



Collaborative Filtering : From a Deep Learning Perspective

2025-07-31

Presenter : Sooho Moon

DMAIS

■ Recommender System

■ Collaborative Filtering

■ NeuMF

■ VAE for CF

■ LightGCN

Recommender system

Too Much Information

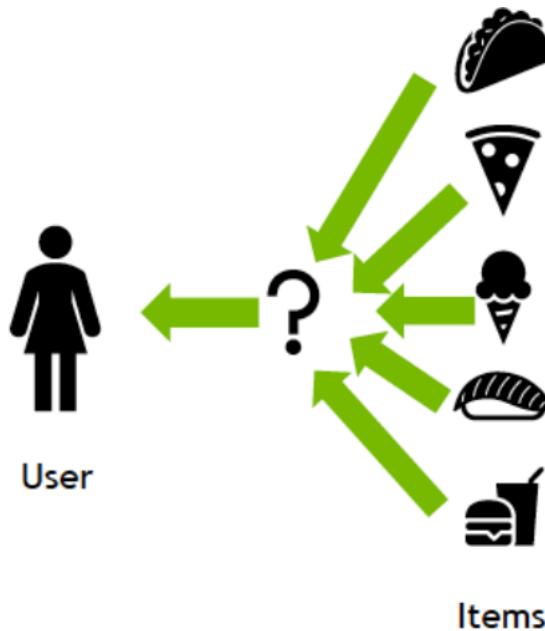
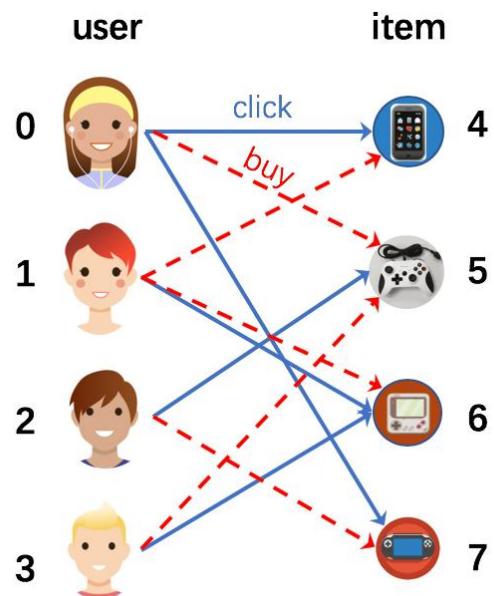
- So many options in the modern world
- We don't have time to surf through everything



Recommender system

■ Need for Recommender System

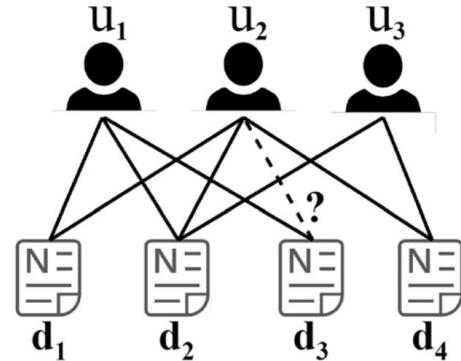
- Variety of external data (e.g., click history, user social network, item meta data)
- Use them to predict user's preference



Recommender system

■ Recommender Systems In the Real-World

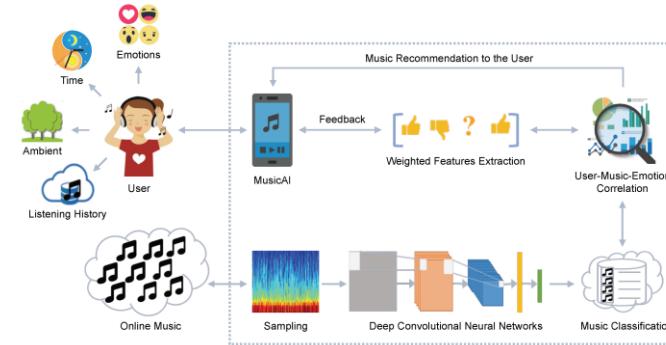
- Core objective : automatically recommend **items** to **users** based on data
- Applicable to various real-world problems



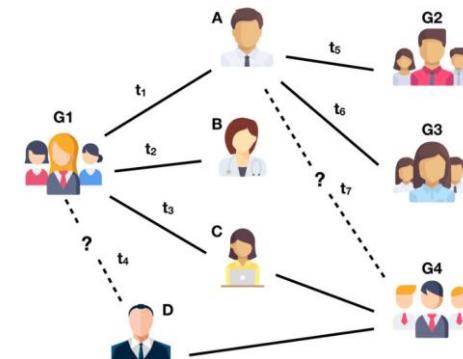
Personalized news recommendation

Patients	Symptoms				Drugs		
u ₁ : Lisa	s ₁ ⁽¹⁾	s ₂ ⁽¹⁾	s ₃ ⁽¹⁾	s ₄ ⁽¹⁾	d ₁ ⁽¹⁾	d ₂ ⁽¹⁾	d ₃ ⁽¹⁾
u ₂ : Jack	s ₁ ⁽²⁾	s ₂ ⁽²⁾	s ₃ ⁽²⁾	s ₄ ⁽²⁾	d ₁ ⁽²⁾	d ₂ ⁽²⁾	d ₃ ⁽²⁾
u ₃ : Mary	s ₁ ⁽³⁾	s ₂ ⁽³⁾	s ₃ ⁽³⁾		d ₁ ⁽³⁾	d ₂ ⁽³⁾	

Set-to-set drug recommendation



Personalized music recommendation



Social network recommendation

Meng, Xiangfu, et al. "A survey of personalized news recommendation." *Data Science and Engineering* 8.4 (2023): 396-416..

Abdul, Ashu, et al. "An emotion-aware personalized music recommendation system using a convolutional neural networks approach." *Applied Sciences* 8.7 (2018): 1103.

Tan, Yanchao, et al. "4sdrug: Symptom-based set-to-set small and safe drug recommendation." *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2022.

Sha, Hao, Mohammad Al Hasan, and George Mohler. "Group link prediction using conditional variational autoencoder." *Proceedings of the International AAAI Conference on Web and Social Media*. Vol. 15. 2021.

Recommender system

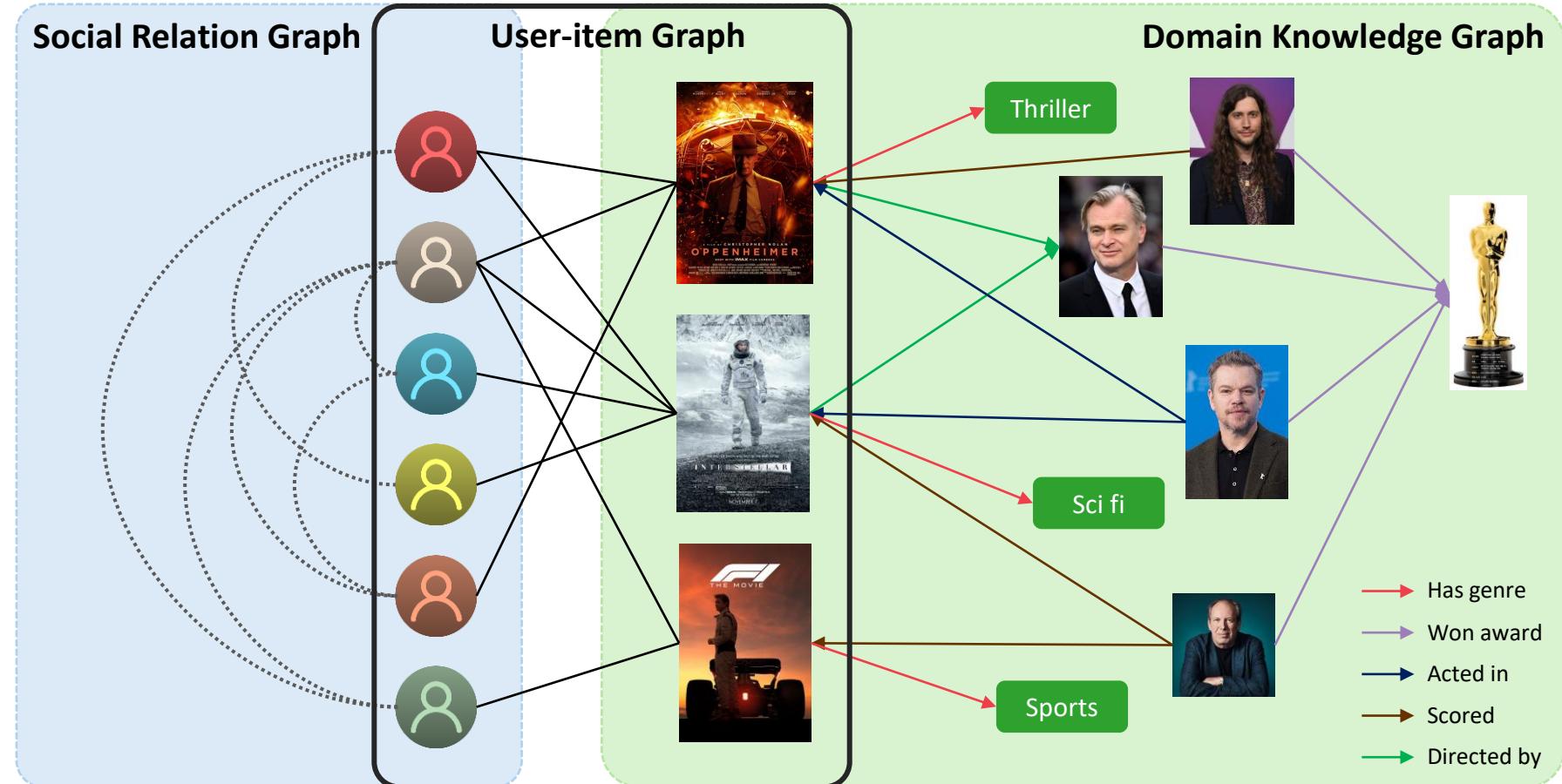
■ How Can We Approach This?

Content based filtering

Collaborative filtering

Hybrid recommendation

etc.



Recommender system

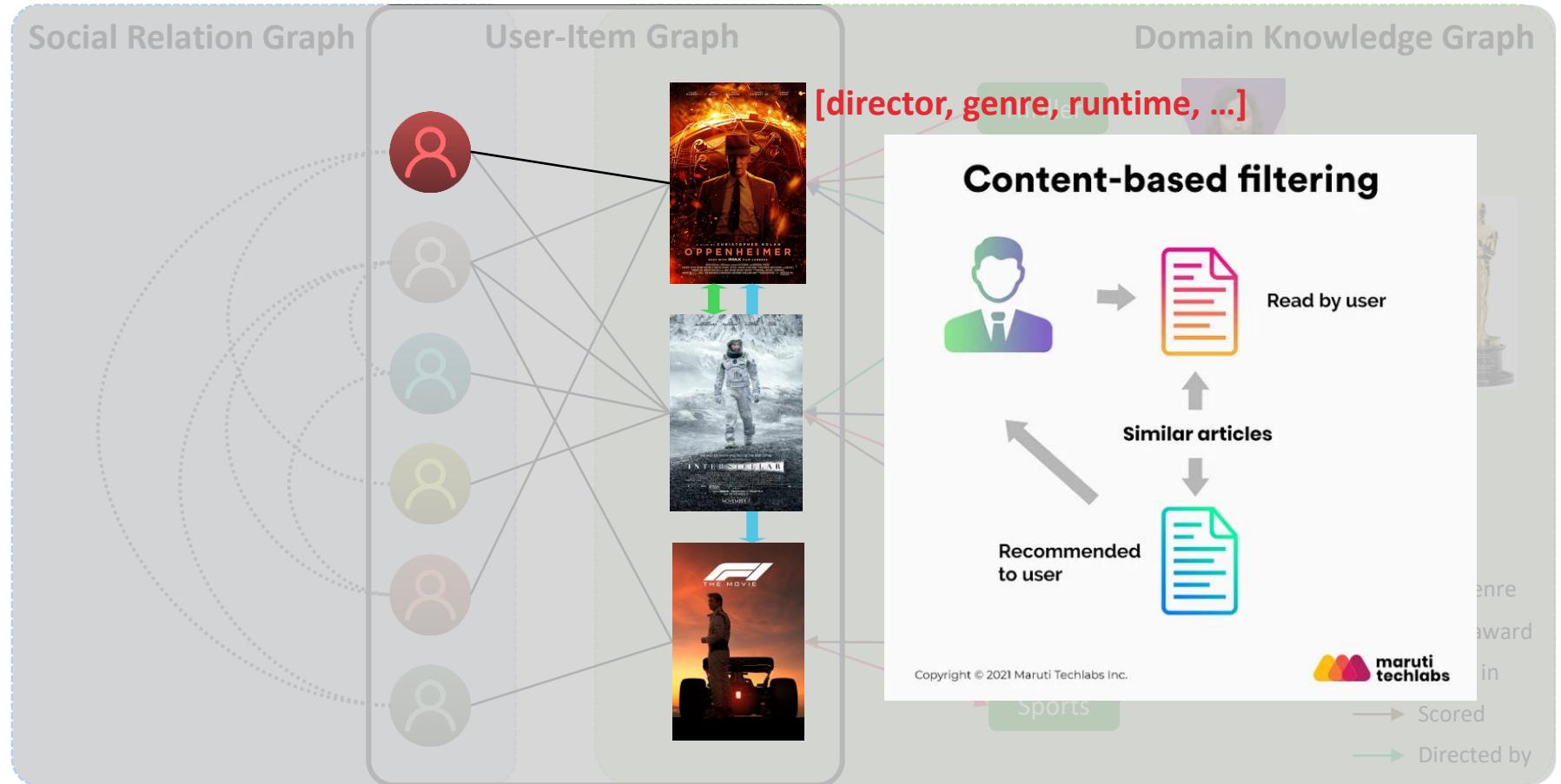
■ How Can We Approach This?

Content based filtering

Collaborative filtering

Hybrid recommendation

etc.



Recommender system

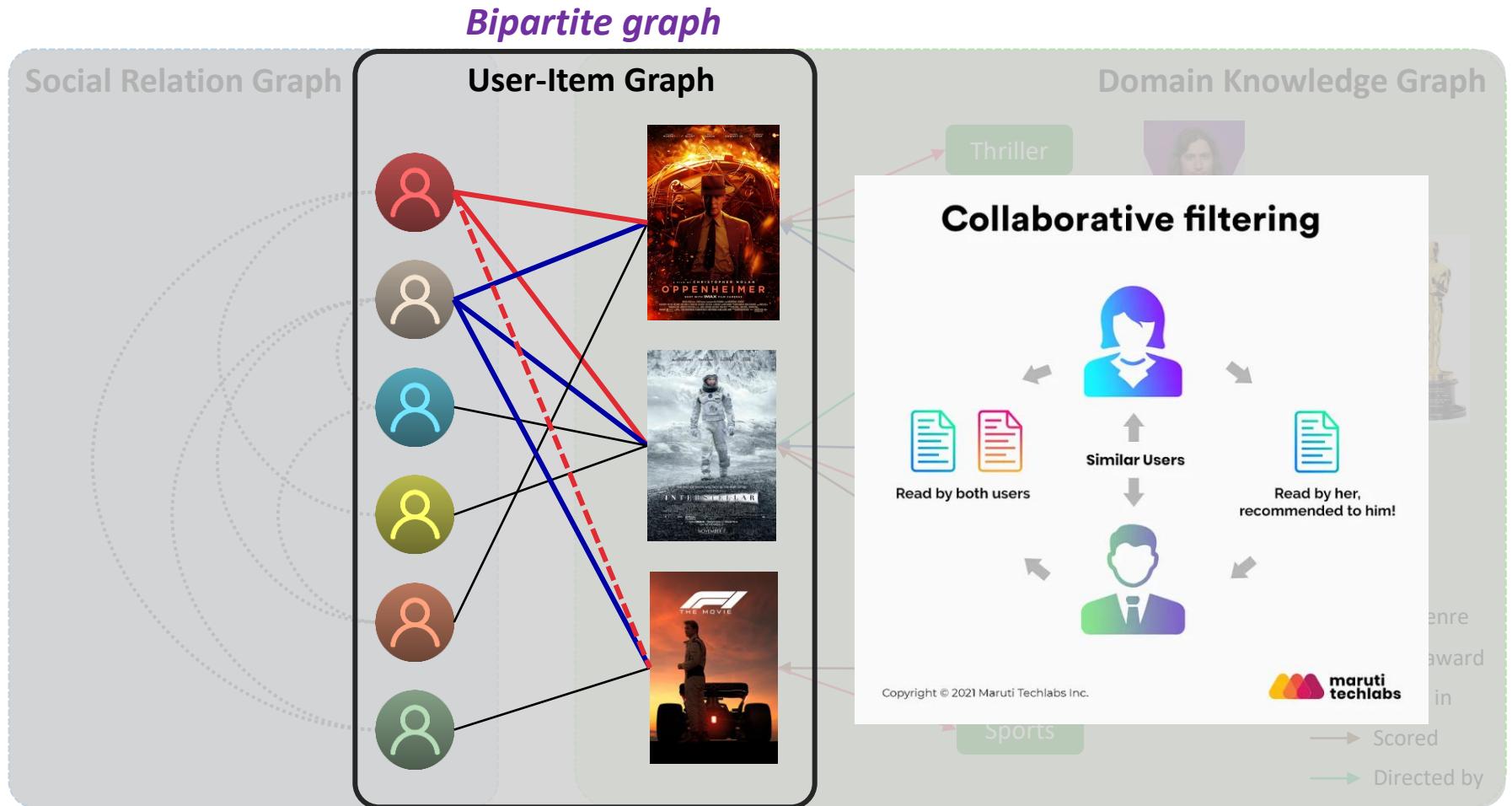
■ How Can We Approach This?

Content based filtering

Collaborative filtering

Hybrid recommendation

etc.



