}}_c^*)^\top$$

Optimization

- BPR Loss [Rendle et al., UAI 2009]

$$-\sum_{u=1}^{|\mathcal{U}|} \sum_{i^c \in \mathcal{N}_u} \sum_{j^r \notin \mathcal{N}_u} \ln \sigma\left(\hat{y}_{u,i^c} - \hat{y}_{u,j^r}\right) + \lambda \|\Theta\|_2^2,$$

Rendle et al., "BPR: Bayesian personalized ranking from implicit feedback". UAI (2009).

Evaluation

Experimental Settings

Dataset description

Dataset	# of users	# of items	# of overall ratings	# of MC ratings	C	γ
TA	4,265	6,275	34,383	202,859	7	5.9
YM	1,821	1,472	46,176	175,468	4	3.8
RB	4,017	3,422	159,755	607,067	4	3.8
YP	58,971	19,820	445,724	1,408,487	3	3.1

- **5-core settings**
  user nodes whose # of ratings lower than 5 are dropped

- **Edge construction**
  ratings more than median

Performance metric

- **Three benchmark metrics for RS:**
  Precision@k, Recall@k, nDCG@k

* Note that the higher the value of each of the three metrics, the better the performance

RQ1. Comparison with MCRS Benchmarks

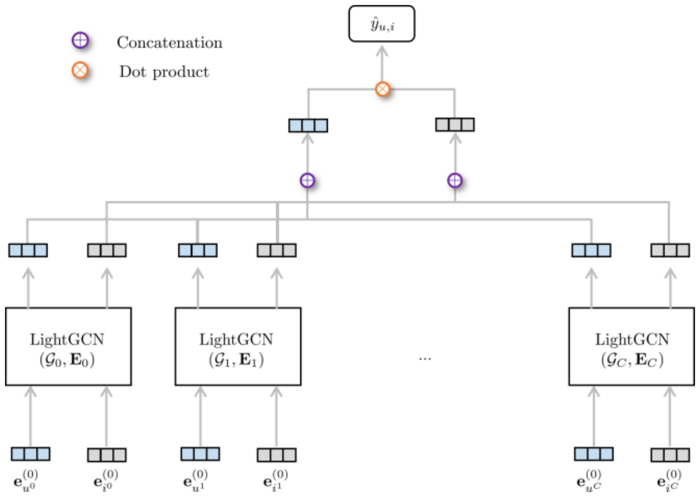
Method	Metric	TA K = 5	TA K = 10	YM K = 5	YM K = 10	RB K = 5	RB K = 10	YP K = 5	YP K = 10
ExtendedSAE	*Precision@K*	0.0012	0.0011	0.0675	0.0480	0.0210	0.0273	OOM	OOM
	Recall@K	0.0031	0.0092	0.0694	0.1000	0.0144	0.0374	OOM	OOM
	NDCG@K	0.0012	0.0043	0.1072	0.1154	0.0285	0.0435	OOM	OOM
UBM	*Precision@K*	0.0158	0.0122	0.0160	0.0223	0.0250	0.0288	0.0137	0.0125
	Recall@K	0.0443	0.0533	0.0264	0.0294	0.0174	0.0355	0.0386	0.0713
	NDCG@K	0.0351	0.0346	0.0202	0.0245	0.0301	0.0440	0.0248	0.0341
DMCF	*Precision@K*	0.0167	0.0137	0.0334	0.0242	0.0816	0.0721	0.0090	0.0075
	Recall@K	0.0174	0.0232	0.0333	0.0470	0.0493	0.0887	0.0102	0.0248
	NDCG@K	0.0115	0.0243	0.0541	0.0614	0.1104	0.1317	0.0304	0.0408
AEMC	*Precision@K*	0.0156	0.0154	0.0358	0.0257	0.0997	0.0807	0.0093	0.0064
	Recall@K	0.0172	0.0251	0.0398	0.0540	0.0671	0.1090	0.0437	0.0671
	NDCG@K	0.0207	0.0241	0.0595	0.0693	0.1534	0.1780	0.0433	0.0544
CFM	*Precision@K*	0.0220	0.0170	0.0420	0.0375	0.0739	0.0720	0.0180	0.0165
	Recall@K	0.0615	0.0740	0.0420	0.0613	0.1111	0.1997	0.0482	0.0891
	NDCG@K	0.0487	0.0480	0.0392	0.0583	0.1078	0.1391	0.0349	0.0492
CPA-LGC	*Precision@K*	**0.0449**	**0.0273**	**0.1012**	**0.0788**	**0.2177**	**0.1739**	**0.0360**	**0.0276**
	Recall@K	**0.0901**	**0.1053**	**0.1211**	**0.1725**	**0.1863**	**0.2745**	**0.0859**	**0.1286**
	NDCG@K	**0.0830**	**0.0880**	**0.1392**	**0.1532**	**0.2823**	**0.2892**	**0.0713**	**0.0859**
Gain	*Precision@K*	+104.09 %	+60.59 %	+49.93 %	+64.17 %	+136.37 %	+141.20 %	+100.00 %	+67.27 %
	Recall@K	+46.50 %	+42.30 %	+74.50 %	+72.50 %	+67.69 %	+37.46 %	+78.22 %	+44.33 %
	NDCG@K	+70.43 %	+83.33 %	+29.85 %	+32.76 %	+88.96 %	+107.91 %	+64.67 %	+57.90%

