

Recommender Systems and DGL

Quan Gan

AWS Shanghai AI Lab

July 12, 2019

Why Recommendation

Why Recommendation

We don't want our customers to think (hard).

Why Recommendation

Good relevant recommendations make the customers adhere to us.

**Added to Cart**
Only 1 left in stock.

Cart subtotal (1 item): \$523.01
Your order qualifies for **FREE Shipping!** Select this option at checkout. [Details](#)


Cart


Proceed to checkout (1 item)

Your recently viewed items and featured recommendations

Related to items you've viewed

Page 1 of 9






Dell OptiPlex 5050 Small Form Factor Desktop, Intel Core i5-6500, 8GB DDR4 RAM, 256GB SSD,...

★★★★☆ 1


\$493.98



Dell OptiPlex 9020 Small Form Factor Desktop, Intel Core i5-8500 3 GHz Hexa-Core, 8GB RAM,...

★★★★☆ 5


\$674.00



Dell OptiPlex 9020 Small Form Factor Desktop, Intel Core i7 4770, 16GB Ram,...

★★★★☆ 59


\$345.00



Dell CFC5C OptiPlex 3050 Micro Form Factor Desktop Computer, Intel Core i5-7500T, 8GB DDR4,...

★★★★☆ 38


13 offers from \$479.00



Acer Aspire TC-885-ACCFLU50 Desktop, 8th Gen Intel Core i5-8400, 8GB DDR4 + 16GB,...


★★★★☆ 59


\$499.99



Inspired by your browsing history

Page 1 of 9






Dell SE2419Hx 23.8" IPS Full HD (1920x1080) Monitor

★★★★☆ 34


\$129.99



Seagate IronWolf 8TB NAS Internal Hard Drive HDD - 3.5 inch SATA 6Gb/s 7200 RPM 256MB Cache for...

★★★★☆ 47


\$219.99



Silicon Power 256GB SSD 3D NAND A55 SLC Cache Performance Boost SATA III 2.5" 7mm (0.28")...

★★★★☆ 950


\$28.99



Ubiquiti Networks Unifi 802.11ac Dual-Radio PRO Access Point (UAP-AC-PRO-US)

★★★★☆ 1,385


\$134.26



AmazonBasics Wired Computer Keyboard and Mouse, 10-Pack

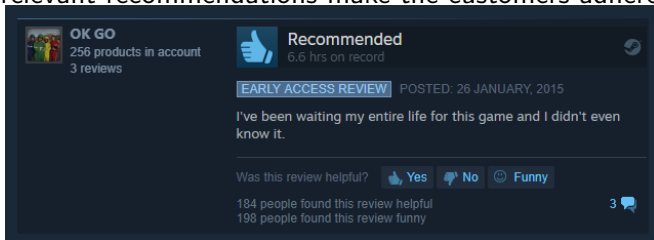
★★★★☆ 1,977

\$122.30



Why Recommendation

Good relevant recommendations make the customers adhere to us.



Recommender System: Problem Statement



Image source: Wikipedia

Collaborative Filtering



Collaborative Filtering



Collaborative Filtering





Image source: Wikipedia

Collaborative Filtering: Classical Methods

- **User-based:** Infer how a user i would act to an item j by looking at how users that have similar interactions to user i acted to item j .

Collaborative Filtering: Classical Methods

- **User-based:** Infer how a user i would act to an item j by looking at how users that have similar interactions to user i acted to item j .
 - We have millions of customers.

Collaborative Filtering: Classical Methods

- **User-based:** Infer how a user i would act to an item j by looking at how users that have similar interactions to user i acted to item j .
 - We have millions of customers.
 - User profiles change constantly and quickly, requiring frequent rebuilds (which are expensive already).


























				
				
				
				
				
				

Collaborative Filtering: Classical Methods

- **User-based:** Infer how a user i would act to an item j by looking at how users that have similar interactions to user i acted to item j .
 - We have millions of customers.
 - User profiles change constantly and quickly, requiring frequent rebuilds (which are expensive already).
 - Not interpretable (can't answer why a user prefers this).