Recommender system

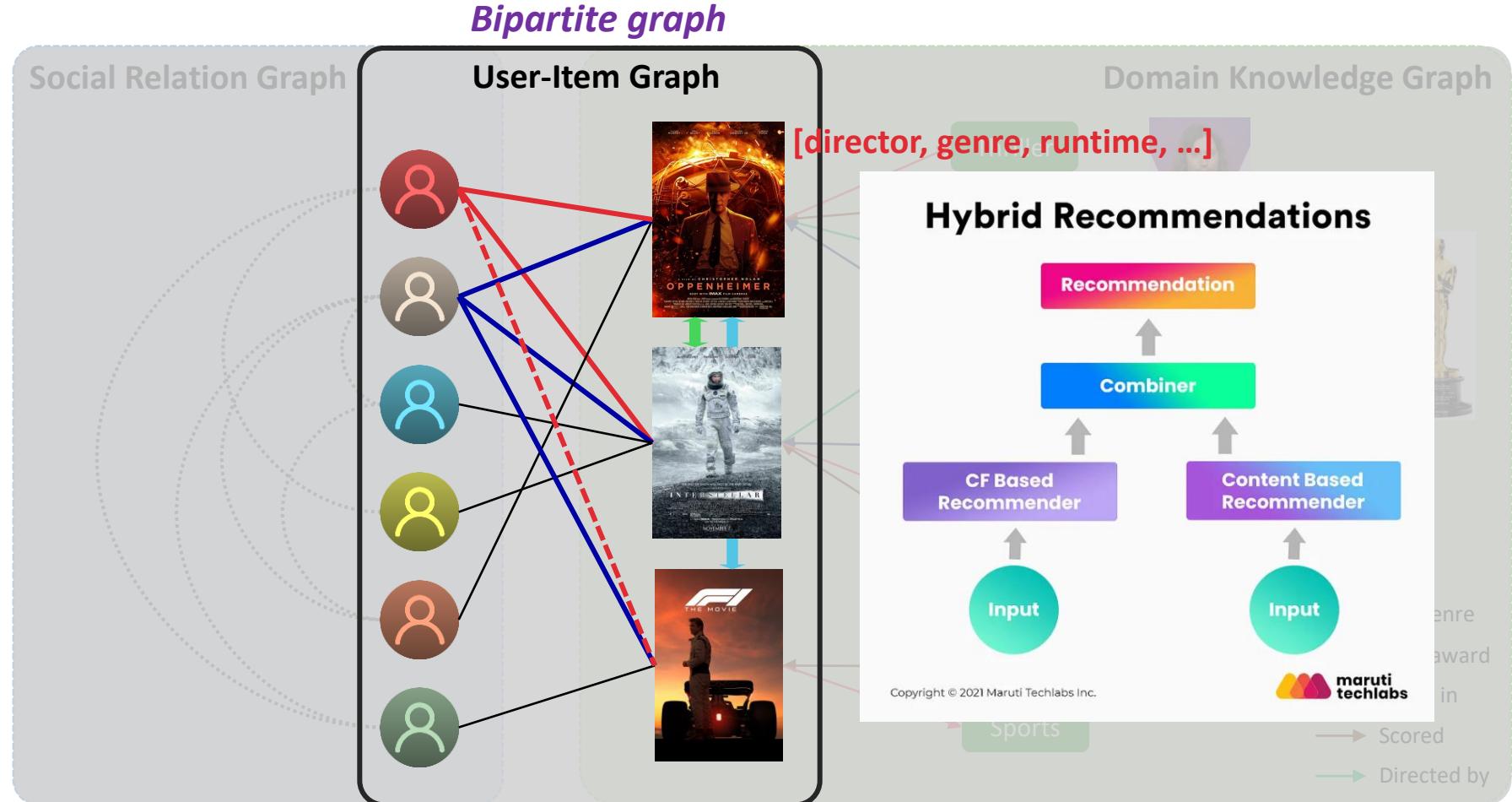
■ How Can We Approach This?

Content based filtering

Collaborative filtering

Hybrid recommendation

etc.



Recommender system

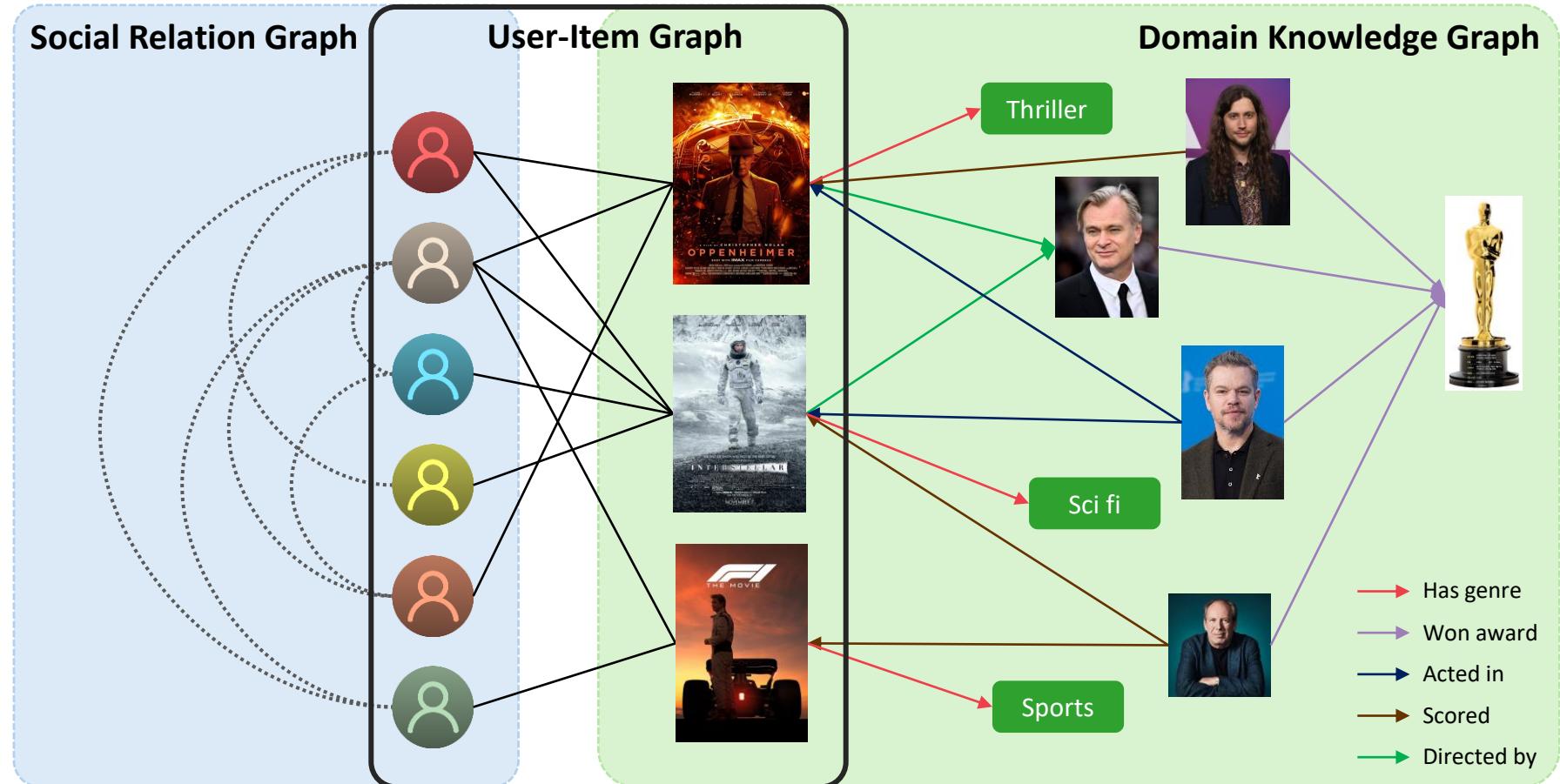
■ How Can We Approach This?

Content based filtering

Collaborative filtering

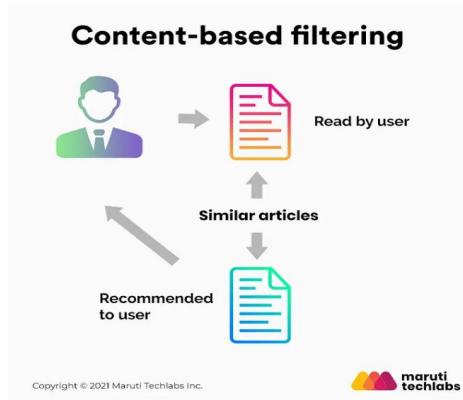
Hybrid recommendation

etc.



Collaborative Filtering

■ Why Collaborative Filtering (CF)?



Pros

- No external user data needed
- System easy to scale
- Relatively good at cold start

Cons

- Construction of good item features
- Can't capture complex patterns across users

Pros

- No need for domain-specific features
- Find patterns directly from user behavior
- Only requires user-item interaction matrix

Cons

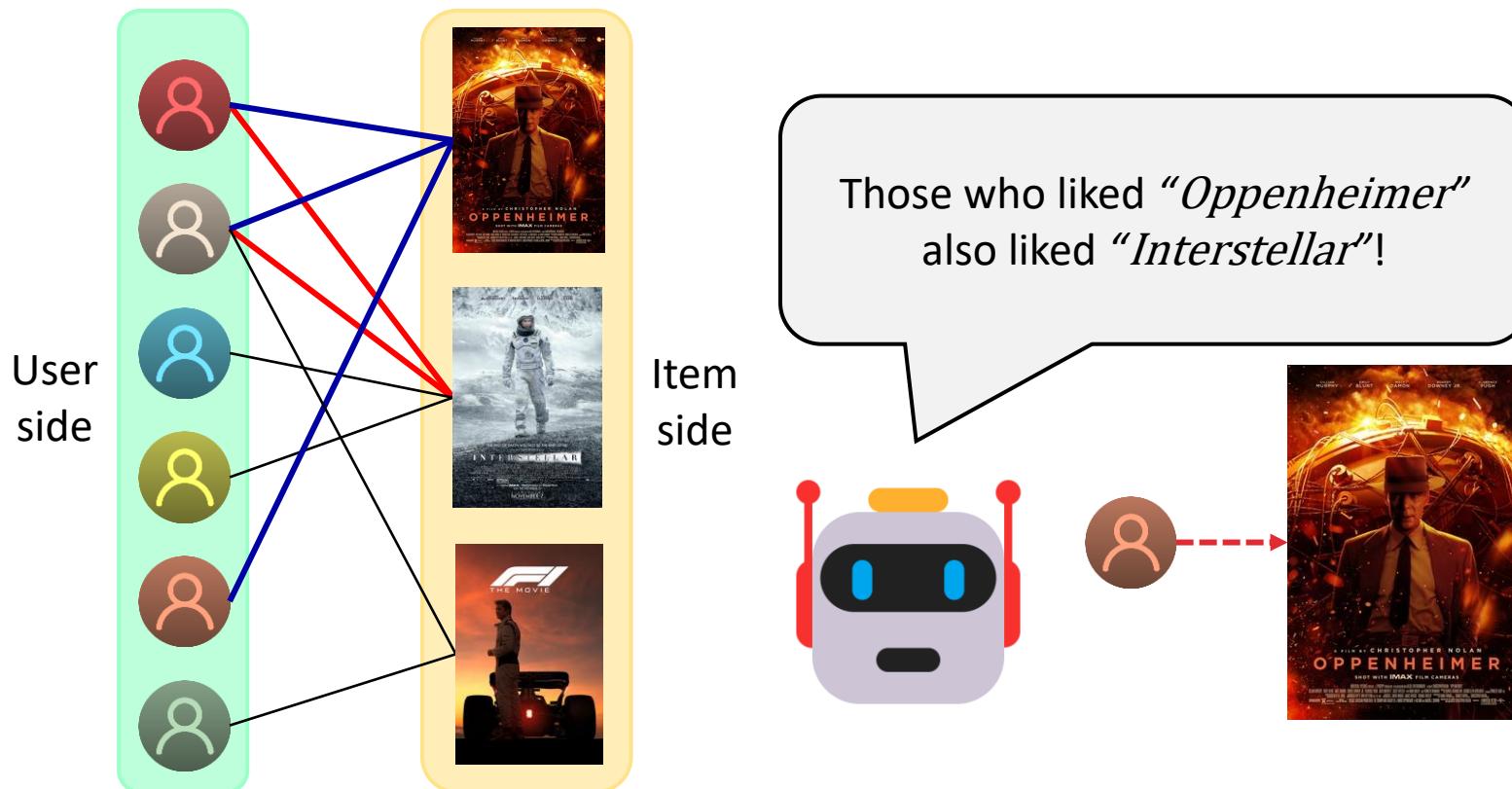
- Relatively struggles at cold start
- Scalability issues

Let's focus on collaborative filtering!

Collaborative Filtering

■ Generalized Flow of CF

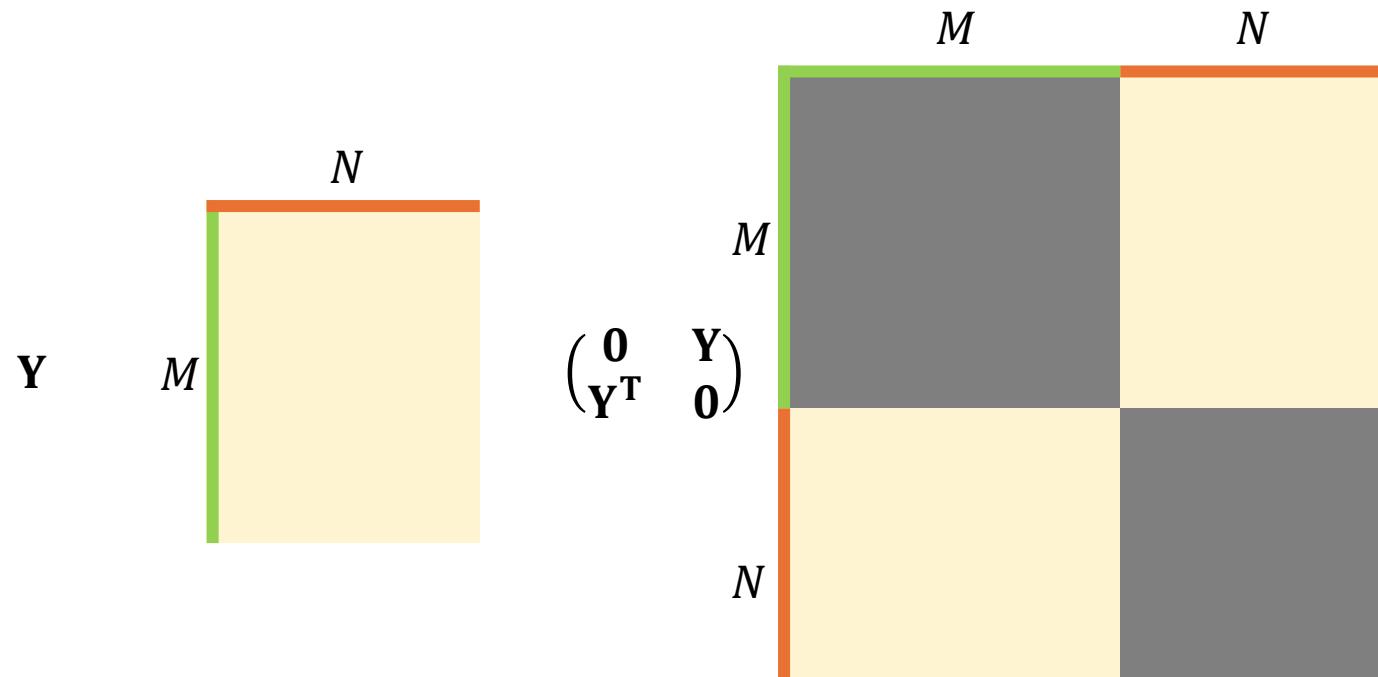
- Collaborative (**users influence each other**) filtering (**recommend items**)
- Underlying graph = bipartite graph, link = rating (e.g., 1 to 5) or interaction (e.g., 0 or 1)



Collaborative Filtering

■ Preliminaries : Notations

- User ($u \in U, |U| = M$), item ($i \in I, |I| = N$)
- User-item graph ($\mathbf{Y} \in \mathbb{R}^{M \times N}$)
- Binarized rating matrix = click matrix \rightarrow **implicit feedback**



Collaborative Filtering

■ Preliminaries : Metrics

□ Recall@K

$$\frac{\# \text{of relevant items in top} - K}{\text{Total } \# \text{ of relevant items}}$$

□ NDCG@K

$$\text{normalize}\left(\sum_{i=1}^K \frac{\text{rel}_i}{\log_2(i+1)}\right)$$

□ HR@K

$$\mathbb{I}[\text{rank} \leq K]$$

□ Matrix Factorization

□ General Framework

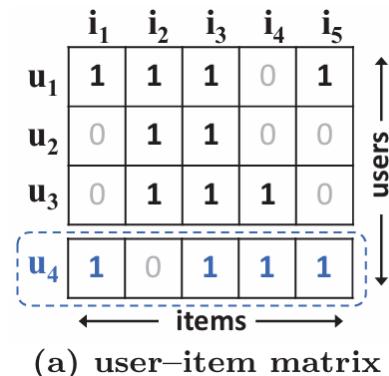
□ Three Models

□ Experiments

□ Conclusion

Xiangnan He
National University of
Singapore, Singapore
xiangnanhe@gmail.com

Liqiang Nie
Shandong University
China
nieliqiang@gmail.com

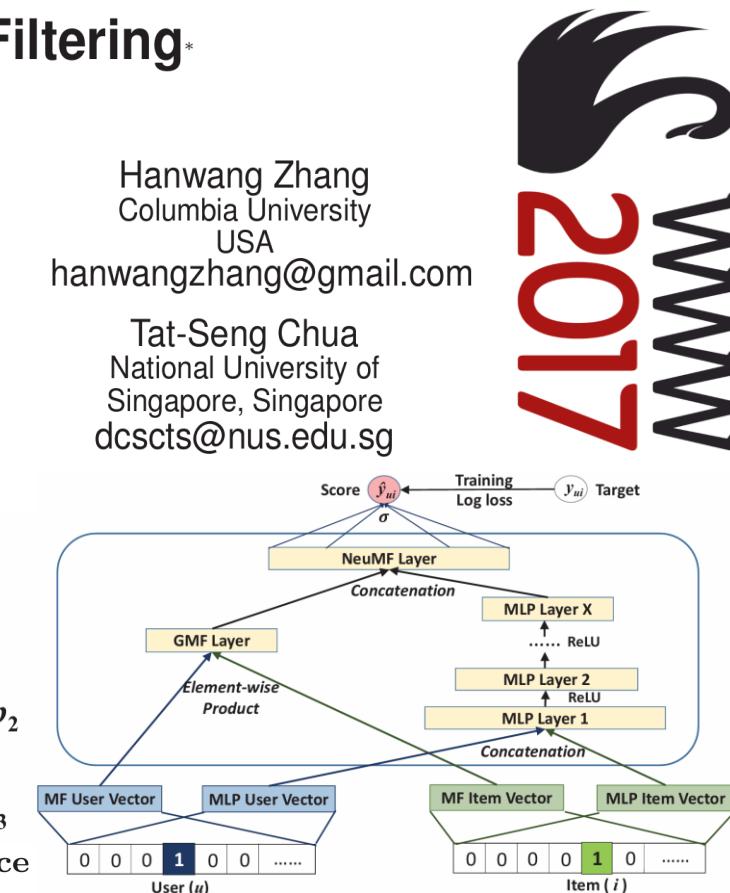
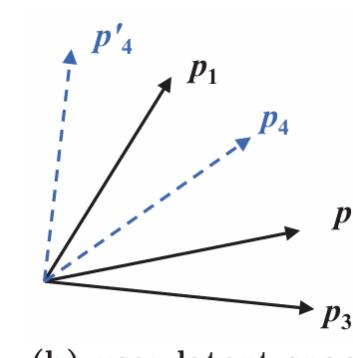