- **Superior performance** (**up to 141%** in precision) compared MCRS benchmarks
- GNN-based approach shows superior results to DNN or MF-based approaches

RQ2. Comparison with GNN-based Benchmarks

Method	Metric	TA		YM		RB		YP	
		$K=5$	$K=10$	$K=5$	$K=10$	$K=5$	$K=10$	$K=5$	$K=10$
GC-MC	*Precision@K*	0.0060	0.0055	0.0603	0.0508	0.1543	0.1234	0.0208	0.0175
	Recall@K	0.0157	0.0284	0.0745	0.1246	0.1762	0.2547	0.0533	0.0895
	NDCG@K	0.0112	0.0159	0.0820	0.0978	0.2232	0.2354	0.0409	0.0530
SpectralCF	*Precision@K*	0.0015	0.0016	0.0594	0.0472	0.1655	0.1306	0.0086	0.0081
	Recall@K	0.0054	0.0111	0.0798	0.1202	0.1646	0.2424	0.0190	0.0351
	NDCG@K	0.0037	0.0058	0.0842	0.0963	0.2255	0.2339	0.0148	0.0204
NGCF	*Precision@K*	0.0181	0.0119	0.0814	0.0609	0.1730	0.1380	0.0232	0.0188
	Recall@K	0.0475	0.0646	0.1010	0.1156	0.1777	0.2648	0.0600	0.0985
	NDCG@K	0.0393	0.0454	0.1139	0.1277	0.2372	0.2534	0.0471	0.0600
DGCF	*Precision@K*	0.0265	0.0168	0.0809	0.0618	0.1635	0.1300	0.0223	0.0188
	Recall@K	0.0729	0.0911	0.1020	0.1506	0.1596	0.2411	0.0494	0.0827
	NDCG@K	0.0603	0.0670	0.1150	0.1275	0.2202	0.2296	0.0410	0.0519
LightGCN	*Precision@K*	0.0267	0.0177	0.0771	0.0616	0.1732	0.1382	0.0235	0.0192
	Recall@K	0.0730	0.0929	0.0959	0.1533	0.1778	0.2622	0.0602	0.0987
	NDCG@K	0.0607	0.0671	0.1108	0.1278	0.2384	0.2512	0.0475	0.0603
LightGCN_{MC}	*Precision@K*	0.0283	0.0211	0.0795	0.0656	0.1802	0.1423	0.0267	0.0205
	Recall@K	0.0799	0.0953	0.0977	0.1566	0.1799	0.2651	0.0633	0.1005
	NDCG@K	0.0632	0.0699	0.1122	0.1301	0.2423	0.2566	0.0520	0.0625
CPA-LGC	*Precision@K*	**0.0449**	**0.0273**	**0.1012**	**0.0788**	**0.2177**	**0.1739**	**0.0360**	**0.0276**
	Recall@K	**0.0901**	**0.1053**	**0.1211**	**0.1725**	**0.1863**	**0.2745**	**0.0859**	**0.1286**
	NDCG@K	**0.0830**	**0.0880**	**0.1392**	**0.1532**	**0.2823**	**0.2892**	**0.0713**	**0.0859**
Gain	*Precision@K*	+58.66%	+29.38%	+25.09%	+20.12%	+20.81%	+22.21%	+34.83%	+34.63%
	Recall@K	+12.77%	+10.49%	+18.73%	+10.15%	+3.04%	+1.55%	+35.70%	+28.22%
	NDCG@K	+31.33%	+25.89%	+21.04%	+17.76%	+16.51%	+12.70%	+37.12%	+37.44%