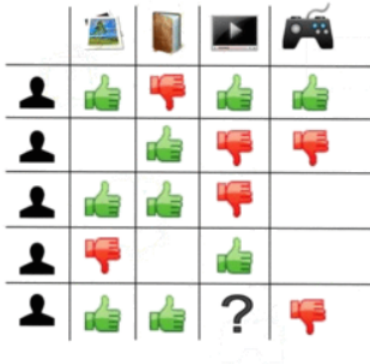
				
				
				
				
				
				

Machine Learning Kicks In

We were "representing" users and items with the items/users that had interactions with them.

Machine Learning Kicks In



	Feature 1	Feature 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

X

	Item 1	Item 2	Item 3	Item 4	Item 5
Feature 1	?	?	?	?	?
Feature 2	?	?	?	?	?

=

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0?	3	0?	3	0?
User 2	4	0?	0?	2	0?
User 3	0?	0?	3	0?	0?
User 4	3	0?	4	0?	3
User 5	4	3	0?	4	0?

We were "representing" users and items with the items/users that had interactions with them.

Source: [Kat Bailey](#)

Can we represent users and items as a set of features?

Latent User/Item Representations

- An item can be described with a set of features (e.g. how sweet some food is).

	Feature 1	Feature 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

X

	Item 1	Item 2	Item 3	Item 4	Item 5
Feature 1	?	?	?	?	?
Feature 2	?	?	?	?	?

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0.?	3	0.?	3	0.?
User 2	4	0.?	0.?	2	0.?
User 3	0.?	0.?	3	0.?	0.?
User 4	3	0.?	4	0.?	3
User 5	4	3	0.?	4	0.?

Source: [Kat Bailey](#)

Latent User/Item Representations

- An item can be described with a set of features (e.g. how sweet some food is).
- A user can be described with preferences of the same set of features (e.g. how much a user likes sweet food).

	Feature 1	Feature 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

	Item 1	Item 2	Item 3	Item 4	Item 5
Feature 1	?	?	?	?	?
Feature 2	?	?	?	?	?

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0?	3	0?	3	0?
User 2	4	0?	0?	2	0?
User 3	0?	0?	3	0?	0?
User 4	3	0?	4	0?	3
User 5	4	3	0?	4	0?

Source: [Kat Bailey](#)

Latent User/Item Representations

- An item can be described with a set of features (e.g. how sweet some food is).
- A user can be described with preferences of the same set of features (e.g. how much a user likes sweet food).
- The interaction is defined by how well the item features match the user preferences.

	Feature 1	Feature 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

X

	Item 1	Item 2	Item 3	Item 4	Item 5
Feature 1	?	?	?	?	?
Feature 2	?	?	?	?	?

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0.?	3	0.?	3	0.?
User 2	4	0.?	0.?	2	0.?
User 3	0.?	0.?	3	0.?	0.?
User 4	3	0.?	4	0.?	3
User 5	4	3	0.?	4	0.?

Source: [Kat Bailey](#)

Latent User/Item Representations

- An item can be described with a **vector** v_j (sweet, organic, etc.).
- A user can be described with preferences of the same set of features (e.g. how much a user likes sweet food).
- The interaction is defined by how well the item features match the user preferences.

	Feature 1	Feature 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

X

	Item 1	Item 2	Item 3	Item 4	Item 5
Feature 1	?	?	?	?	?
Feature 2	?	?	?	?	?

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0.?	3	0.?	3	0.?
User 2	4	0.?	0.?	2	0.?
User 3	0.?	0.?	3	0.?	0.?
User 4	3	0.?	4	0.?	3
User 5	4	3	0.?	4	0.?

Source: [Kat Bailey](#)

Latent User/Item Representations

- An item can be described with a vector v_j (sweet, organic, etc.).
- A user can be described with **another vector u_i** (likes sweet, likes organic, etc.)
- The interaction is defined by how well the item features match the user preferences.

	Feature 1	Feature 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

X

	Item 1	Item 2	Item 3	Item 4	Item 5
Feature 1	?	?	?	?	?
Feature 2	?	?	?	?	?

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0.?	3	0.?	3	0.?
User 2	4	0.?	0.?	2	0.?
User 3	0.?	0.?	3	0.?	0.?
User 4	3	0.?	4	0.?	3
User 5	4	3	0.?	4	0.?