Lizi Liao
National University of
Singapore, Singapore
liaolizi.llz@gmail.com

Xia Hu
Texas A&M University
USA
hu@cse.tamu.edu

Hanwang Zhang
Columbia University
USA
hanwangzhang@gmail.com

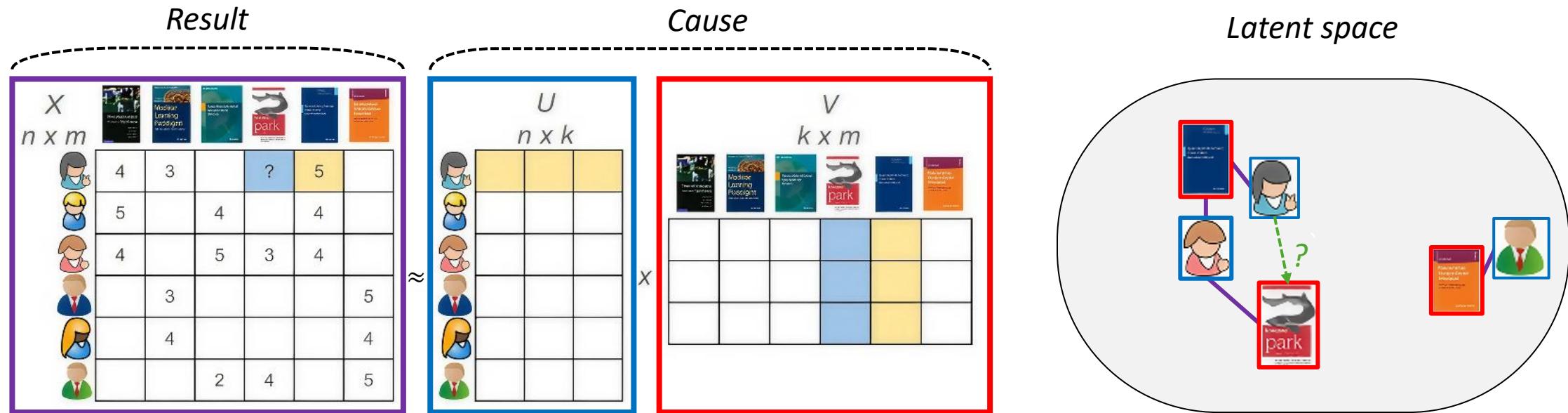
Tat-Seng Chua
National University of
Singapore, Singapore
dcscts@nus.edu.sg



NeuMF / Matrix Factorization

Matrix Factorization : 'De facto' Approach of CF

- User-item matrix as explicit/implicit data
- User-item matrix \approx user matrix \times item matrix
- Modeling the cause for new recommendations



NeuMF / Matrix Factorization

■ ML with MF

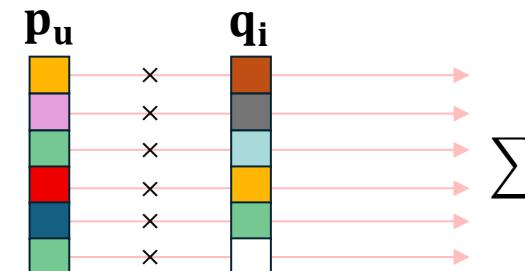
- Objective : real rating(r_{ui}) \approx predicted rating(\hat{r}_{ui})
- Inner product of user(\mathbf{p}_u) and item(\mathbf{q}_i) directly used for prediction
- Handcrafting all the features

$$\hat{r}_{ui} = \mu + b_u + b_i + \mathbf{p}_u^\top \mathbf{q}_i$$

■ Key Limitations

- (1) Inherent limitation of ML's **feature engineering**
- (2) Independency between features \rightarrow prediction is a simple **linear combination**

$$\begin{aligned} & \min_{\mathbf{p}, \mathbf{q}, \mathbf{b}, \mathbf{v}, \mathbf{d}} \sum_{(u, i) \in \mathcal{K}} (r_{ui} - \hat{r}_{ui})^2 + \lambda_1 (\|\mathbf{p}_u\|^2 + \|\mathbf{q}_i\|^2) + \lambda_2 (b_u^2 + b_i^2) + \lambda_3 \sum_{n \in N_i} \|\mathbf{v}_n\|^2 & (6) \\ & \min_{\mathbf{p}, \mathbf{q}, \mathbf{b}, \mathbf{v}, \mathbf{d}} \sum_{(u, i) \in \mathcal{K}} (r_{ui} - \hat{r}_{ui})^2 + \lambda_1 (\|\mathbf{p}_u\|^2 + \|\mathbf{q}_i\|^2) + \lambda_2 (b_u^2 + b_i^2) + \lambda_3 \left(\sum_{n \in N_i} \|\mathbf{v}_n\|^2 + \sum_{c \in C_i} \|\mathbf{d}_c\|^2 \right) & (7) \\ & \min_{\mathbf{p}, \mathbf{q}, \mathbf{b}, \mathbf{v}, \mathbf{d}} \sum_{(u, i) \in \mathcal{K}} (r_{ui} - \hat{r}_{ui})^2 + \lambda_1 \left(\|\mathbf{p}_u\|^2 + \sum_{w \in R_i} \|\mathbf{q}_w\|^2 \right) + \lambda_2 (b_u^2 + b_i^2) + \lambda_3 \left(\sum_{n \in N_i} \|\mathbf{v}_n\|^2 + \sum_{c \in C_i} \|\mathbf{d}_c\|^2 \right) & (8) \\ & \min_{\mathbf{p}, \mathbf{q}, \mathbf{b}, \mathbf{v}, \mathbf{d}, \beta} \sum_{(u, i) \in \mathcal{K}} (r_{ui} - \hat{r}_{ui})^2 + \lambda_1 \left(\|\mathbf{p}_u\|^2 + \sum_{w \in R_i} \|\mathbf{q}_w\|^2 \right) + \lambda_2 (b_u^2 + b_i^2 + \beta_i^2 + \beta_u^2) + \lambda_3 \left(\sum_{n \in N_i} \|\mathbf{v}_n\|^2 + \sum_{c \in C_i} \|\mathbf{d}_c\|^2 \right) & (9) \end{aligned}$$

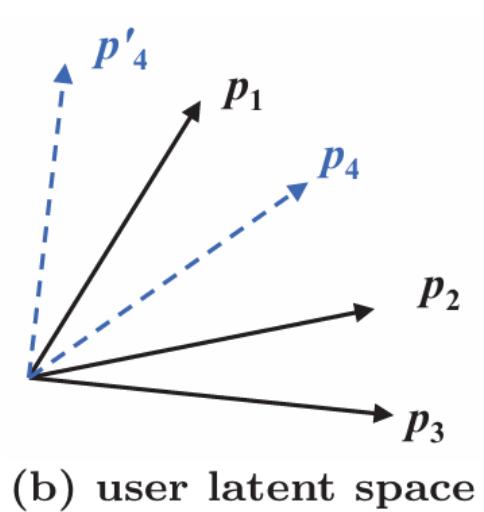
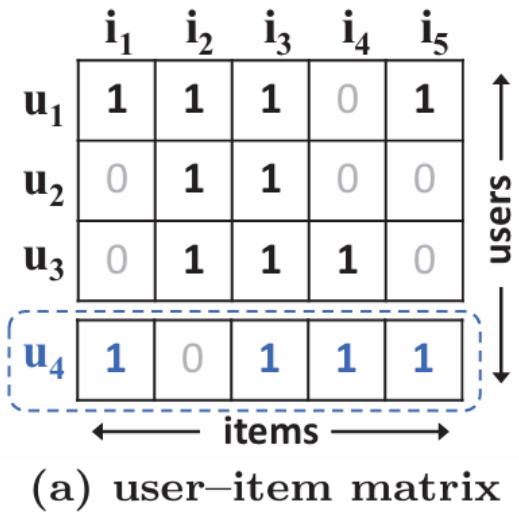


NeuMF / Matrix Factorization

Lack of Expressiveness

- WLOG, MF needs to recover similarity between two users (Jaccard Coefficient)
- Position of p_4 can't be placed without ranking loss
- Higher dimension = solution? / Prone to overfit

$$s_{ij} = \frac{|\mathcal{R}_i \cap \mathcal{R}_j|}{|\mathcal{R}_i \cup \mathcal{R}_j|} = \frac{\text{AND}}{\text{OR}}$$



$$s_{23}(0.66) > s_{12}(0.5) > s_{13}(0.4)$$

Consider user 4

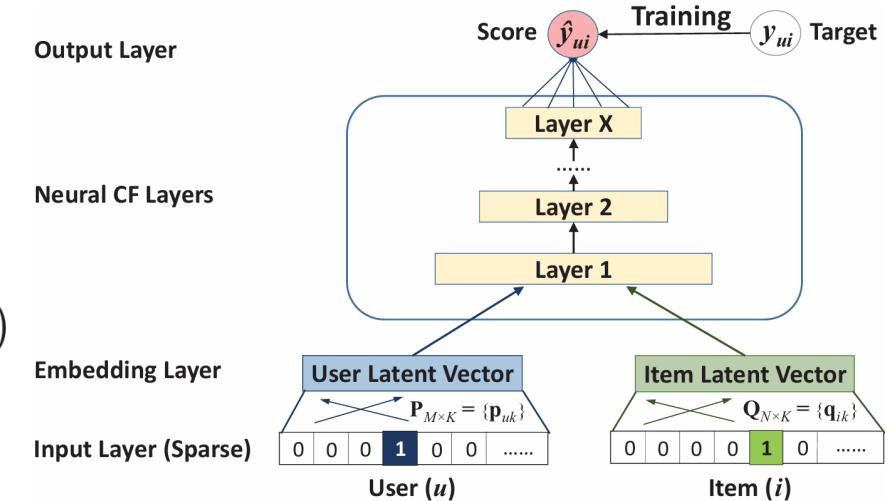
$$s_{41}(0.6) > s_{43}(0.4) > s_{42}(0.2)$$

NeuMF / General Framework

■ NCF's Predictive Model

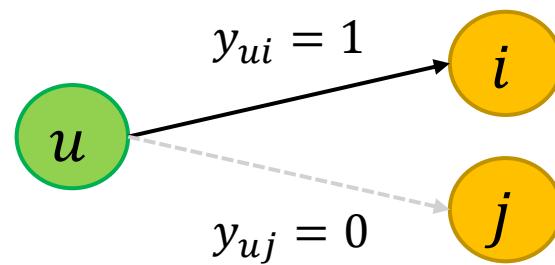
$$\hat{y}_{ui} = f(\mathbf{P}^T \mathbf{v}_u^U, \mathbf{Q}^T \mathbf{v}_i^I | \mathbf{P}, \mathbf{Q}, \Theta_f)$$

$$f(\mathbf{P}^T \mathbf{v}_u^U, \mathbf{Q}^T \mathbf{v}_i^I) = \phi_{out}(\phi_X(\dots \phi_2(\phi_1(\mathbf{P}^T \mathbf{v}_u^U, \mathbf{Q}^T \mathbf{v}_i^I))\dots))$$



■ Learning Objective for NCF

- Focus is on NN modeling → consider only pointwise learning



Pointwise : $\underset{\mathbf{P}, \mathbf{Q}, \Theta_f}{\operatorname{argmin}} |y_{ui} - \hat{y}_{ui}|$

Pairwise : $\underset{\mathbf{P}, \mathbf{Q}, \Theta_f}{\operatorname{argmax}} (\hat{y}_{ui} - \hat{y}_{uj})$

* The objective terms of argmin and argmax are abstract and can vary depending on the situation.

NeuMF / General Framework

■ Pointwise Learning NCF

- Assuming Gaussian distribution, existing methods used **regression** + **squared loss**
- However, recommendation is about recommend (1) or not (0) problem

$$L_{sqr} = \sum_{(u,i) \in \mathcal{Y} \cup \mathcal{Y}^-} w_{ui} (\underline{y_{ui}} - \underline{\hat{y}_{ui}})^2$$



Binary cross entropy problem!

$$\begin{aligned} L &= - \sum_{(u,i) \in \mathcal{Y}} \log \hat{y}_{ui} - \sum_{(u,j) \in \mathcal{Y}^-} \log(1 - \hat{y}_{uj}) \\ &= - \sum_{(u,i) \in \mathcal{Y} \cup \mathcal{Y}^-} y_{ui} \log \hat{y}_{ui} + (1 - y_{ui}) \log(1 - \hat{y}_{ui}) \end{aligned}$$

NeuMF / Three Models

■ Generalized Matrix Factorization (GMF)

- Single layer ϕ_1 (dot product)
- If $a_{out}(x) = x$ (activation) and $\mathbf{h} = \mathbf{1} \in \mathbb{R}^K$ (edge weights of output layer)

$$\mathbf{p}_u = \mathbf{P}^T \mathbf{v}_u^U \quad \mathbf{q}_i = \mathbf{Q}^T \mathbf{v}_i^I$$

$$\phi_1(\mathbf{p}_u, \mathbf{q}_i) = \mathbf{p}_u \odot \mathbf{q}_i$$

$$\hat{y}_{ui} = a_{out}(\mathbf{h}^T (\underline{\mathbf{p}_u \odot \mathbf{q}_i}))$$

Generalizes MF methods!

$$\hat{r}_{ui} = \mu + b_u + b_i + \underline{\mathbf{p}_u^\top \mathbf{q}_i}$$

NeuMF / Three Models

■ Multi-Layer Perceptron (MLP)

- Concatenate → MLP
- Closer to the output layer, smaller the layer size (abstracting features)

$$\mathbf{z}_1 = \phi_1(\mathbf{p}_u, \mathbf{q}_i) = \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix},$$

$$\phi_2(\mathbf{z}_1) = a_2(\mathbf{W}_2^T \mathbf{z}_1 + \mathbf{b}_2),$$

.....

$$\phi_L(\mathbf{z}_{L-1}) = a_L(\mathbf{W}_L^T \mathbf{z}_{L-1} + \mathbf{b}_L),$$

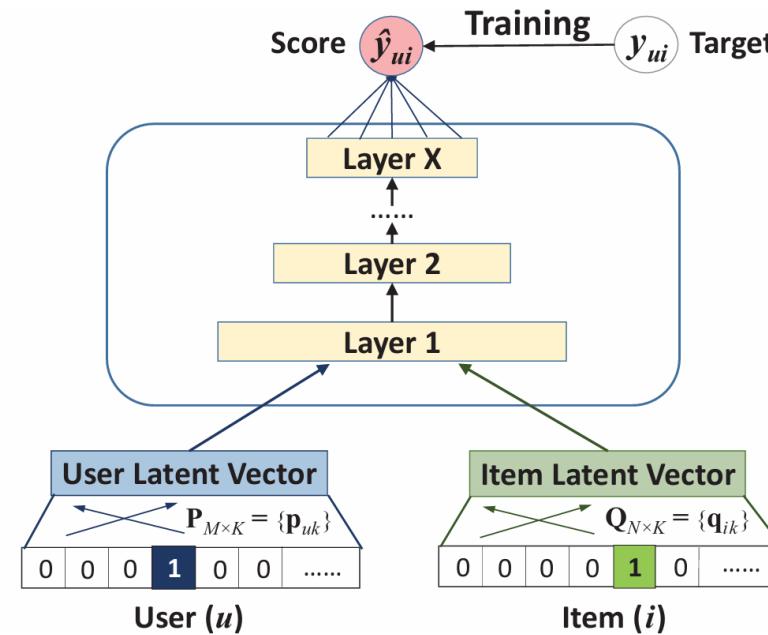
$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \phi_L(\mathbf{z}_{L-1})),$$

Output Layer

Neural CF Layers

Embedding Layer

Input Layer (Sparse)



NeuMF / Three Models

■ Fusion of GMF and MLP (NeuMF)

- Sharing embeddings of GMF and MLP limits the performance
- However, non-convexity of objective function leads to underfitting

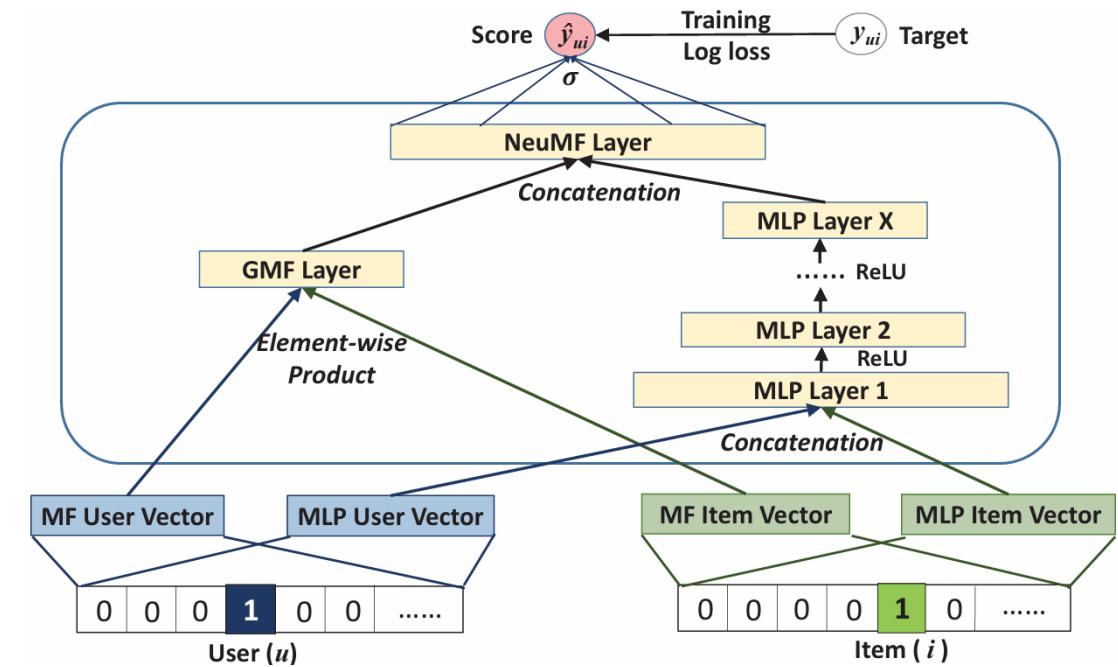
$$\hat{y}_{ui} = \sigma(\mathbf{h}^T a(\mathbf{p}_u \odot \mathbf{q}_i + \mathbf{W} \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix} + \mathbf{b}))$$

\downarrow
Manage own embeddings

$$\phi^{GMF} = \mathbf{p}_u^G \odot \mathbf{q}_i^G,$$

$$\phi^{MLP} = a_L(\mathbf{W}_L^T(a_{L-1}(\dots a_2(\mathbf{W}_2^T \begin{bmatrix} \mathbf{p}_u^M \\ \mathbf{q}_i^M \end{bmatrix} + \mathbf{b}_2) \dots)) + \mathbf{b}_L),$$

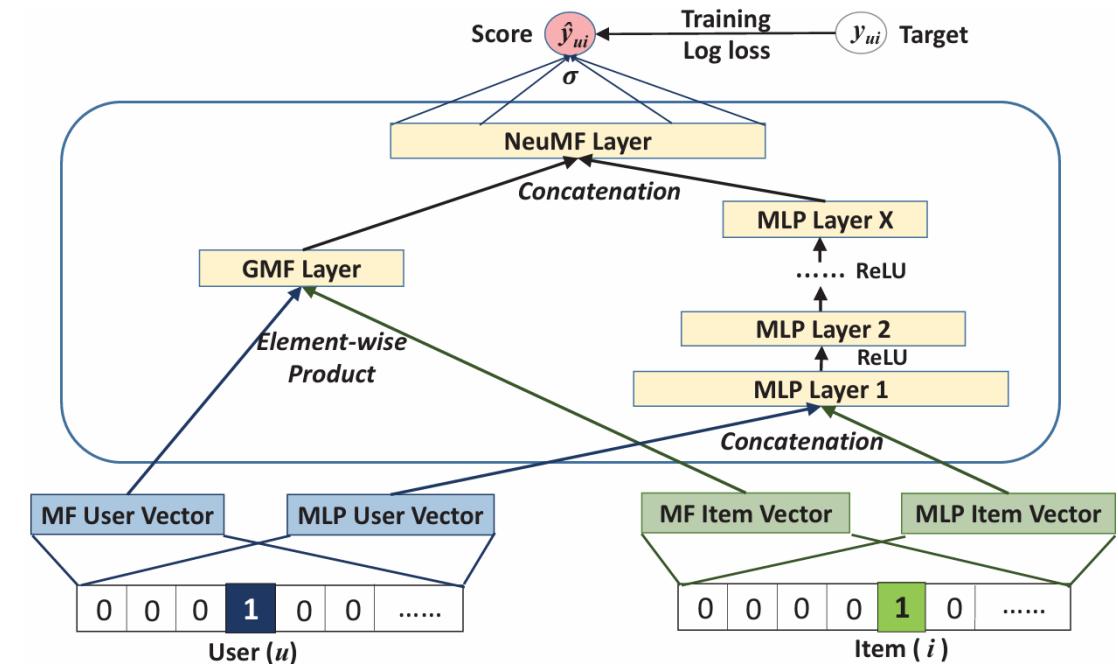
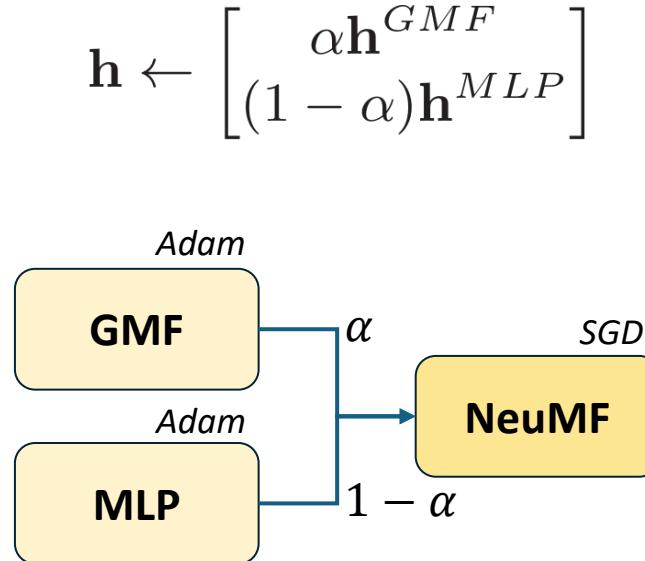
$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \begin{bmatrix} \phi^{GMF} \\ \phi^{MLP} \end{bmatrix}),$$



NeuMF / Three Models

■ Fusion of GMF and MLP (NeuMF) : Pre-Training

- Initialization is important (Erhan et al.) → initialize NeuMF with pretrained GMF and MLP
- Only NeuMF layer is updated



NeuMF / Experiments

■ Research Questions

- RQ1** : Do NCF methods outperform SOTA methods?
- RQ2** : Does the log loss with negative sampling work?
- RQ3** : Does deep learning help?