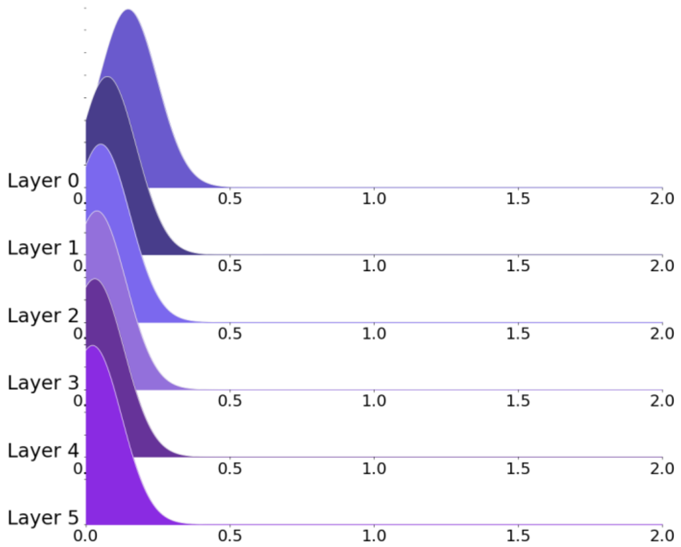
- **Superior performance** (**up to 58.66%** in precision) compared to GNN benchmarks
- Large gain is also observed compared to the **naïve GNN implementation using MC ratings** ($LightGCN_{MC}$)