Source: [Kat Bailey](#)

Latent User/Item Representations

- An item can be described with a vector v_j (sweet, organic, etc.).
- A user can be described with another vector u_i (likes sweet, likes organic, etc.)
- The rating on item j by user i is defined by $u_i^\top v_j$.

The diagram illustrates the process of matrix factorization. It shows a user-item rating matrix being decomposed into two lower-dimensional matrices, User Latent Factors and Item Latent Factors, which are then multiplied back together to reconstruct the original matrix.

User-Item Rating Matrix:

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	?	?	?	?	?
User 2	?	?	?	?	?
User 3	?	?	?	?	?
User 4	?	?	?	?	?
User 5	?	?	?	?	?

User Latent Factors Matrix:

	Latent Factor 1	Latent Factor 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

Item Latent Factors Matrix:

	Latent Factor 1	Latent Factor 2
Item 1	?	?
Item 2	?	?
Item 3	?	?
Item 4	?	?
Item 5	?	?

Reconstruction:

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0.?	3	0.?	3	0.?
User 2	4	0.?	0.?	2	0.?
User 3	0.?	0.?	3	0.?	0.?
User 4	3	0.?	4	0.?	3
User 5	4	3	0.?	4	0.?

Source: [Kat Bailey](#)

Latent User/Item Representations

- An item can be described with a vector v_j (sweet, organic, etc.).
- A user can be described with another vector u_i (likes sweet, likes organic, etc.)
- The rating on item j by user i . is defined by $u_i^\top v_j$.
- We minimize

$$\sum_{i,j} \left(r_{i,j} - \left(u_i^\top v_j \right) \right)^2$$

	Feature 1	Feature 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

X

	Item 1	Item 2	Item 3	Item 4	Item 5
Feature 1	?	?	?	?	?
Feature 2	?	?	?	?	?

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0.?	3	0.?	3	0.?
User 2	4	0.?	0.?	2	0.?
User 3	0.?	0.?	3	0.?	0.?
User 4	3	0.?	4	0.?	3
User 5	4	3	0.?	4	0.?

Source: Kat Bailey

Latent User/Item Representations

- An item can be described with a vector v_j (sweet, organic, etc.).
- A user can be described with another vector u_i (likes sweet, likes organic, etc.)
- The rating on item j by user i . is defined by $u_i^\top v_j$.
- We minimize

$$\sum_{i,j} \left(r_{i,j} - \left(u_i^\top v_j + b_{u_i} + b_{v_j} \right) \right)^2$$

	Feature 1	Feature 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

X

	Item 1	Item 2	Item 3	Item 4	Item 5
Feature 1	?	?	?	?	?
Feature 2	?	?	?	?	?

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0.?	3	0.?	3	0.?
User 2	4	0.?	0.?	2	0.?
User 3	0.?	0.?	3	0.?	0.?
User 4	3	0.?	4	0.?	3
User 5	4	3	0.?	4	0.?

Source: [Kat Bailey](#)

Latent User/Item Representations

- An item can be described with a vector v_j (sweet, organic, etc.).
- A user can be described with another vector u_i (likes sweet, likes organic, etc.)
- The rating on item j by user i . is defined by $u_i^\top v_j$.
- We minimize

$$\sum_{i,j} \left(r_{i,j} - \left(u_i^\top v_j + b_{u_i} + b_{v_j} \right) \right)^2$$

$+\alpha \mathcal{R}(U, V)$

	Feature 1	Feature 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

X

	Item 1	Item 2	Item 3	Item 4	Item 5
Feature 1	?	?	?	?	?
Feature 2	?	?	?	?	?

=

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0.?	3	0.?	3	0.?
User 2	4	0.?	0.?	2	0.?
User 3	0.?	0.?	3	0.?	0.?
User 4	3	0.?	4	0.?	3
User 5	4	3	0.?	4	0.?