■ Dataset

Dataset	Interaction#	Item#	User#	Sparsity
MovieLens	1,000,209	3,706	6,040	95.53%
Pinterest	1,500,809	9,916	55,187	99.73%

■ Evaluation Protocol

- User's latest interaction is in the test set
- Rank items : random sample 100 items not interacted by user
- Metrics : HR@K, NDCG@K

NeuMF / Experiments

■ RQ1 : Do NCF Methods Outperform SOTA Methods?

□ Predictive factor = last layer's dimension

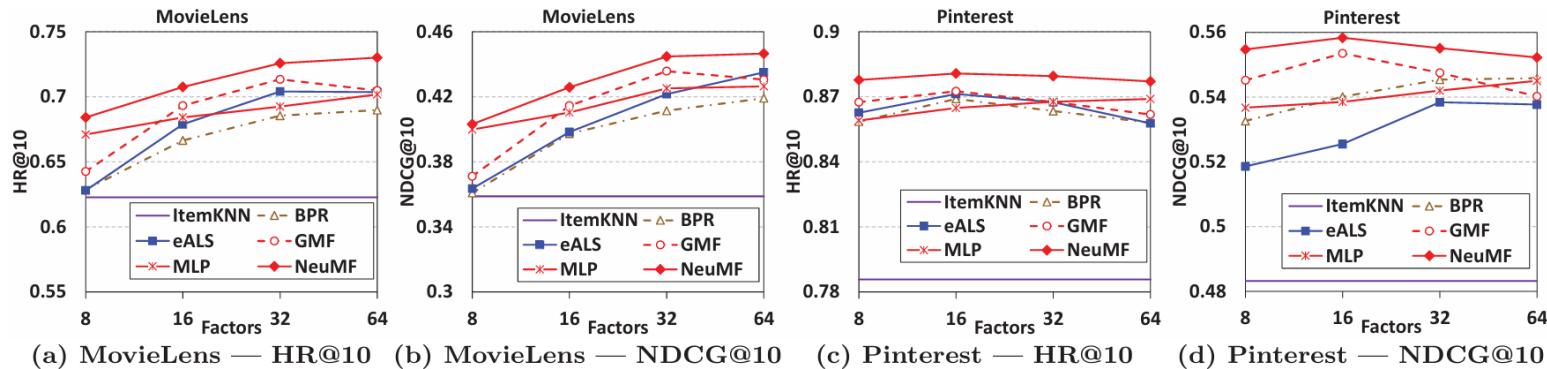


Figure 4: Performance of HR@10 and NDCG@10 w.r.t. the number of predictive factors on the two datasets.

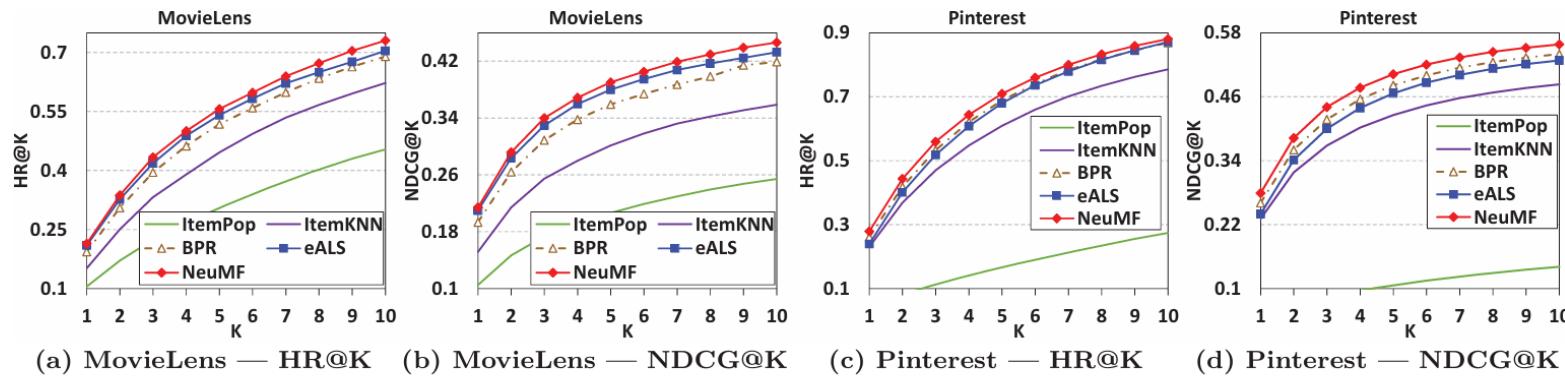


Figure 5: Evaluation of Top-K item recommendation where K ranges from 1 to 10 on the two datasets.

NeuMF / Experiments

■ RQ1 : Do NCF Methods Outperform SOTA Methods?

- Affect of pre-training

Factors	With Pre-training		Without Pre-training	
	HR@10	NDCG@10	HR@10	NDCG@10
MovieLens				
8	0.684	0.403	0.688	0.410
16	0.707	0.426	0.696	0.420
32	0.726	0.445	0.701	0.425
64	0.730	0.447	0.705	0.426
Pinterest				
8	0.878	0.555	0.869	0.546
16	0.880	0.558	0.871	0.547
32	0.879	0.555	0.870	0.549
64	0.877	0.552	0.872	0.551

NeuMF / Experiments

■ RQ2 : Does the Log Loss with Negative Sampling Work?

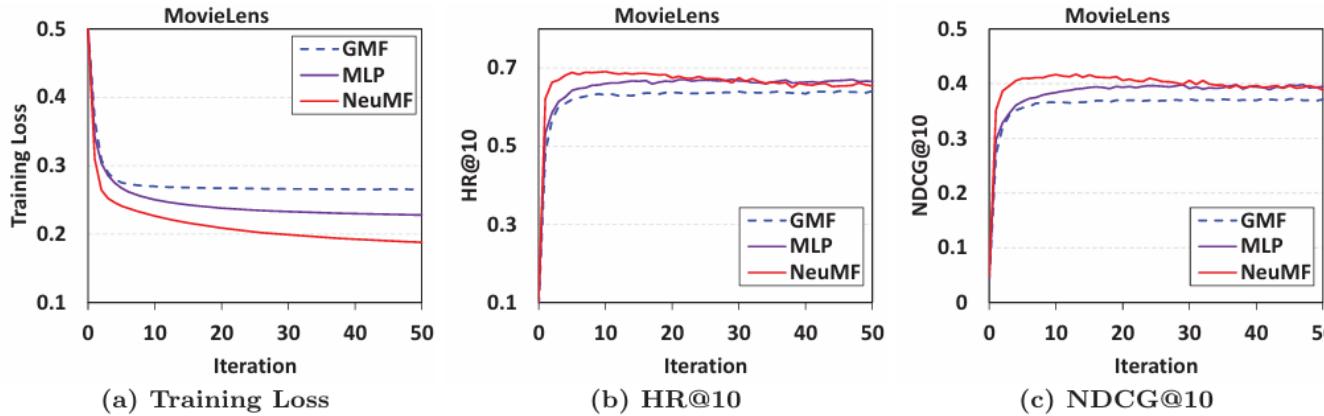


Figure 6: Training loss and recommendation performance of NCF methods *w.r.t.* the number of iterations on MovieLens (factors=8).

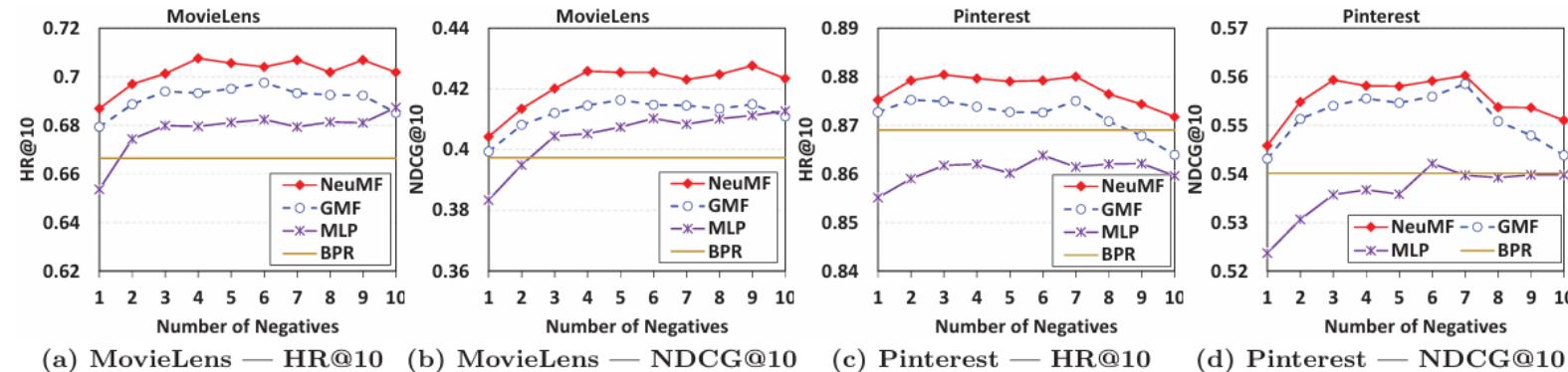


Figure 7: Performance of NCF methods *w.r.t.* the number of negative samples per positive instance (factors=16). The performance of BPR is also shown, which samples only one negative instance to pair with a positive instance for learning.

NeuMF / Experiments

■ RQ3 : Does Deep Learning Help?

MLP-0 : no hidden layers

Table 3: HR@10 of MLP with different layers.

Factors	MLP-0	MLP-1	MLP-2	MLP-3	MLP-4
MovieLens					
8	0.452	0.628	0.655	0.671	0.678
16	0.454	0.663	0.674	0.684	0.690
32	0.453	0.682	0.687	0.692	0.699
64	0.453	0.687	0.696	0.702	0.707
Pinterest					
8	0.275	0.848	0.855	0.859	0.862
16	0.274	0.855	0.861	0.865	0.867
32	0.273	0.861	0.863	0.868	0.867
64	0.274	0.864	0.867	0.869	0.873

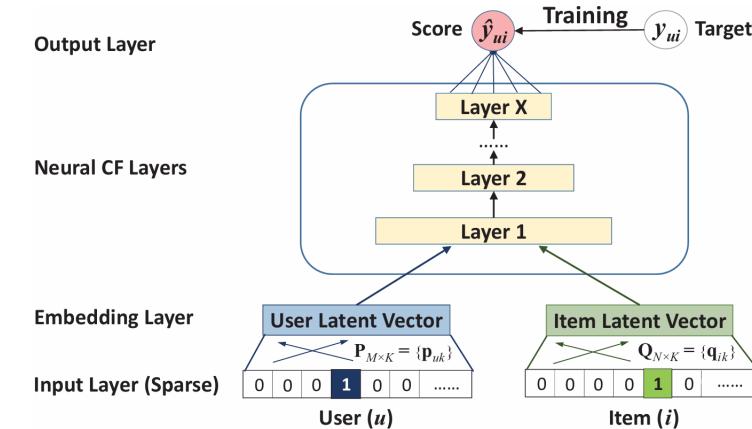
Table 4: NDCG@10 of MLP with different layers.

Factors	MLP-0	MLP-1	MLP-2	MLP-3	MLP-4
MovieLens					
8	0.253	0.359	0.383	0.399	0.406
16	0.252	0.391	0.402	0.410	0.415
32	0.252	0.406	0.410	0.425	0.423
64	0.251	0.409	0.417	0.426	0.432
Pinterest					
8	0.141	0.526	0.534	0.536	0.539
16	0.141	0.532	0.536	0.538	0.544
32	0.142	0.537	0.538	0.542	0.546
64	0.141	0.538	0.542	0.545	0.550

NeuMF / Conclusion

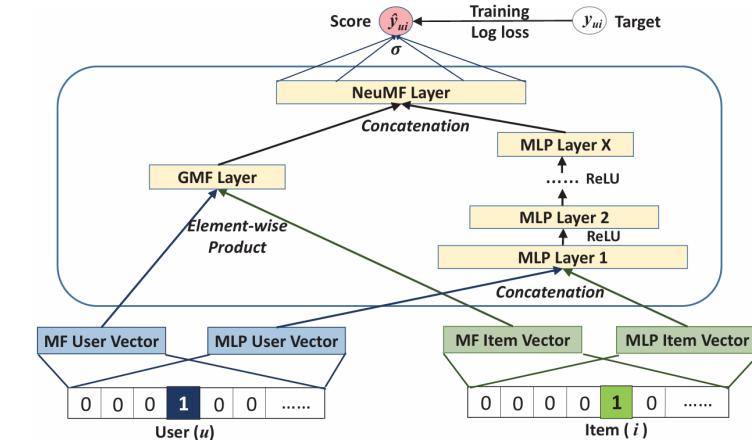
■ Neural Network into CF

- Proposed general framework NCF
- Generalized MF methods into NCF



■ NeuMF

- Proved that neural networks can help CF
- Modularized architecture enables greater flexibility and generality



Mult-VAE

□ Non-Linearity for CF

□ VAE for CF

□ ELBO Variant

□ General View of AE

□ Experiments

□ Conclusion

Variational Autoencoders for Collaborative Filtering

Dawen Liang

Netflix

Los Gatos, CA

dliang@netflix.com

Matthew D. Hoffman

Google AI

San Francisco, CA

mhoffman@google.com

Rahul G. Krishnan

MIT

Cambridge, MA

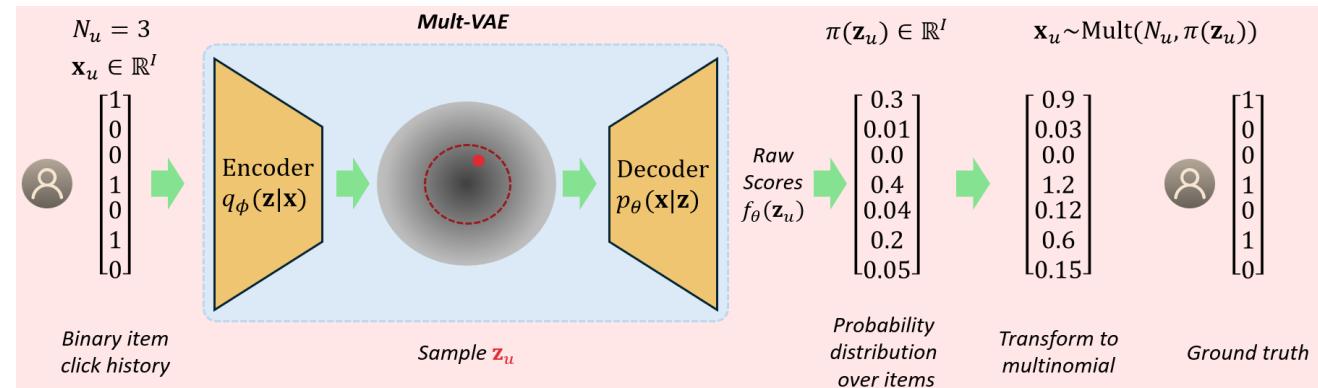
rahulgk@mit.edu

Tony Jebara

Netflix

Los Gatos, CA

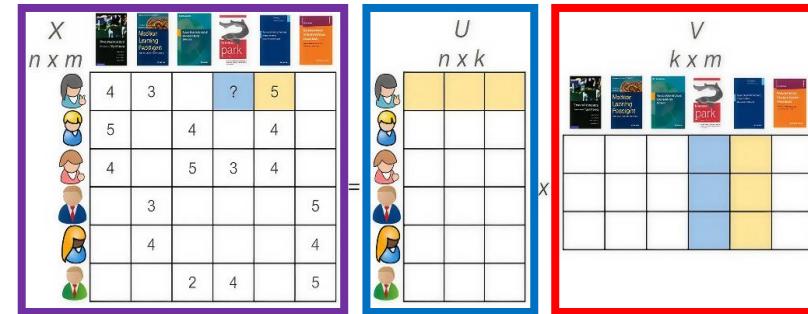
tjebara@netflix.com



Mult-VAE / Non-Linearity for CF

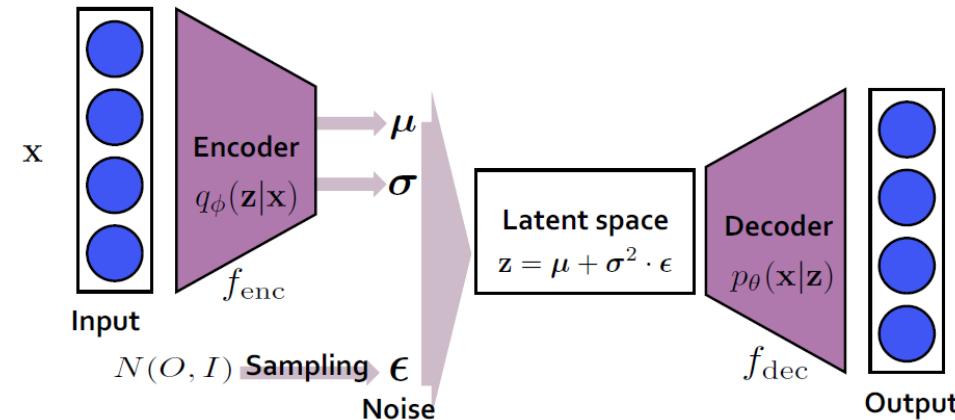
Linear Factor Models

- They did prove its simplicity and effectiveness
- However, inherent linearity limits its capacity



VAE for CF

- VAEs can generalize linear latent-factor models due to non-linearity
- **But how can we apply VAE for CF?**

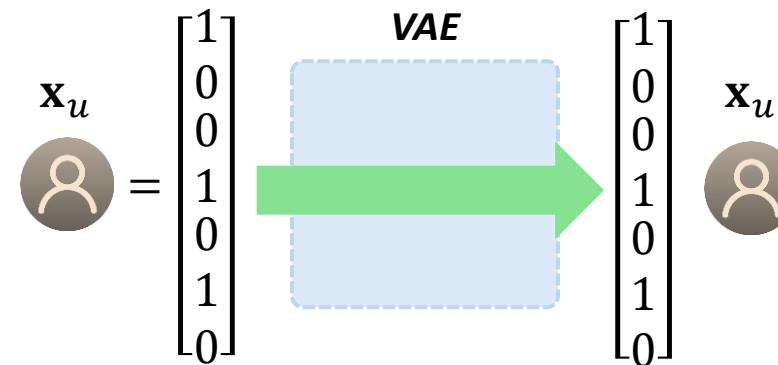


Mult-VAE / Non-Linearity for CF

■ Input and Output?