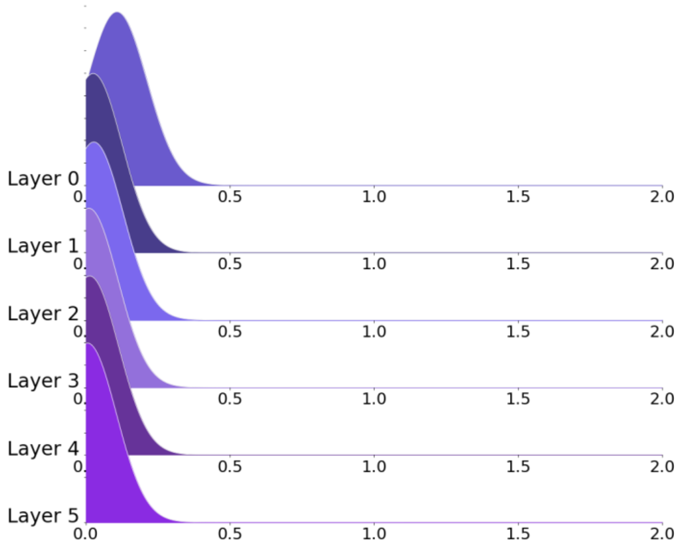
RQ3. Over-Smoothing Alleviation

Dataset: TripAdvisor

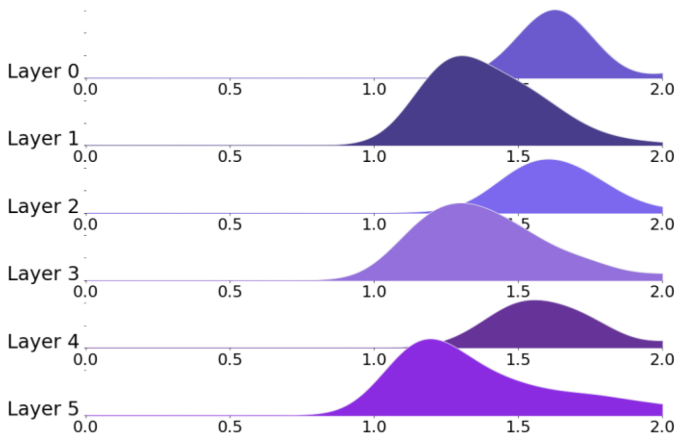
Distribution of pairwise distance of representations