Source: Kat Bailey

Latent User/Item Representations

- An item can be described with a vector v_j (sweet, organic, etc.).
- A user can be described with another vector u_i (likes sweet, likes organic, etc.)
- The rating on item j by user i . is defined by $u_i^\top v_j$.
- We minimize

$$\sum_{(i,j) \in \mathcal{B}} \left(r_{i,j} - \left(u_i^\top v_j + b_{u_i} + b_{v_j} \right) \right)^2 + \alpha \mathcal{R}(U, V)$$

	Feature 1	Feature 2
User 1	?	?
User 2	?	?
User 3	?	?
User 4	?	?
User 5	?	?

X

	Item 1	Item 2	Item 3	Item 4	Item 5
Feature 1	?	?	?	?	?
Feature 2	?	?	?	?	?

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0.?	3	0.?	3	0.?
User 2	4	0.?	0.?	2	0.?
User 3	0.?	0.?	3	0.?	0.?
User 4	3	0.?	4	0.?	3
User 5	4	3	0.?	4	0.?

Source: [Kat Bailey](#)

What if we don't have ratings?



	Item 1	Item 2	Item 3	Item 4	Item 5
User 1		3	0		0
User 2	4			2	0
User 3	0			0	0
User 4		0	4		3
User 5	4	3	0	4	0

	i_1	i_2	i_3	i_4	
u_1	?	+	+	?	user
u_2	+	?	?	+	
u_3	+	+	?	?	
u_4	?	?	+	+	
u_5	?	?	+	?	
	item				

Source: BPR: Bayesian Personalized Ranking from

Implicit Feedback, Rendle et al. 2012

Implicit Feedback

- For a given user i , an item being interacted j should have a higher score than another item k which was never being interacted.

	i_1	i_2	i_3	i_4	
u_1	?	+	+	?	user ↑ ↓
u_2	+	?	?	+	
u_3	+	+	?	?	
u_4	?	?	+	+	
u_5	?	?	+	?	
	← item →				

Source: *BPR: Bayesian Personalized Ranking*
from *Implicit Feedback*, Rendle et al. 2012

Implicit Feedback

- For a given user i , an item being interacted j should have a higher score than another item k which was never being interacted.
- We maximize

$$\sum_{i,j,k \in I \setminus I_{u_i}} \log \text{sigmoid}((u_i^\top v_j - u_i^\top v_k))$$

	i_1	i_2	i_3	i_4	
u_1	?	+	+	?	user ↑ ↓
u_2	+	?	?	+	
u_3	+	+	?	?	
u_4	?	?	+	+	
u_5	?	?	+	?	
	← item →				

Source: *BPR: Bayesian Personalized Ranking*
from *Implicit Feedback*, Rendle et al. 2012

Implicit Feedback

- For a given user i , an item being interacted j should have a higher score than another item k which was never being interacted.
- We maximize

$$\sum_{i,j,k \in I \setminus I_{u_i}} \log \text{sigmoid}((u_i^\top v_j - u_i^\top v_k))$$

- We usually *sample* one or multiple k when computing gradients (**negative sampling**).
 - Commonly uniformly, but adaptive sampling often helps.

	i_1	i_2	i_3	i_4	
u_1	?	+	+	?	user ↑ ↓
u_2	+	?	?	+	
u_3	+	+	?	?	
u_4	?	?	+	+	
u_5	?	?	+	?	
	← item →				

Source: *BPR: Bayesian Personalized Ranking*
from *Implicit Feedback*, Rendle et al. 2012

How to get u_i and v_j

- The score function: $u_i^\top v_j$

How to get u_i and v_j

- The score function: $u_i^\top v_j$
- Matrix Factorization: where user and item representations are static and independent of each other (fixing the model).

How to get u_i and v_j

- The score function: $u_i^\top v_j$
- Matrix Factorization: where user and item representations are static and independent of each other (fixing the model).
- RNN To integrate user history, $u_i = f(v_{u,1}, v_{u,2}, \dots, v_{u,n})$
 - The user representation now depends on his/her previously interacted items.

How to get u_i and v_j

- The score function: $u_i^\top v_j$
- Matrix Factorization: where user and item representations are static and independent of each other (fixing the model).
- RNN To integrate user history, $u_i = f(v_{u,1}, v_{u,2}, \dots, v_{u,n})$
 - The user representation now depends on his/her previously interacted items.
- Graph-based models to integrate neighboring items/users (in the next slide).
 - The user representation could also depend on behaviors of other users/items.