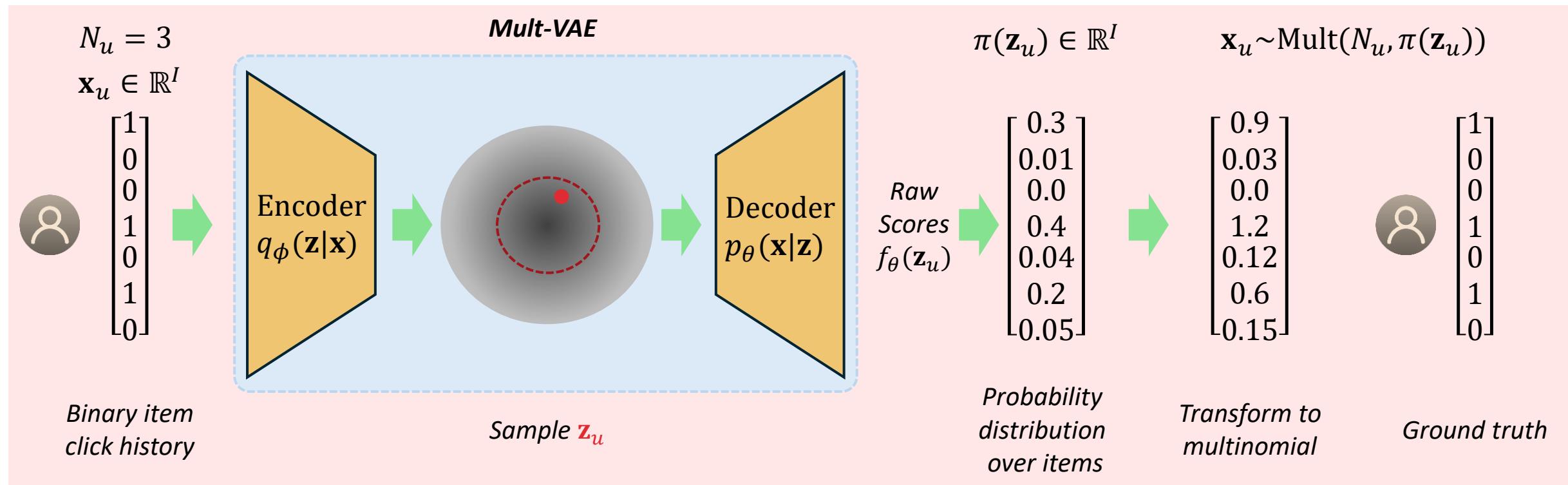
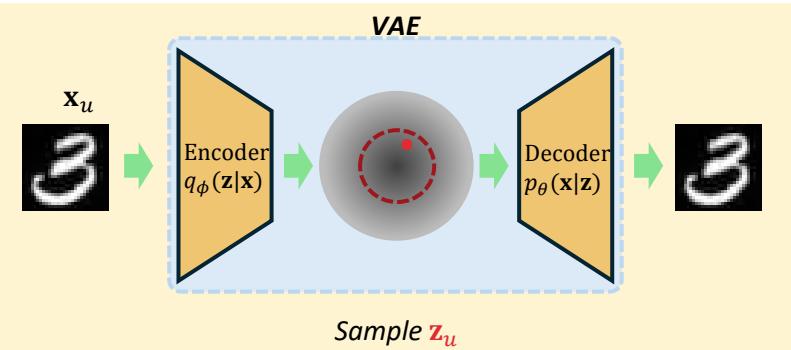
- Input = user, output = predicted preference (i.e., probabilities) over all items
- However, input = output in VAEs
- Solution : binarized click history = input = output = \mathbf{x}_u
- **Objective : generate the multinomial distribution!!!**

$\mathbf{x}_u \sim \text{Mult}(\{\text{total clicks}\}, \{\text{click probability for each item}\})$



Mult-VAE / VAE for CF

■ Overview of Mult-VAE



Mult-VAE / VAE for CF

■ Optimization Process of Mult-VAE

- Based on ELBO, lower bound of the log marginal likelihood

Similarity between encoded input and decoded output (\uparrow)

$$\log p(\mathbf{x}_u; \theta) \geq \mathbb{E}_{q_{\phi}(\mathbf{z}_u | \mathbf{x}_u)} [\log p_{\theta}(\mathbf{x}_u | \mathbf{z}_u)] - \text{KL}(q_{\phi}(\mathbf{z}_u | \mathbf{x}_u) || p(\mathbf{z}_u))$$

* $p(\mathbf{z}_u)$ is assumed to be $\mathcal{N}(\mathbf{0}, \mathbf{I})$ and is approximated with $q_{\phi}(\mathbf{z}_u | \mathbf{x}_u)$

Acts as a **regularization** so that encoded distribution(q_{ϕ}) doesn't deviate too much from the prior(p) (\downarrow)

Mult-VAE / VAE for CF

■ Negative Reconstruction Loss

$$\log p(\mathbf{x}_u; \theta) \geq \mathbb{E}_{q_\phi(\mathbf{z}_u | \mathbf{x}_u)} [\log p_\theta(\mathbf{x}_u | \mathbf{z}_u)] - \text{KL}(q_\phi(\mathbf{z}_u | \mathbf{x}_u) \| p(\mathbf{z}_u))$$

- Assumed multinomial environment → multinomial likelihood most natural
- Gaussian log-likelihood and logistic log-likelihood also possible
- **Author's claim** : multinomial likelihood models click data better

Multinomial likelihood $\log p_\theta(\mathbf{x}_u | \mathbf{z}_u) \stackrel{c}{=} \sum_i x_{ui} \log \pi_i(\mathbf{z}_u)$

Gaussian likelihood $\log p_\theta(\mathbf{x}_u | \mathbf{z}_u) \stackrel{c}{=} - \sum_i \frac{c_{ui}}{2} (x_{ui} - f_{ui})^2$

Logistic log-likelihood $\log p_\theta(\mathbf{x}_u | \mathbf{z}_u) = \sum_i x_{ui} \log \sigma(f_{ui}) + (1 - x_{ui}) \log(1 - \sigma(f_{ui}))$

Mult-VAE / ELBO Variant

■ Balancing Between Two Terms

$$\log p(\mathbf{x}_u; \theta) \geq \mathbb{E}_{q_\phi(\mathbf{z}_u | \mathbf{x}_u)} [\log p_\theta(\mathbf{x}_u | \mathbf{z}_u)] - \text{KL}(q_\phi(\mathbf{z}_u | \mathbf{x}_u) \| p(\mathbf{z}_u))$$

- Primary goal : good recommendation (i.e., good reconstruction)
- Do need to follow up ELBO → partially regularized VAE

Mult – VAE^{PR}

$$\mathcal{L}_\beta(\mathbf{x}_u; \theta, \phi) \equiv \mathbb{E}_{q_\phi(\mathbf{z}_u | \mathbf{x}_u)} [\log p_\theta(\mathbf{x}_u | \mathbf{z}_u)] - \beta \cdot \text{KL}(q_\phi(\mathbf{z}_u | \mathbf{x}_u) \| p(\mathbf{z}_u))$$

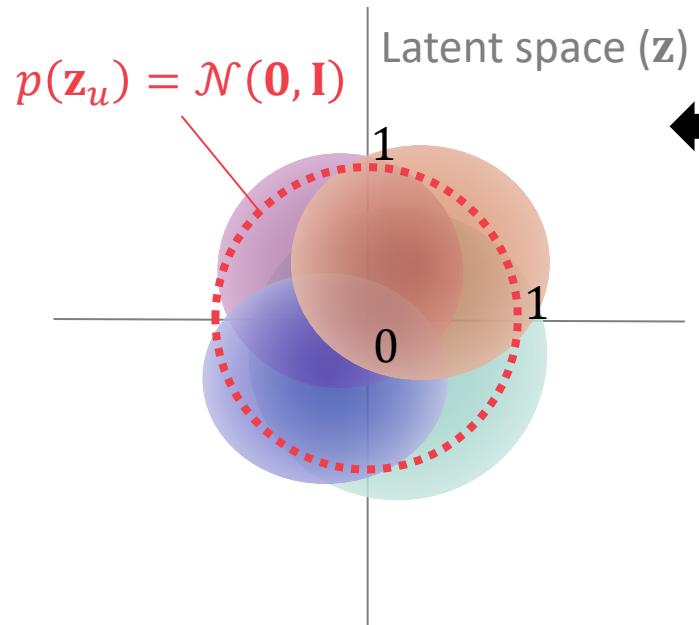
Weak regularization
(Stronger reconstruction) ↓ ← → ↑ Strong regularization
(Weaker reconstruction)

Mult-VAE / ELBO Variant

■ What Does β Control?

- $\beta > 1$ (left): posterior $q_\phi(\mathbf{z}_u|\mathbf{x}_u)$ is close to the prior $p(\mathbf{z}_u) = \mathcal{N}(\mathbf{0}, \mathbf{I})$
- $\beta < 1$ (right): posterior $q_\phi(\mathbf{z}_u|\mathbf{x}_u)$ is far away from the prior $p(\mathbf{z}_u) = \mathcal{N}(\mathbf{0}, \mathbf{I})$

$$\beta \cdot \text{KL}(q_\phi(\mathbf{z}_u|\mathbf{x}_u) \parallel p(\mathbf{z}_u))$$



$p(\mathbf{z}_u) = \mathcal{N}(\mathbf{0}, \mathbf{I})$

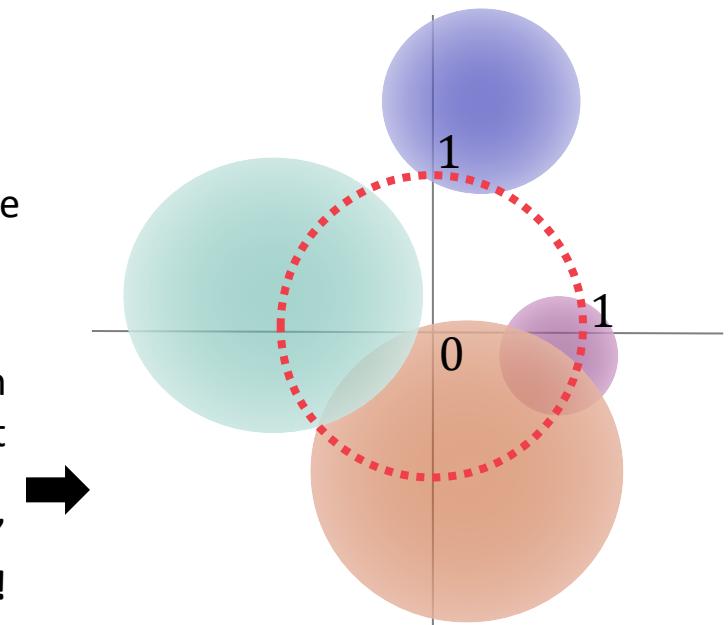
Latent space (z)

Smoother interpolation → Favorable in learning disentangled representations (Higgins, Irina, et al.)

Close resemblance to input data is hard to generate

Good reconstruction → Favorable when reconstruction is more important

Not a proper “generative model”
Suited for recommendation!!!



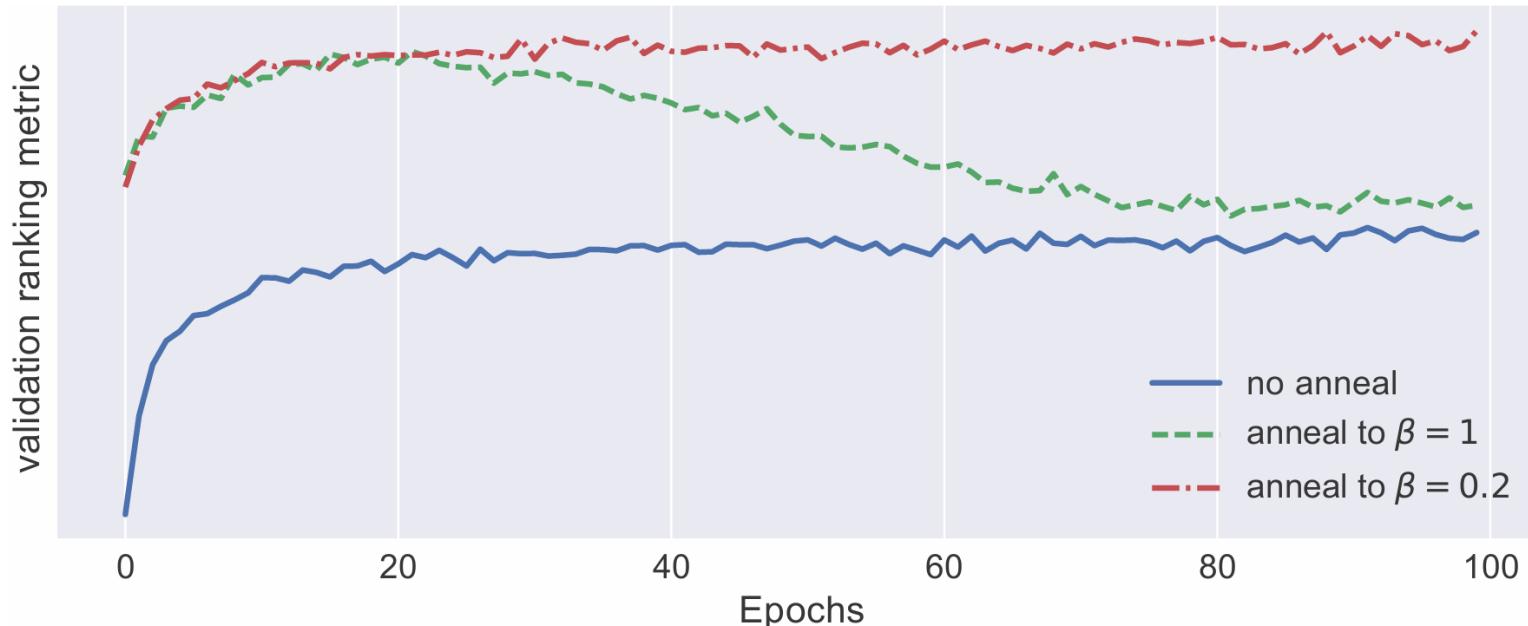
Mult-VAE / ELBO Variant

■ Annealing β

- Increase β from 0 to 1 until validation metric decreases \rightarrow fix β afterwards (**annealing**)
- Inherently sub-optimal scheduling, but no additional tuning required

■ Analysis on Annealing β

- Model focusses on reconstruction at initial stage of learning
- When reconstruction is hindered by regularization (e.g., epoch 20), fix β



Mult-VAE / General View of AE

■ Autoencoders (AEs)

- Let's look through other types of AEs
- Maximum-likelihood estimation of AE is as follows

$$\theta^{\text{AE}}, \phi^{\text{AE}} = \arg \max_{\theta, \phi} \sum_u \mathbb{E}_{\delta(\mathbf{z}_u - g_\phi(\mathbf{x}_u))} [\log p_\theta(\mathbf{x}_u | \mathbf{z}_u)]$$

Unit pulse (δ) has value
only when $\mathbf{z}_u = g_\phi(\mathbf{x}_u)$

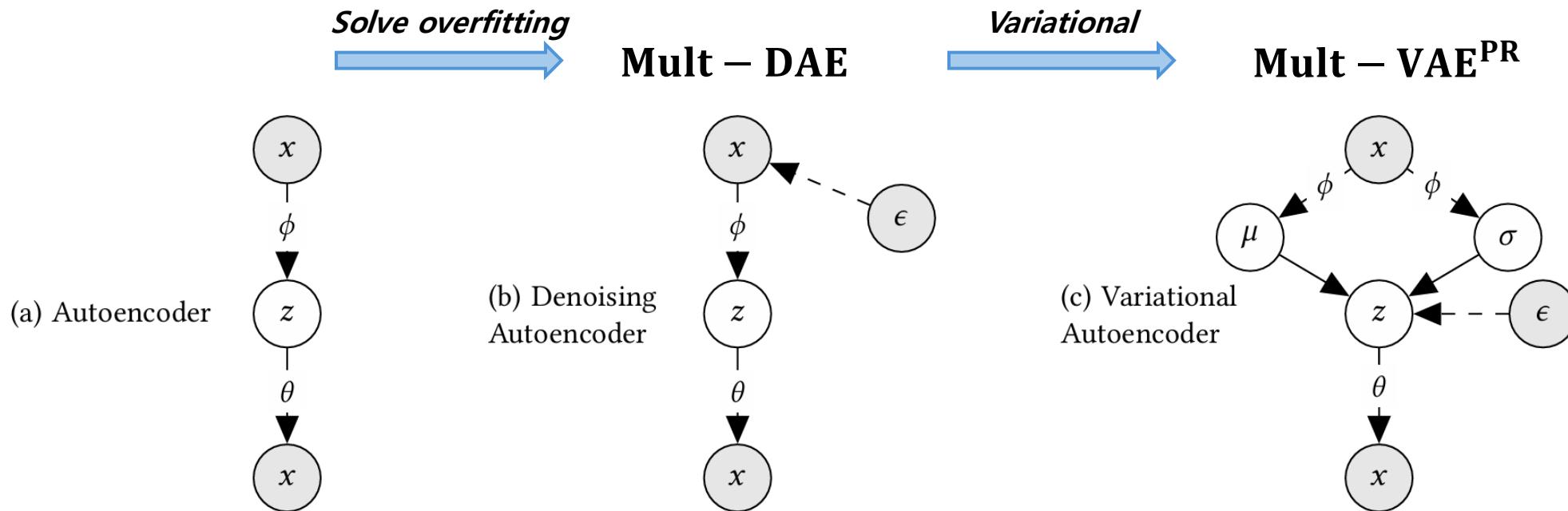
$$= \arg \max_{\theta, \phi} \sum_u \underline{\log p_\theta(\mathbf{x}_u | g_\phi(\mathbf{x}_u))}$$