(1) Single-criterion graph



(2) MC expansion graph **w/o PairNorm**

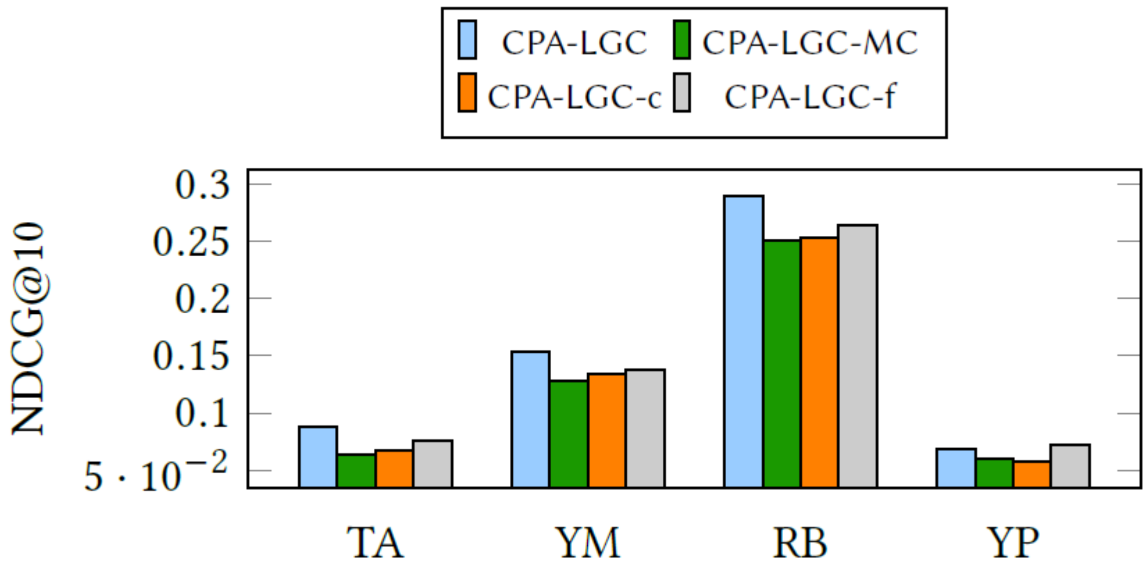


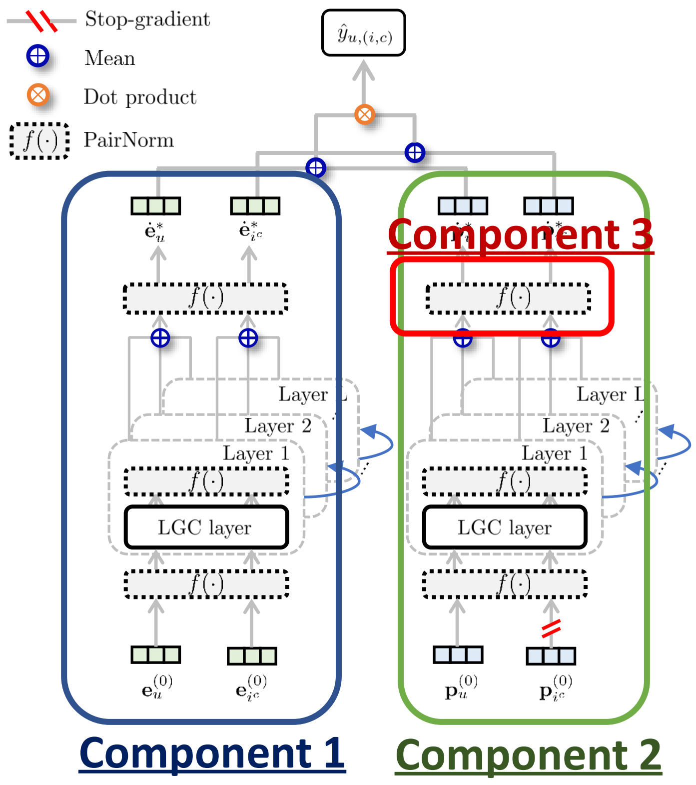
(3) MC expansion graph **w/ PairNorm**

	Layer	0	1	2	3	4	5
(1)	Baseline graph, LGC w/o $f(.)$	0.092	0.052	0.036	0.028	0.022	0.018
(2)	MC expansion graph, LGC w/o $f(.)$	0.092	0.043	0.031	0.027	0.019	0.014
(3)	MC expansion graph, LGC w/ $f(.)$	2.061	1.414	2.546	1.447	2.420	1.369

- Over-smoothing is intensified in the MC expansion graph
- It is relieved by the layer-wise PairNorm

RQ4. Ablation Study



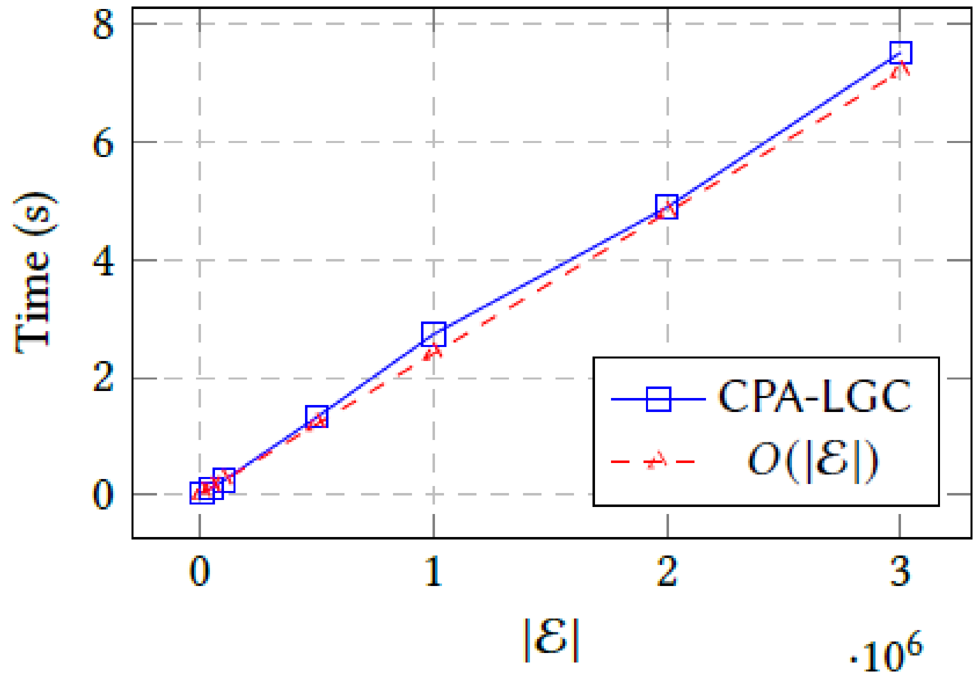


- CPA-LGC-MC: model w/o MC expansion graph (graph construction with overall ratings)
- CPA-LGC-c: model w/o user criteria preference / criterion embeddings (w/o component 2)
- CPA-LGC-f: model w/o layer-wise PairNorm (w/o component 3)

* for YP, PairNorm is not useful since it's γ value is the least.

RQ5. Scalability of CPA-LGC

THEOREM 3.1. *The computational complexity of CPA-LGC is at most linear in* $|\mathcal{E}|$.