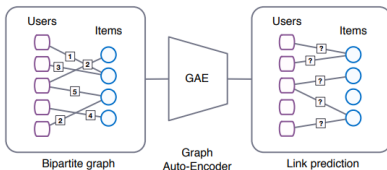
How to get u_i and v_j

- The score function: $u_i^\top v_j$
- Matrix Factorization: where user and item representations are static and independent of each other (fixing the model).
- RNN To integrate user history, $u_i = f(v_{u,1}, v_{u,2}, \dots, v_{u,n})$
 - The user representation now depends on his/her previously interacted items.
- Graph-based models to integrate neighboring items/users (in the next slide).
 - The user representation could also depend on behaviors of other users/items.
- Can combine with content-based recommendation (i.e. with user and item features).

How to get u_i and v_j

- The score function: $u_i^\top v_j$
- Matrix Factorization: where user and item representations are static and independent of each other (fixing the model).
- RNN To integrate user history, $u_i = f(v_{u,1}, v_{u,2}, \dots, v_{u,n})$
 - The user representation now depends on his/her previously interacted items.
- Graph-based models to integrate neighboring items/users (in the next slide).
 - The user representation could also depend on behaviors of other users/items.
- Can combine with content-based recommendation (i.e. with user and item features).
- Scoring function can also change (e.g. to bilinear $u_i^\top Q v_j$)

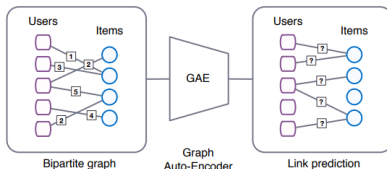
GCMC: Learning u_i and v_j from User-Item Graph



Source: *Graph Convolutional Matrix Completion*, van den Berg et al. 2017

1.
$$\mu_{v_j \rightarrow u_i, r} = \frac{1}{c_{u_i v_j}} W_r x_{v_j}$$

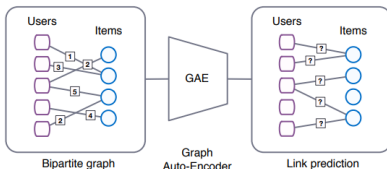
GCMC: Learning u_i and v_j from User-Item Graph



Source: *Graph Convolutional Matrix Completion*, van den Berg et al. 2017

1. $\mu_{v_j \rightarrow u_i, r} = \frac{1}{c_{u_i v_j}} W_r x_{v_j}$
2. $h_{u_i} = \sigma \left[\text{agg} \left(\sum_{v_j \in \mathcal{N}_{u_i, 1}} \mu_{v_j \rightarrow u_i, 1}, \dots, \sum_{v_j \in \mathcal{N}_{u_i, R}} \mu_{v_j \rightarrow u_i, R} \right) \right]$

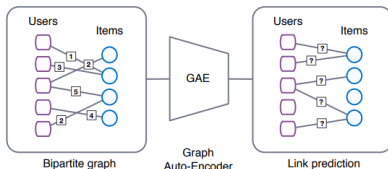
GCMC: Learning u_i and v_j from User-Item Graph



Source: *Graph Convolutional Matrix Completion*, van den Berg et al. 2017

1. $\mu_{v_j \rightarrow u_i, r} = \frac{1}{c_{u_i v_j}} W_r x_{v_j}$
2. $h_{u_i} = \sigma \left[\text{agg} \left(\sum_{v_j \in \mathcal{N}_{u_i, 1}} \mu_{v_j \rightarrow u_i, 1}, \dots, \sum_{v_j \in \mathcal{N}_{u_i, R}} \mu_{v_j \rightarrow u_i, R} \right) \right]$
3. $u_i = \sigma(W_u h_{u_i})$ and similarly we compute v_j

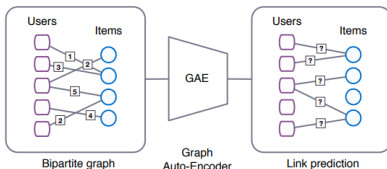
GCMC: Learning u_i and v_j from User-Item Graph