No regularization (KLD) since
AEs project single point to latent space

Mult-VAE / General View of AE

■ AE Variants

- AEs are prone to overfit : put all the probability mass to non-zero entries in \mathbf{x}_u
- Mixing AE with dropout, denoising AE (DAE) achieves better results



Mult-VAE / Experiments

■ Prediction

- Input : user click history / Output : multinomial probability
- Advantage of AEs : recommendation possible for users **not seen in training data**

■ Experiment Setup

- # of interactions : density of user-item click matrix
- # of held-out users : number of users on in training data

	ML-20M	Netflix	MSD
# of users	136,677	463,435	571,355
# of items	20,108	17,769	41,140
# of interactions	10.0M	56.9M	33.6M
% of interactions	0.36%	0.69%	0.14%
# of held-out users	10,000	40,000	50,000

Mult-VAE / Experiments

■ Comparison with Baselines

NCF (=NeuMF) only compatible in ML-1M, Pinterest dataset

(a) ML-20M			(b) Netflix			(c) MSD			
	Recall@20	Recall@50	Recall@20	Recall@50	NDCG@100		Recall@20	Recall@50	NDCG@100
Mult-VAE ^{PR}	0.395	0.537	0.426	Mult-VAE ^{PR}	0.351	0.444	0.386	Mult-VAE ^{PR}	0.266
Mult-DAE	0.387	0.524	0.419	Mult-DAE	0.344	0.438	0.380	Mult-DAE	0.266
WMF	0.360	0.498	0.386	WMF	0.316	0.404	0.351	WMF	0.211
SLIM	0.370	0.495	0.401	SLIM	0.347	0.428	0.379	SLIM	—
CDAE	0.391	0.523	0.418	CDAE	0.343	0.428	0.376	CDAE	0.188

(a) ML-1M

	NCF	NCF (pre-train)	Mult-DAE
Recall@10	0.705	0.730	0.722
NDCG@10	0.426	0.447	0.446

(b) Pinterest

	NCF	NCF (pre-train)	Mult-DAE
Recall@10	0.872	0.880	0.886
NDCG@10	0.551	0.558	0.580

Mult-VAE / Experiments

■ Reconstruction Losses

$$\log p_{\theta}(\mathbf{x}_u | \mathbf{z}_u) \stackrel{\text{Mult}}{=} \sum_i x_{ui} \log \pi_i(\mathbf{z}_u)$$

$$\log p_{\theta}(\mathbf{x}_u | \mathbf{z}_u) \stackrel{\text{Gaussian}}{=} - \sum_i \frac{c_{ui}}{2} (x_{ui} - f_{ui})^2$$

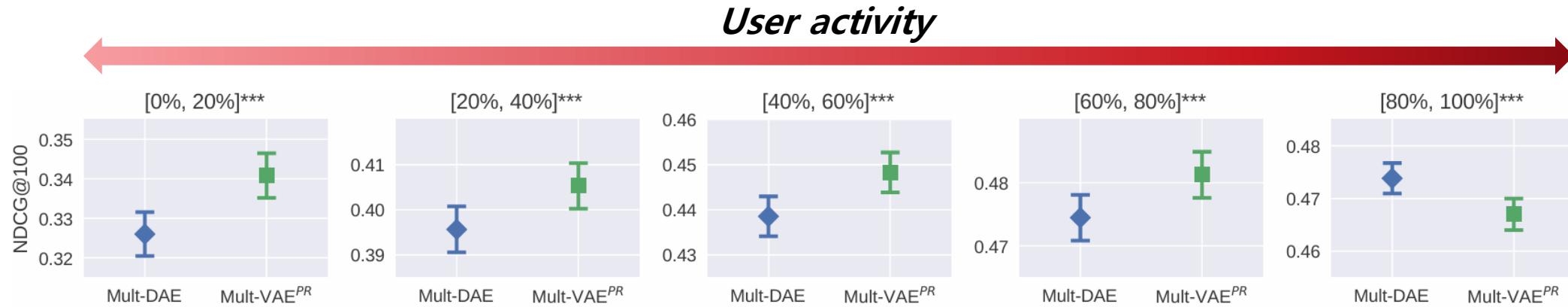
$$\log p_{\theta}(\mathbf{x}_u | \mathbf{z}_u) \stackrel{\text{Logistic}}{=} \sum_i x_{ui} \log \sigma(f_{ui}) + (1 - x_{ui}) \log(1 - \sigma(f_{ui}))$$

ML-20M

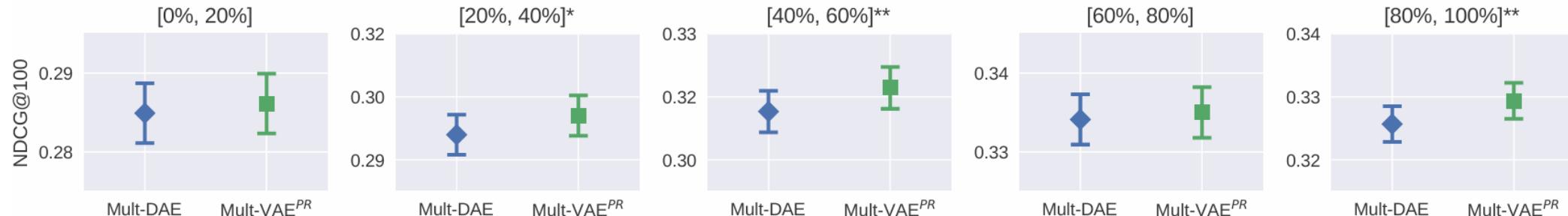
	Recall@20	Recall@50	NDCG@100
Mult-VAE ^{PR}	0.395	0.537	0.426
Gaussian-VAE ^{PR}	0.383	0.523	0.415
Logistic-VAE ^{PR}	0.388	0.523	0.419
Mult-DAE	0.387	0.524	0.419
Gaussian-DAE	0.376	0.515	0.409
Logistic-DAE	0.381	0.516	0.414

Mult-VAE / Experiments

■ DAE vs VAE



(a) ML-20M



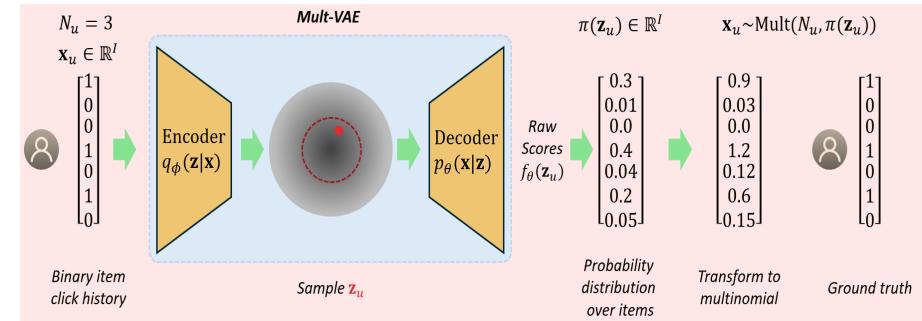
(b) MSD

#*: amount of statistical significance

Mult-VAE / Conclusion

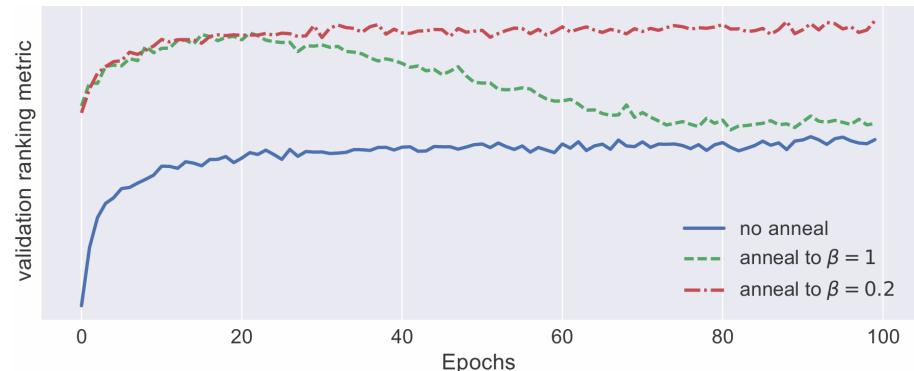
■ VAE for CF

- Incorporated generative models to CF
- Performed promising results based on multinomial view



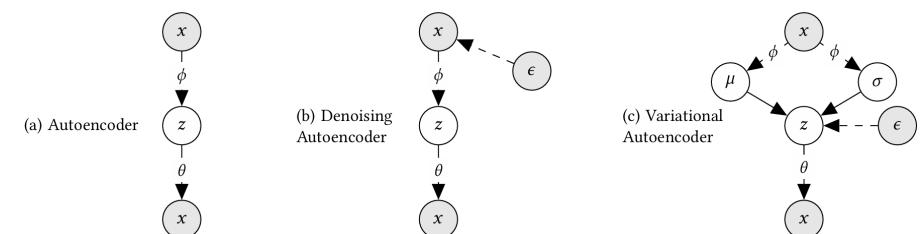
■ Information Theory Connections

- Provided re-interpretation of ELBO in this literature
- Showed connection to other domains



■ Investigation of Aes

- Identified pros and cons of both Mult-VAE^{PR} and Mult-DAE
- Showed robustness of principled Bayesian approach



□ GCN for CF

□ Too Heavy!

□ Architecture

□ Analysis

□ Experiments

□ Conclusion

LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation

Xiangnan He

University of Science and Technology
of China
xiangnanhe@gmail.com

Kuan Deng

University of Science and Technology
of China
dengkuan@mail.ustc.edu.cn

Xiang Wang

National University of Singapore
xiangwang@u.nus.edu

Yan Li

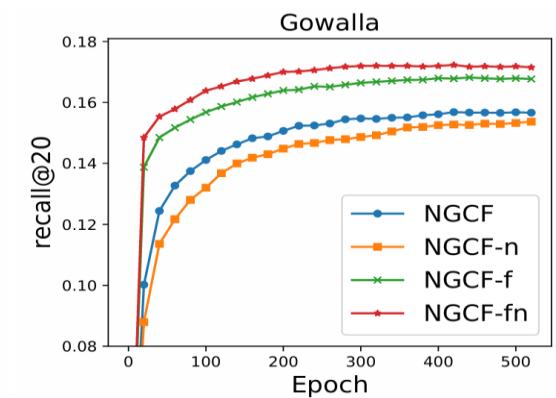
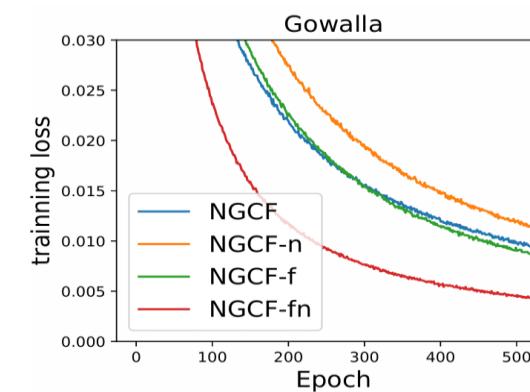
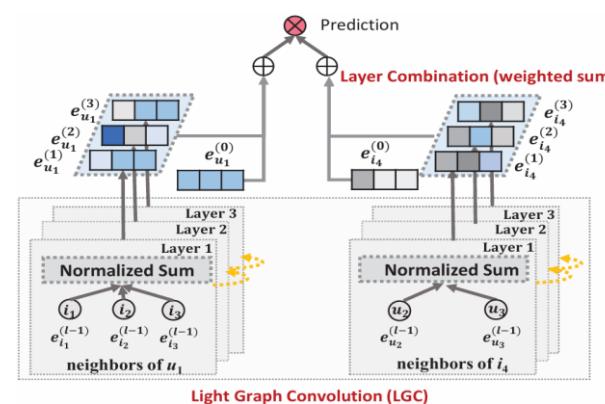
Beijing Kuaishou Technology
Co., Ltd.
liyan@kuaishou.com

Yongdong Zhang

University of Science and Technology
of China
zhyd73@ustc.edu.cn

Meng Wang*

Hefei University of Technology
eric.mengwang@gmail.com

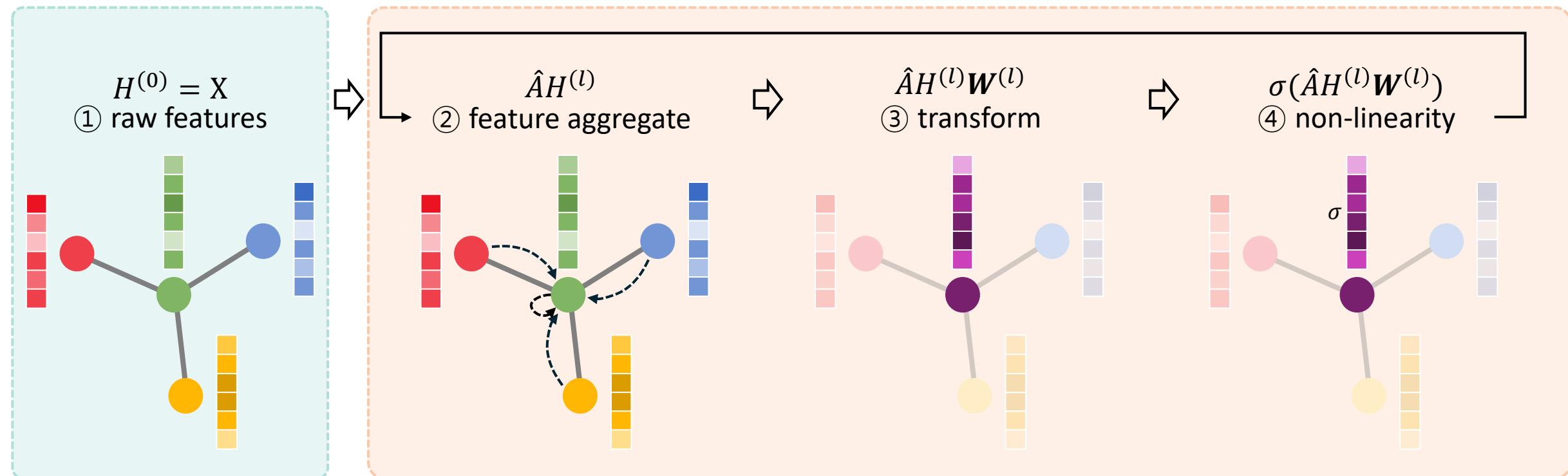


LightGCN / GCN for CF

■ Graph Convolution Network (GCN)

- GCN : rising star in many graph representation learning tasks
- Node-wise learning view of GCN is as follows

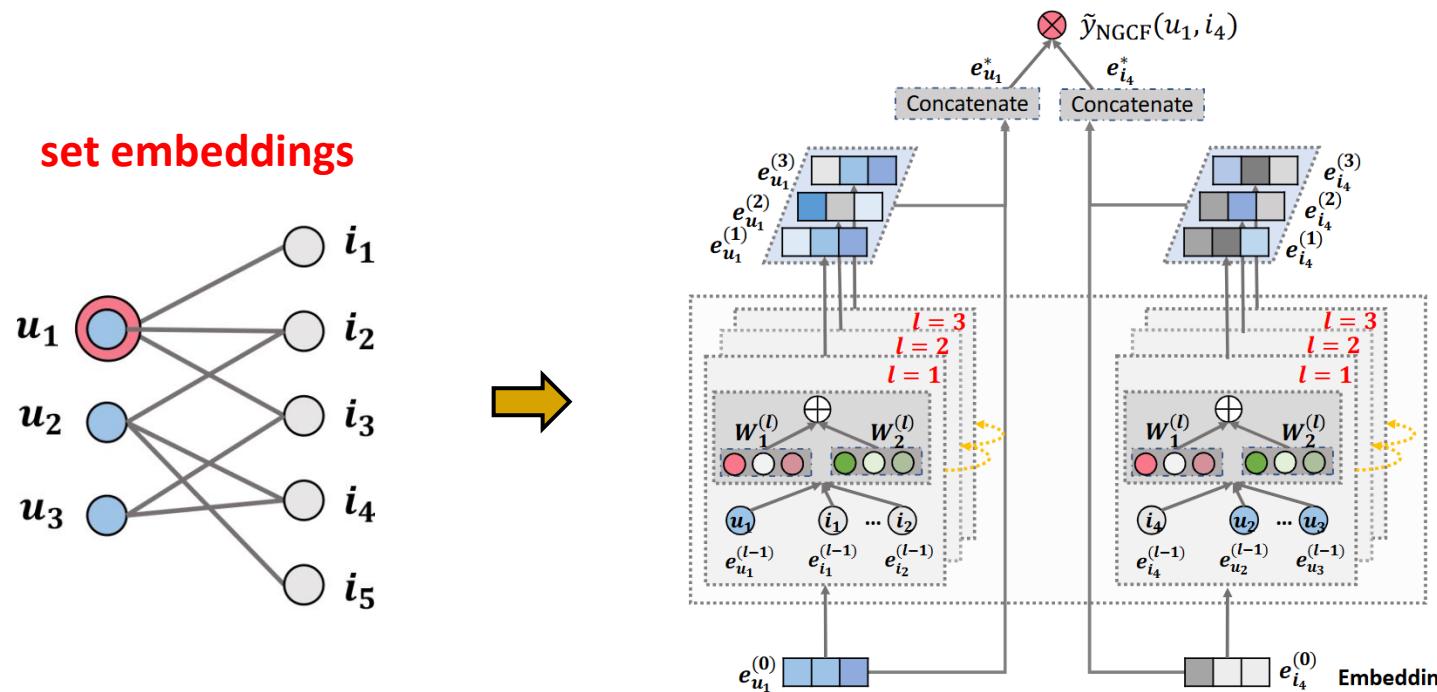
$$H^{(l+1)} = \sigma(\hat{A}H^{(l)}W^{(l)})$$



LightGCN / GCN for CF

■ Inspired by GCN, Creation of NGCF

- Incorporated every GCN architecture to CF
- Only difference : no initial raw features for users and items



LightGCN / GCN for CF

■ Posing Doubt on NGCF Architecture

- Environment between node classification(GCN) and CF(NGCF) is different
- This should lead in architecture change, rather than naively copying GCN
- Authors point out 2 problematic components of NGCF → $\mathbf{W}_1, \mathbf{W}_2$ and $\sigma(\cdot)$

Embedding update formulas (NGCF)

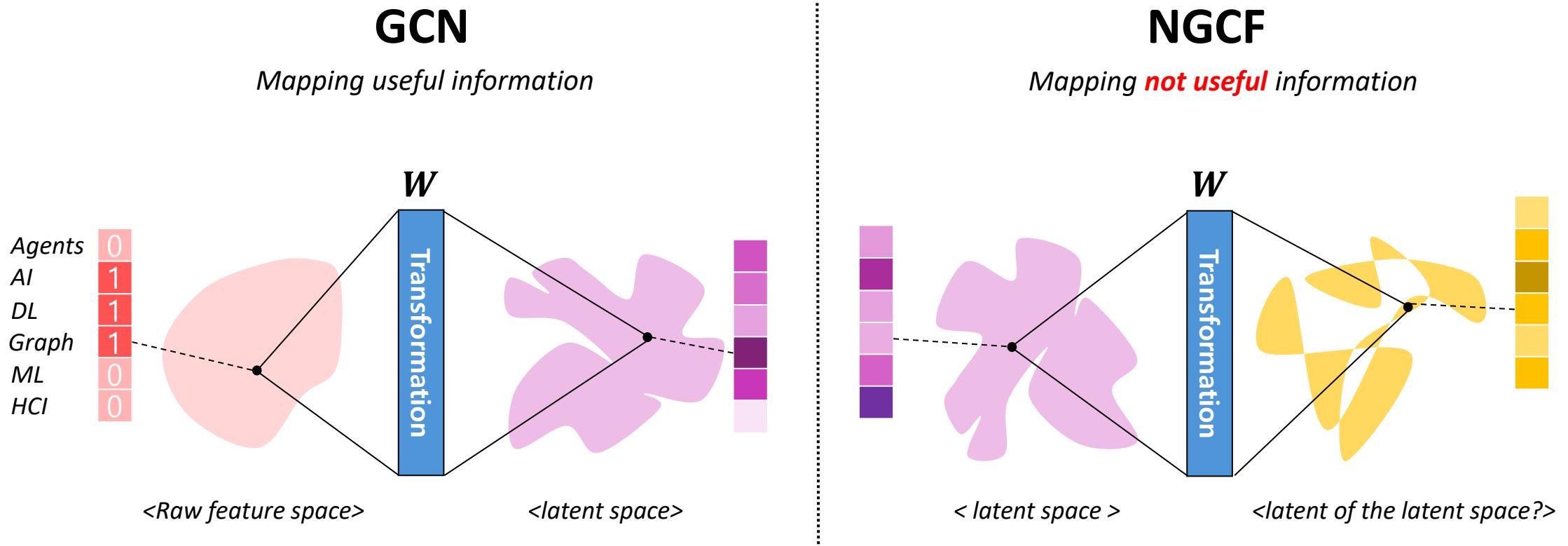
$$\mathbf{e}_u^{(k+1)} = \sigma \left(\mathbf{W}_1 \mathbf{e}_u^{(k)} + \sum_{i \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u| |\mathcal{N}_i|}} (\mathbf{W}_1 \mathbf{e}_i^{(k)} + \mathbf{W}_2 (\mathbf{e}_i^{(k)} \odot \mathbf{e}_u^{(k)})) \right)$$

$$\mathbf{e}_i^{(k+1)} = \sigma \left(\mathbf{W}_1 \mathbf{e}_i^{(k)} + \sum_{u \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_u| |\mathcal{N}_i|}} (\mathbf{W}_1 \mathbf{e}_u^{(k)} + \mathbf{W}_2 (\mathbf{e}_u^{(k)} \odot \mathbf{e}_i^{(k)})) \right)$$

LightGCN / Too Heavy!