- Our CPA-LGC can be easily implemented with simple matrix multiplications

PairNorm

$$\dot{\mathbf{E}}^{(l)} = f\left(\tilde{\mathbf{A}}\mathbf{E}^{(l-1)}\right) \qquad \dot{\mathbf{P}}^{(l)} = f\left(\tilde{\mathbf{A}}\mathbf{P}^{(l-1)}\right)$$

Normalized adj

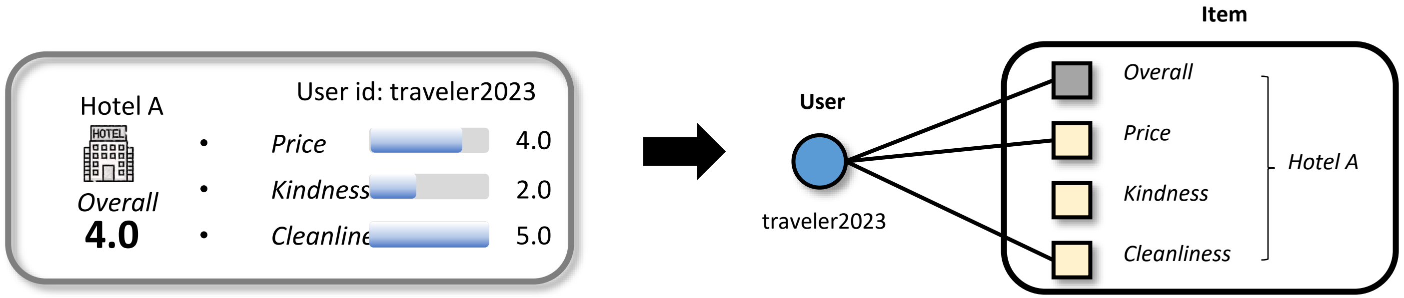
- Computational complexity of CPA-LGC is *linear* in the number of edges in the given graph

- The empirical evaluation also supports our theoretical claim.

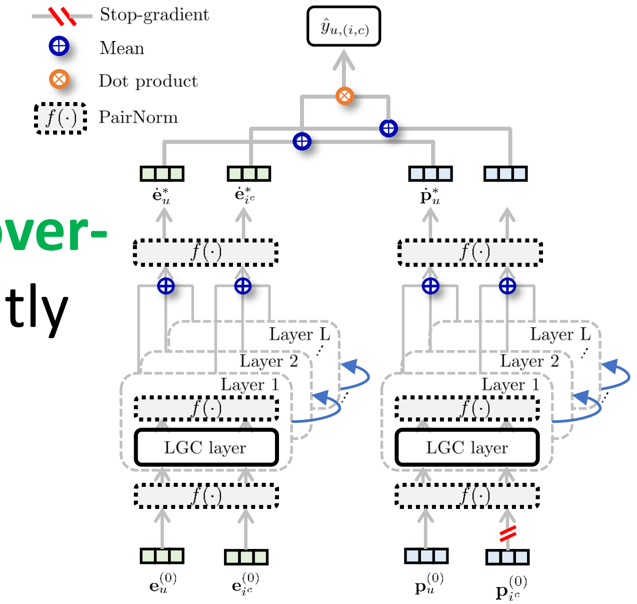
Takeaways

Takeaways

- We devise a novel **graph construction** method in MCRS, called **MC expansion graph**, to capture **complex semantics** in MC ratings via multi-layer graph convolution



- **CPA-LGC** is **light and effective** GNN, which can **prevent possible over-smoothing problem** in the MC expansion graph along with explicitly capturing **criteria preference of users**



CPA-LGC

Thank you for your attention!