Source: *Graph Convolutional Matrix Completion*, van den Berg et al. 2017

1. $\mu_{v_j \rightarrow u_i, r} = \frac{1}{c_{u_i v_j}} W_r x_{v_j}$
2. $h_{u_i} = \sigma \left[\text{agg} \left(\sum_{v_j \in \mathcal{N}_{u_i, 1}} \mu_{v_j \rightarrow u_i, 1}, \dots, \sum_{v_j \in \mathcal{N}_{u_i, R}} \mu_{v_j \rightarrow u_i, R} \right) \right]$
3. $u_i = \sigma(W_u h_{u_i})$ and similarly we compute v_j
4. $p(\hat{M}_{ij} = r) = \text{softmax}(u_i^\top Q_r v_j)$

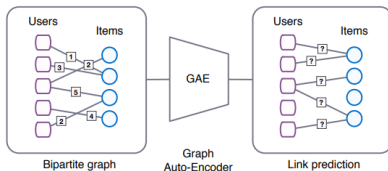
GCMC: Learning u_i and v_j from User-Item Graph



Source: *Graph Convolutional Matrix Completion*, van den Berg et al. 2017

1. $\mu_{v_j \rightarrow u_i, r} = \frac{1}{c_{u_i v_j}} W_r x_{v_j}$
2. $h_{u_i} = \sigma \left[\text{agg} \left(\sum_{v_j \in \mathcal{N}_{u_i, 1}} \mu_{v_j \rightarrow u_i, 1}, \dots, \sum_{v_j \in \mathcal{N}_{u_i, R}} \mu_{v_j \rightarrow u_i, R} \right) \right]$
3. $u_i = \sigma(W_u h_{u_i})$ and similarly we compute v_j
4. $p(\hat{M}_{ij} = r) = \text{softmax}(u_i^\top Q_r v_j)$
5. When new interactions are added, just re-run the forward pass on the new graph to get new u_i and v_j .

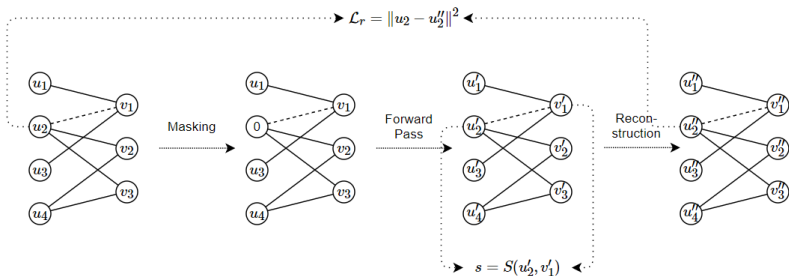
Simplifying GCMC to GraphSAGE



Source: *Graph Convolutional Matrix Completion*, van den Berg et al. 2017

1. $\mu_{v_j \rightarrow u_i, r} = \frac{1}{c_{u_i v_j}} \mathbf{W} \mathbf{x}_{v_j}$
2. $h_{u_i} = \sigma \left[\text{agg} \left(\sum_{v_j \in \mathcal{S}(\mathcal{N}_{u_i, 1})} \mu_{v_j \rightarrow u_i, 1}, \dots, \sum_{v_j \in \mathcal{S}(\mathcal{N}_{u_i, R})} \mu_{v_j \rightarrow u_i, R} \right) \right]$
3. $u_i = \sigma(\mathbf{W}_u h_{u_i})$ and similarly we compute v_j
4. $r_{i,j} = u_i^\top v_j$ to predict rating
5. When new interactions are added, just re-run the forward pass on the new graph to get new u_i and v_j .

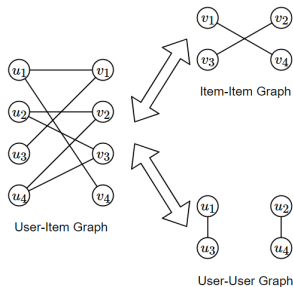
Learning u_i and v_j with Star-GCN



- Vanilla GCMC can't deal with new users/items without features (but with a few interactions).
- STAR-GCN
 - "Mask" the user/item embedding to 0 as if it is new.
 - Reconstruct the embedding after the forward pass and reconstruction pass.