■ Do We Even Need W ?

- When do we need feature transformation?
- Transformation allows rich attributes to be mapped into meaningful latent space

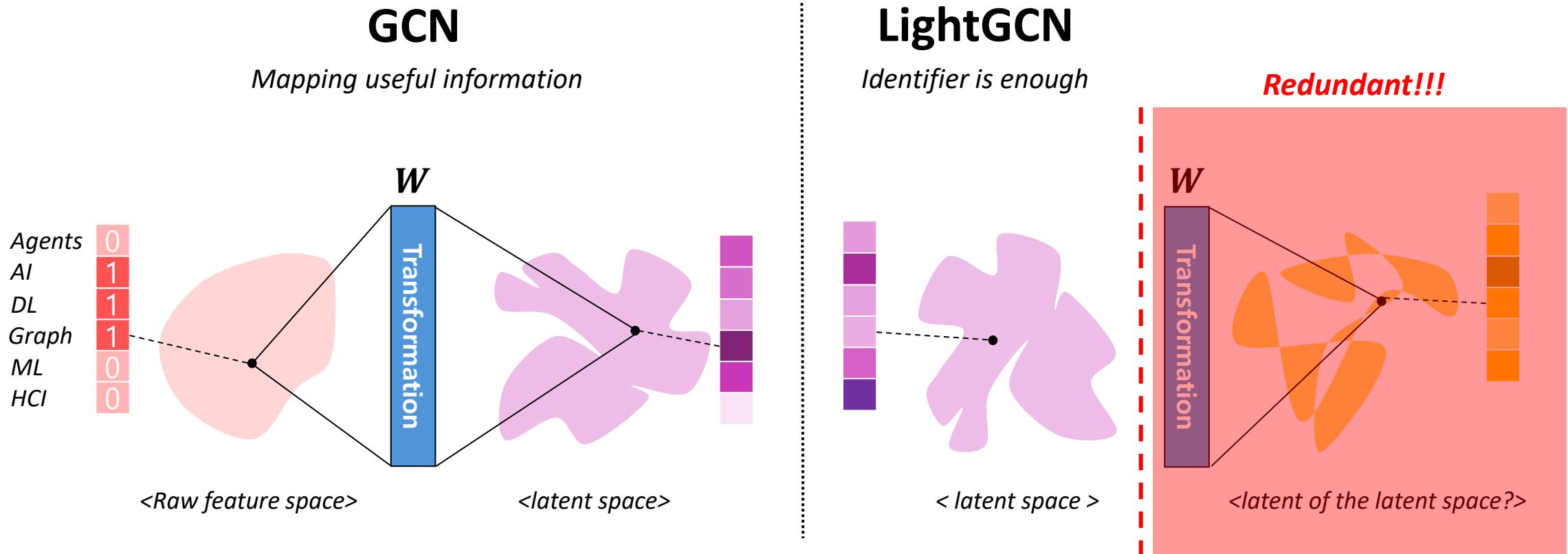


* The feature scenario of GCN is based on CiteSeer dataset

LightGCN / Too Heavy!

■ No Need for Feature Transformation W

- Node classification (GCN) : node has rich attributes
- Collaborative filtering (NGCF) : node ID (only attribute) is nothing but an identifier



* The feature scenario of GCN is based on CiteSeer dataset

LightGCN / Too Heavy!

■ Hypothesis on W and $\sigma(\cdot)$

- e_u and e_i serves as identity and latent traits → **no need for feature transformation**
- No need for feature transformation → **no need for nonlinear activation**

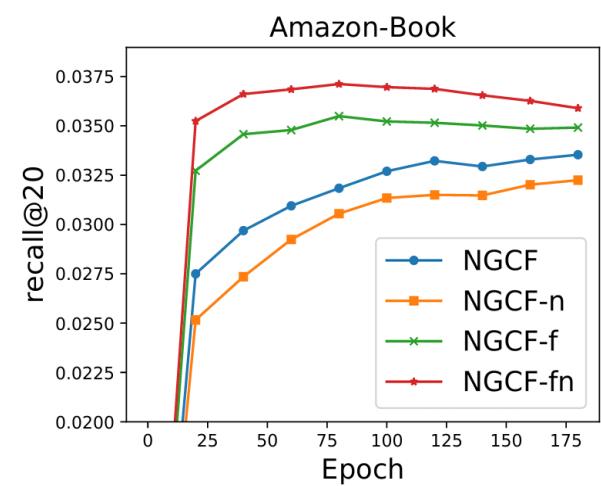
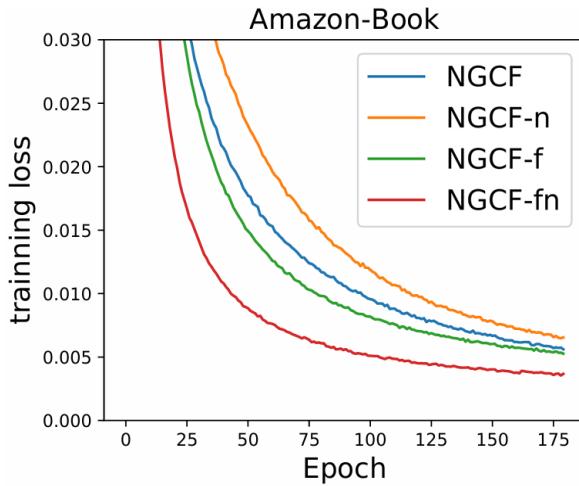
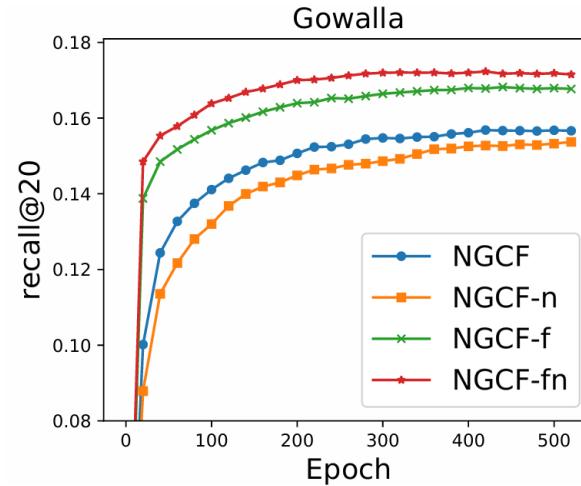
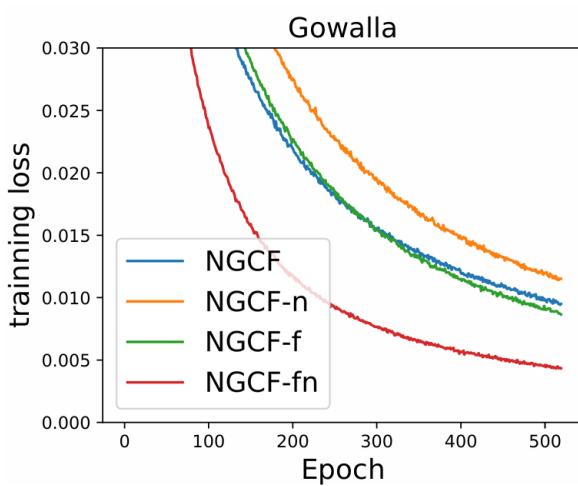
$$\mathbf{e}_u^{(k+1)} = \cancel{\sigma} \left(\cancel{W_1} \mathbf{e}_u^{(k)} + \sum_{i \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u| |\mathcal{N}_i|}} (\cancel{W_1} \mathbf{e}_i^{(k)} + \cancel{W_2} (\mathbf{e}_i^{(k)} \odot \mathbf{e}_u^{(k)})) \right)$$

$$\mathbf{e}_i^{(k+1)} = \cancel{\sigma} \left(\cancel{W_1} \mathbf{e}_i^{(k)} + \sum_{u \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_u| |\mathcal{N}_i|}} (\cancel{W_1} \mathbf{e}_u^{(k)} + \cancel{W_2} (\mathbf{e}_u^{(k)} \odot \mathbf{e}_i^{(k)})) \right)$$

LightGCN / Too Heavy!

■ Ablation : NGCF with 3 Variants

□ Don't increase the training difficulty!!!



NGCF, original

NGCF-f, remove $\mathbf{W}_1, \mathbf{W}_2$

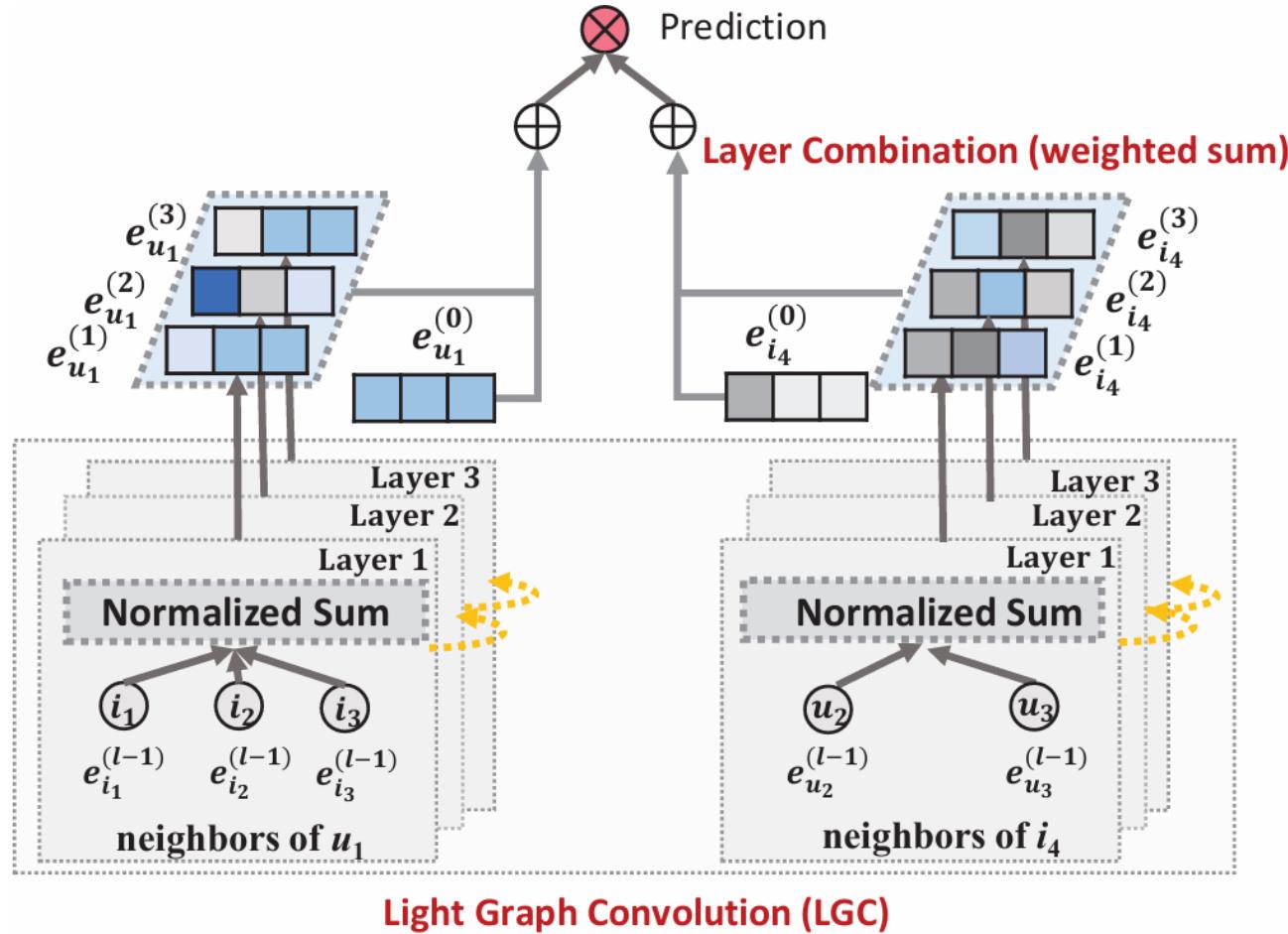
NGCF-n, remove σ

NGCF-fn, remove $\mathbf{W}_1, \mathbf{W}_2$ and σ

→ improved result : the absence of meaningless transformation
→ degraded result : linearity of transformations
→ improved result : **removing every meaningless counterparts**

LightGCN / Architecture

■ Overview of LightGCN



LightGCN / Architecture

■ Light Graph Convolution (LGC)

- Aggregation module of LightGCN
- Symmetric normalization follows standard GCN
- Other options are possible, later experimented

$$\left. \begin{aligned} \mathbf{e}_u^{(k+1)} &= \sum_{i \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|} \sqrt{|\mathcal{N}_i|}} \mathbf{e}_i^{(k)}, \\ \mathbf{e}_i^{(k+1)} &= \sum_{u \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i|} \sqrt{|\mathcal{N}_u|}} \mathbf{e}_u^{(k)}. \end{aligned} \right\}$$

Controls the influence of super nodes

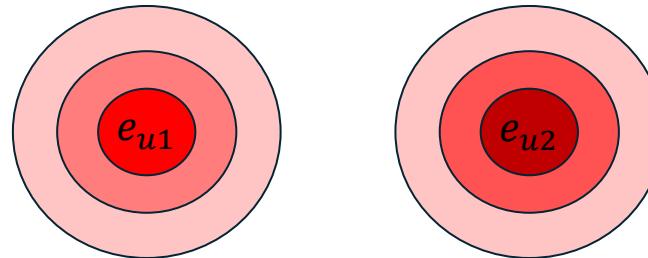
No self-connection?

LightGCN / Architecture

■ Layer Combination

- Final node representation = weighted (α_k) sum over **all layer (hop) representations**
- Weights uniformly set to $1/(K + 1)$ to maintain simplicity

$$\mathbf{e}_u = \sum_{k=0}^K \alpha_k \mathbf{e}_u^{(k)}; \quad \mathbf{e}_i = \sum_{k=0}^K \alpha_k \mathbf{e}_i^{(k)}$$



■ Prediction

- Dot product, used for ranking score

$$\hat{y}_{ui} = \mathbf{e}_u^T \mathbf{e}_i$$

LightGCN / Analysis

■ Is the Simple Design okay?

- Authors investigate the rationality behind LightGCN
- **1) Connections with other GCN variants**
- **2) Second-order embedding smoothness**

$$\mathbf{R} \in \mathbb{R}^{M \times N} \quad \mathbf{E}^{(k+1)} = (\mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}) \mathbf{E}^{(k)}$$
$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{R} \\ \mathbf{R}^T & \mathbf{0} \end{pmatrix} \quad \Rightarrow \quad \mathbf{E} = \alpha_0 \mathbf{E}^{(0)} + \alpha_1 \mathbf{E}^{(1)} + \alpha_2 \mathbf{E}^{(2)} + \dots + \alpha_K \mathbf{E}^{(K)}$$
$$= \alpha_0 \mathbf{E}^{(0)} + \alpha_1 \tilde{\mathbf{A}} \mathbf{E}^{(0)} + \alpha_2 \tilde{\mathbf{A}}^2 \mathbf{E}^{(0)} + \dots + \alpha_K \tilde{\mathbf{A}}^K \mathbf{E}^{(0)}$$

LightGCN / Analysis

■ Connections with Other GCN Variants (SGCN)

- Simplified GCN : removed nonlinearities + collapse weight metrices into one

SGCN
$$\left\{ \begin{array}{l} \mathbf{E}^{(k+1)} = (\mathbf{D} + \mathbf{I})^{-\frac{1}{2}} \boxed{(\mathbf{A} + \mathbf{I})} (\mathbf{D} + \mathbf{I})^{-\frac{1}{2}} \mathbf{E}^{(k)} \\ \quad \text{Self-connection} \\ \mathbf{E}^{(K)} = (\mathbf{A} + \mathbf{I}) \mathbf{E}^{(K-1)} = (\mathbf{A} + \mathbf{I})^K \mathbf{E}^{(0)} \\ \quad = \binom{K}{0} \mathbf{E}^{(0)} + \binom{K}{1} \mathbf{A} \mathbf{E}^{(0)} + \binom{K}{2} \mathbf{A}^2 \mathbf{E}^{(0)} + \dots + \binom{K}{K} \mathbf{A}^K \mathbf{E}^{(0)} \end{array} \right.$$

$(\mathbf{D} + \mathbf{I})^{-\frac{1}{2}}$ only rescales

LightGCN
$$\left\{ \mathbf{E} = \underline{\alpha_0 \mathbf{E}^{(0)} + \alpha_1 \tilde{\mathbf{A}} \mathbf{E}^{(0)} + \alpha_2 \tilde{\mathbf{A}}^2 \mathbf{E}^{(0)} + \dots + \alpha_K \tilde{\mathbf{A}}^K \mathbf{E}^{(0)}}$$

Essentially equivalent!

LightGCN subsumes self-connection!

The weight matrix in SGCN can be absorbed into the 0-th layer embedding parameters, thus it is omitted in the analysis.