Contact: jindeok6@yonsei.ac.kr

Website QR

Appendix: Types of Feedbacks in RS

Explicit feedback:

- Explicit input by users regarding their interest in products.

e.g.) user ratings (1~5 scores), preference (thumbs-up/down button)

Implicit feedback

- Indirectly reflect opinion through observing user behavior

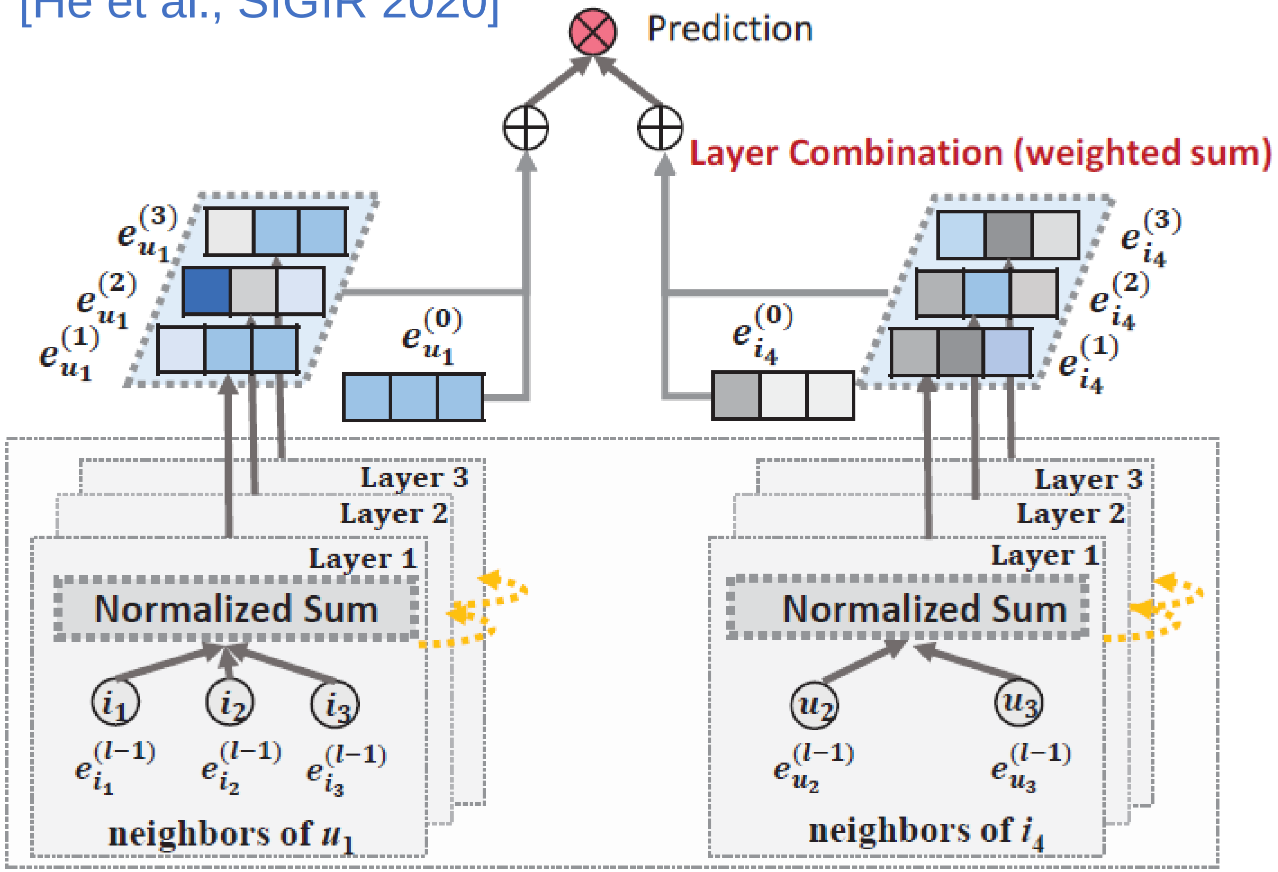
e.g.) purchase history, browsing history, search patterns, or even mouse movement.

Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 2009.

Appendix: LightGCN

LightGCN

[He et al., SIGIR 2020]



- ✓ One of the powerful and efficient GNN-based RS
- ✓ Removing non-linearity and feature transformation

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

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

He et al. "Lightgcn: Simplifying and powering graph convolution network for recommendation." SIGIR 2020.