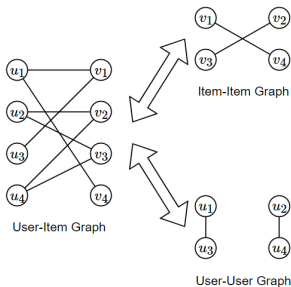
Learning v_j and u_i from Item-Item and User-User Graph

- Decompose the user-item graph into user-user graph and item-item graph.



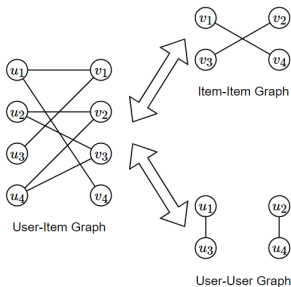
Learning v_j and u_i from Item-Item and User-User Graph

- Decompose the user-item graph into user-user graph and item-item graph.
- Get u_i with a Graph Convolutional Network on user-user graph and v_j on item-item graph.



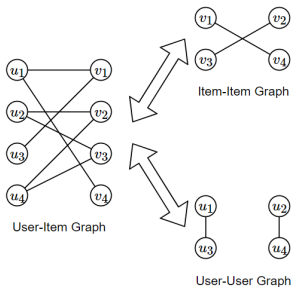
Learning v_j and u_i from Item-Item and User-User Graph

- Decompose the user-item graph into user-user graph and item-item graph.
- Get u_i with a Graph Convolutional Network on user-user graph and v_j on item-item graph.
- Compute $r_{i,j} = u_i^\top v_j$



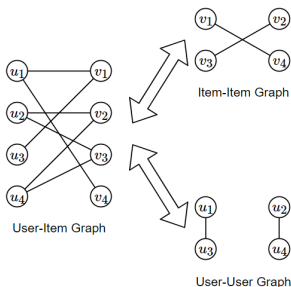
Learning v_j and u_i from Item-Item and User-User Graph

- Decompose the user-item graph into user-user graph and item-item graph.
- Get u_i with a Graph Convolutional Network on user-user graph and v_j on item-item graph.
- Compute $r_{i,j} = u_i^\top v_j$
- u_i and v_j can be learned with
 - Direct neighbor sampling (GraphSAGE)
 - Random-walk based neighbor sampling (PinSAGE)



Learning v_j and u_i from Item-Item and User-User Graph

- Decompose the user-item graph into user-user graph and item-item graph.
- Get u_i with a Graph Convolutional Network on user-user graph and v_j on item-item graph.
- Compute $r_{i,j} = u_i^\top v_j$
- u_i and v_j can be learned with
 - Direct neighbor sampling (GraphSAGE)
 - Random-walk based neighbor sampling (PinSAGE)
- When new interactions are added, just re-run the forward pass on the new graph to get new u_i and v_j .



Other Meaningful Aspects to Consider

- **Cold-start:** What if we have *new* users and items coming in, with few to no historical interactions?

Other Meaningful Aspects to Consider

- **Cold-start:** What if we have *new* users and items coming in, with few to no historical interactions?
- **Bias correction:** The training dataset usually comes from the result of a *previous recommender system*. How to mitigate the bias?

Other Meaningful Aspects to Consider

- **Cold-start:** What if we have *new* users and items coming in, with few to no historical interactions?
- **Bias correction:** The training dataset usually comes from the result of a *previous recommender system*. How to mitigate the bias?
- **Diversity:** Always recommending the same items (or even the same kind of item) to a user would make him/her feel *bored*.

Other Meaningful Aspects to Consider

- **Cold-start:** What if we have *new* users and items coming in, with few to no historical interactions?
- **Bias correction:** The training dataset usually comes from the result of a *previous recommender system*. How to mitigate the bias?
- **Diversity:** Always recommending the same items (or even the same kind of item) to a user would make him/her feel *bored*.
- **Fraud:** How to detect and deal with fabricated explicit feedbacks (e.g. fake ratings and reviews)?

Coding Session

GraphSAGE on bipartite user-item graph.