LightGCN / Analysis

■ Connections with Other GCN Variants (APPNP)

□ APPNP : restart design into GCN to alleviate oversmoothing

□ APPNP adds self-connection (already discussed)

Restart prob

$$\text{APPNP} \left\{ \begin{array}{l} \mathbf{E}^{(k+1)} = \beta \mathbf{E}^{(0)} + (1 - \beta) \tilde{\mathbf{A}} \mathbf{E}^{(k)} \\ \mathbf{E}^{(K)} = \beta \mathbf{E}^{(0)} + (1 - \beta) \tilde{\mathbf{A}} \mathbf{E}^{(K-1)}, \\ \quad = \beta \mathbf{E}^{(0)} + \beta(1 - \beta) \tilde{\mathbf{A}} \mathbf{E}^{(0)} + (1 - \beta)^2 \tilde{\mathbf{A}}^2 \mathbf{E}^{(K-2)} \\ \quad = \beta \mathbf{E}^{(0)} + \beta(1 - \beta) \tilde{\mathbf{A}} \mathbf{E}^{(0)} + \beta(1 - \beta)^2 \tilde{\mathbf{A}}^2 \mathbf{E}^{(0)} + \dots + (1 - \beta)^K \tilde{\mathbf{A}}^K \mathbf{E}^{(0)}. \end{array} \right.$$

Essentially equivalent!

$$\text{LightGCN} \left\{ \mathbf{E} = \alpha_0 \mathbf{E}^{(0)} + \alpha_1 \tilde{\mathbf{A}} \mathbf{E}^{(0)} + \alpha_2 \tilde{\mathbf{A}}^2 \mathbf{E}^{(0)} + \dots + \alpha_K \tilde{\mathbf{A}}^K \mathbf{E}^{(0)} \right.$$

LightGCN can allow long-range modeling

LightGCN / Analysis

■ Second-Order Embedding Smoothness

□ How much smoothening happen between two users (u, v)?

□ Smoothness strength($c_{v \rightarrow u}$) : $\mathbf{e}_u^{(2)} = \sum_v c_{v \rightarrow u} \mathbf{e}_v^{(0)}$

$$\mathbf{e}_u^{(2)} = \sum_{i \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|} \sqrt{|\mathcal{N}_i|}} \mathbf{e}_i^{(1)} = \sum_{i \in \mathcal{N}_u} \frac{1}{|\mathcal{N}_i|} \sum_{v \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_u|} \sqrt{|\mathcal{N}_v|}} \mathbf{e}_v^{(0)}$$

$$c_{v \rightarrow u} = \frac{1}{\sqrt{|\mathcal{N}_u|} \sqrt{|\mathcal{N}_v|}} \sum_{i \in \mathcal{N}_u \cap \mathcal{N}_v} \frac{1}{|\mathcal{N}_i|} \quad \Rightarrow \quad \textit{Explains well on CF measuring user (item) similarity}$$

LightGCN / Experiments

■ Model Training

- Bayesian Personalized Ranking loss (pairwise)

$$L_{BPR} = - \sum_{u=1}^M \sum_{i \in \mathcal{N}_u} \sum_{j \notin \mathcal{N}_u} \ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda \|\mathbf{E}^{(0)}\|^2$$

■ Datasets

Table 2: Statistics of the experimented data.

Dataset	User #	Item #	Interaction #	Density
Gowalla	29,858	40,981	1,027,370	0.00084
Yelp2018	31,668	38,048	1,561,406	0.00130
Amazon-Book	52,643	91,599	2,984,108	0.00062

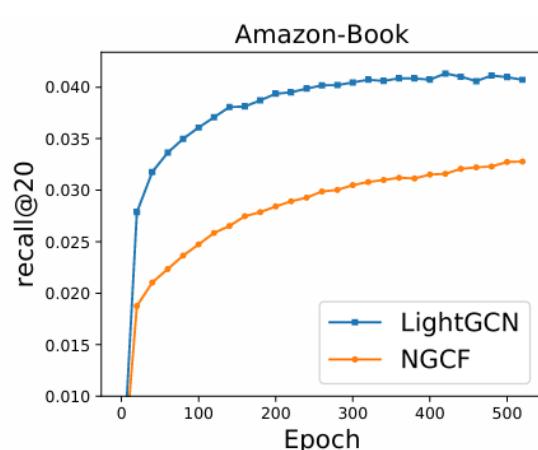
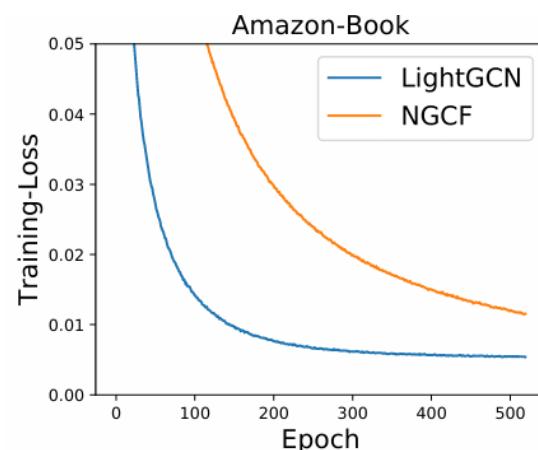
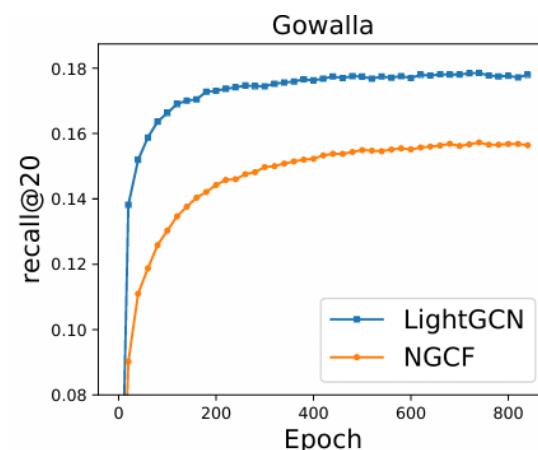
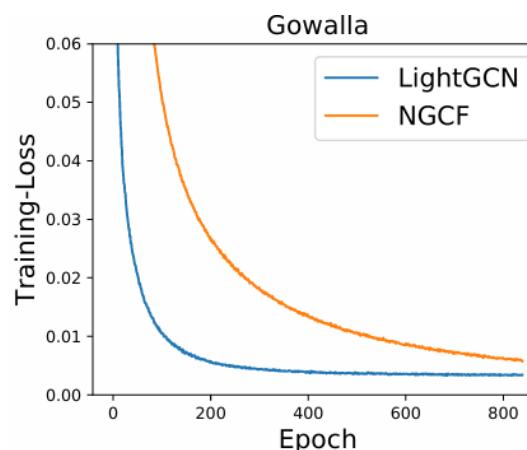
■ Metrics

- Recall@20, NDCG@20

LightGCN / Experiments

■ NGCF vs LightGCN with Different Layers

Dataset		Gowalla		Yelp2018		Amazon-Book	
Layer #	Method	recall	ndcg	recall	ndcg	recall	ndcg
1 Layer	NGCF	0.1556	0.1315	0.0543	0.0442	0.0313	0.0241
	LightGCN	0.1755(+12.79%)	0.1492(+13.46%)	0.0631(+16.20%)	0.0515(+16.51%)	0.0384(+22.68%)	0.0298(+23.65%)
2 Layers	NGCF	0.1547	0.1307	0.0566	0.0465	0.0330	0.0254
	LightGCN	0.1777(+14.84%)	0.1524(+16.60%)	0.0622(+9.89%)	0.0504(+8.38%)	0.0411(+24.54%)	0.0315(+24.02%)
3 Layers	NGCF	0.1569	0.1327	0.0579	0.0477	0.0337	0.0261
	LightGCN	0.1823(+16.19%)	0.1555(+17.18%)	0.0639(+10.38%)	0.0525(+10.06%)	0.0410(+21.66%)	0.0318(+21.84%)
4 Layers	NGCF	0.1570	0.1327	0.0566	0.0461	0.0344	0.0263
	LightGCN	0.1830(+16.56%)	0.1550(+16.80%)	0.0649(+14.58%)	0.530(+15.02%)	0.0406(+17.92%)	0.0313(+18.92%)



LightGCN / Experiments

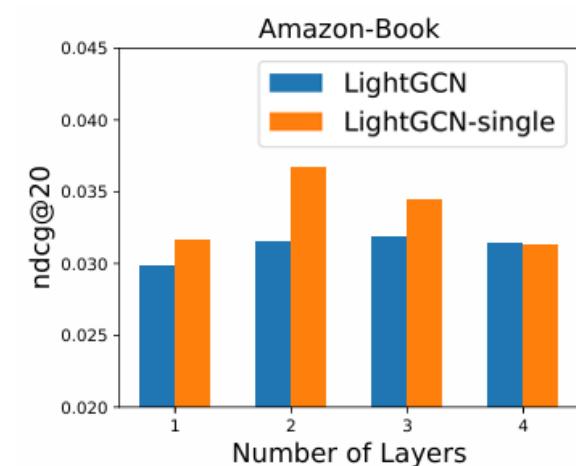
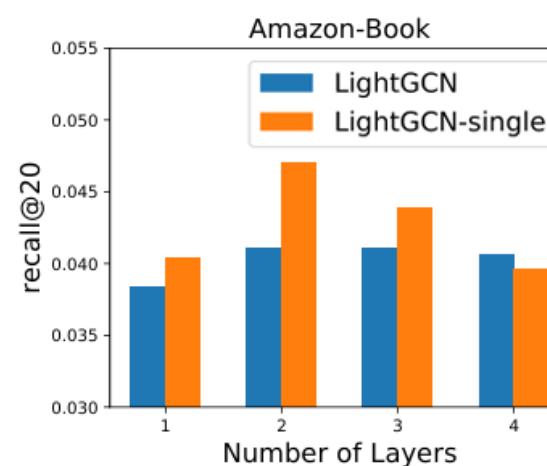
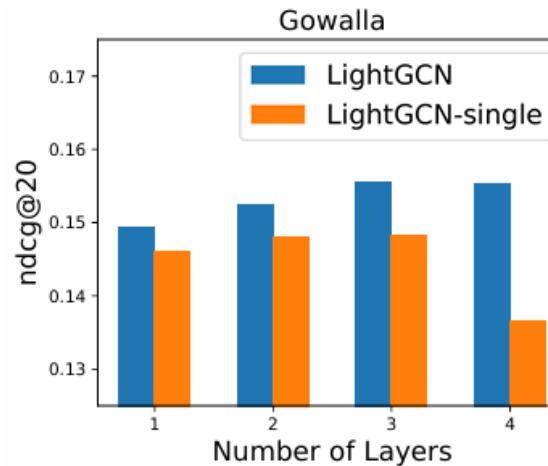
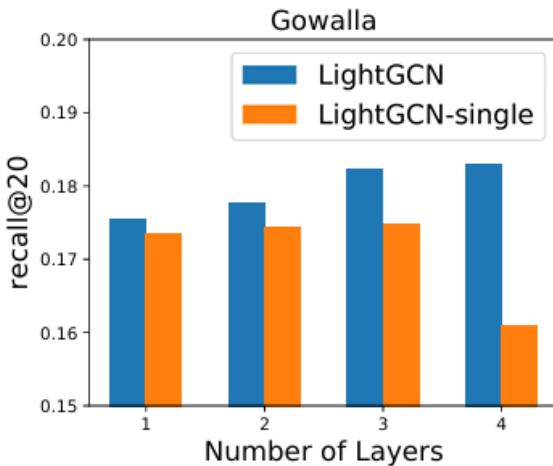
■ LightGCN vs Different Models

Dataset	Gowalla		Yelp2018		Amazon-Book	
Method	recall	ndcg	recall	ndcg	recall	ndcg
NGCF	0.1570	0.1327	0.0579	0.0477	0.0344	0.0263
Mult-VAE	0.1641	0.1335	0.0584	0.0450	0.0407	0.0315
GRMF	0.1477	0.1205	0.0571	0.0462	0.0354	0.0270
GRMF-norm	0.1557	0.1261	0.0561	0.0454	0.0352	0.0269
LightGCN	0.1830	0.1554	0.0649	0.0530	0.0411	0.0315

LightGCN / Experiments

■ Ablation : Layer Combination

- LightGCN-single : only use the last embedding $\mathbf{E}^{(K)}$ for prediction
- LightGCN tackles oversmoothing



LightGCN / Experiments

■ Ablation : Others

□ -L : target node's coefficient / -R : neighbor node's coefficient

Table 5: Performance of the 3-layer LightGCN with different choices of normalization schemes in graph convolution.

Dataset	Gowalla		Yelp2018		Amazon-Book	
Method	recall	ndcg	recall	ndcg	recall	ndcg
LightGCN-L-L	0.1724	0.1414	0.0630	0.0511	0.0419	0.0320
LightGCN-L-R	0.1578	0.1348	0.0587	0.0477	0.0334	0.0259
LightGCN-L ₁	0.159	0.1319	0.0573	0.0465	0.0361	0.0275
LightGCN-L	0.1589	0.1317	0.0619	0.0509	0.0383	0.0299
LightGCN-R	0.1420	0.1156	0.0521	0.0401	0.0252	0.0196
LightGCN	0.1830	0.1554	0.0649	0.0530	0.0411	0.0315

Method notation: -L means only the left-side norm is used, -R means only the right-side norm is used, and -L₁ means the L₁ norm is used.

$$S_U = \sum_{u=1}^M \sum_{v=1}^M c_{v \rightarrow u} \left(\frac{\mathbf{e}_u}{\|\mathbf{e}_u\|^2} - \frac{\mathbf{e}_v}{\|\mathbf{e}_v\|^2} \right)^2$$

Table 6: Smoothness loss of the embeddings learned by LightGCN and MF (the lower the smoother).

Dataset	Gowalla	Yelp2018	Amazon-book
	Smoothness of User Embeddings		
MF	15449.3	16258.2	38034.2
LightGCN-single	12872.7	10091.7	32191.1
Smoothness of Item Embeddings			
MF	12106.7	16632.1	28307.9
LightGCN-single	5829.0	6459.8	16866.0

LightGCN / Experiments

■ Hyper-Parameter Sensitivity

$$L_{BPR} = - \sum_{u=1}^M \sum_{i \in \mathcal{N}_u} \sum_{j \notin \mathcal{N}_u} \ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda \|\mathbf{E}^{(0)}\|^2$$

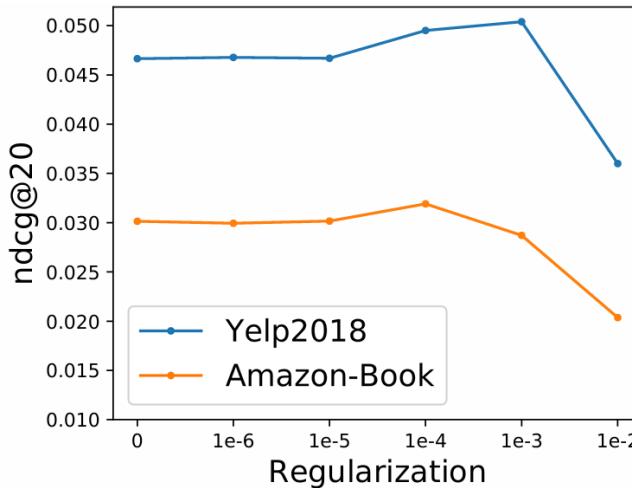
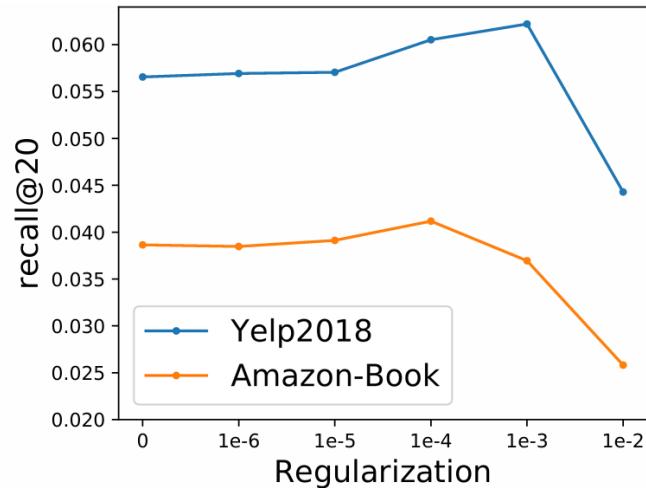


Figure 5: Performance of 2-layer LightGCN w.r.t. different regularization coefficient λ on Yelp and Amazon-Book.

LightGCN / Conclusion

■ LightGCN

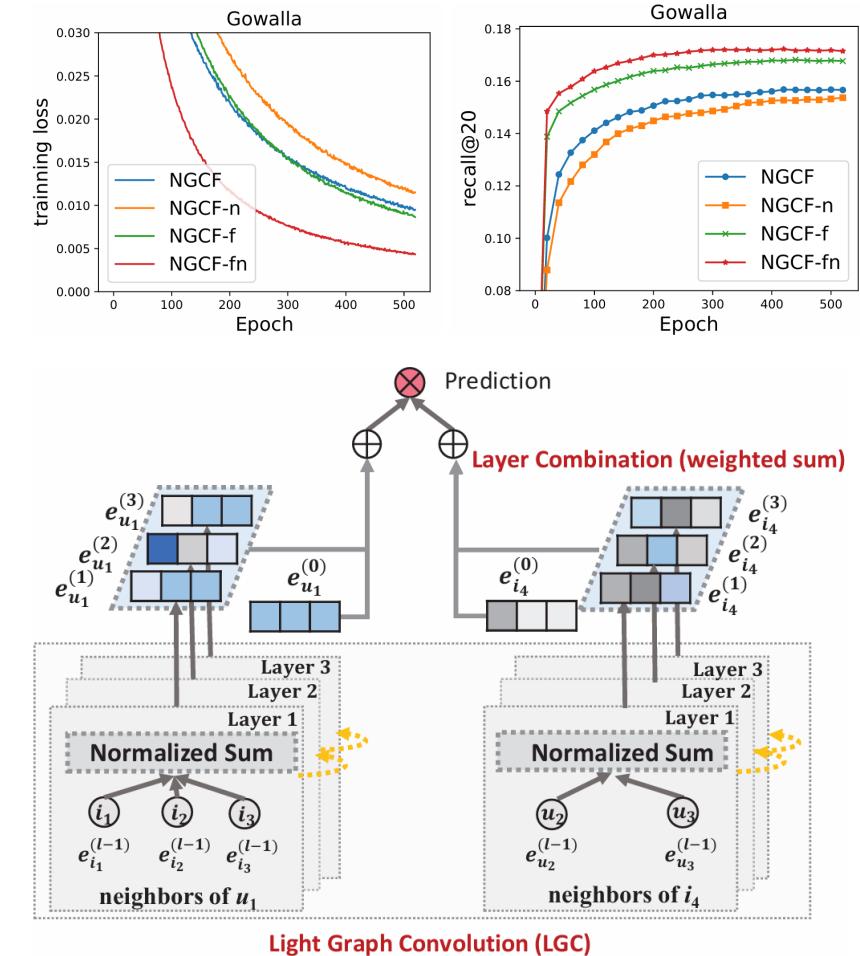
- Motivated by the CF environment, simplified NGCF
- Proved that complicated designs are unnecessary for CF

■ Advantages of Simplicity

- Simple structure allows for more interpretability
- Proved LightGCN subsumes the effect of self-connection
- Proved LightGCN has rational smoothing mechanism

■ Lessons

- Ablations critical for understanding the impact of each operation
- Performance doesn't stem from complexity



**Thank you
Q&A**