

# Neural Graph Collaborative Filtering

## Collaborative Filtering:

유저-아이템에 유용해 유저 - 아이템 간의 상호작용을 통해 유저에게 알맞은 새로운 아이템을 추천하는 방법을 의미.

### Collaborative Filtering

#### ① Neighborhood Method

: 많은 계산

#### ② Latent Factor Model \*

#### Matrix Factorization

Matrix Factorization: 유저 - 아이템 상호작용에는 유저의 행동, 아이템 특성에 영향을 주는 잠재인요인 (latent factor)이 있을텐데 그 잠재인 요인을 고려하여 유저에게 적합한 아이템을 추천해주는 방법 (잠재요인 고려)

$$\text{유저-아이템 행렬} = \text{유저-잠재요인} \times \text{아이템-잠재요인}$$

$$m \times n \qquad \qquad m \times k \qquad \qquad k \times n$$

$$\hat{j}_{ui} = f(u_i, i | P_u, q_i) = P_u^T q_i = \sum_{k=1}^k P_{uk} q_{ik},$$

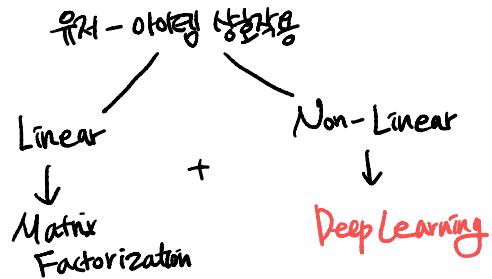
유저와 아이템을 위한  
잠재변수

유저와 아이템  
잠재변수의 내적

Latent Factors 간의 상호작용은 내적 (inner product)로  
유저-아이템 상호작용에 대한 예상 결과를 계산하게 됨.

→ 내적을 사용해 예상과 실제가 차이 나는 경우가 발생  
(예를 들어 내적은 차이를 표기해 이를 표기해야 한다)

NGCF: Deep Neural Network을 사용하여 유저-아이템 상호작용을 학습하는 프레임워크 제안.



## [Neural Collaborative Filtering]

Embedding Layer : 유저-아이템 간의 implicit features 사용하고,  
회복한 유저(아이템) implicit은 현재 표시 부대에서  
유저(아이템)의 예상, 장르별로 볼 수 있다.

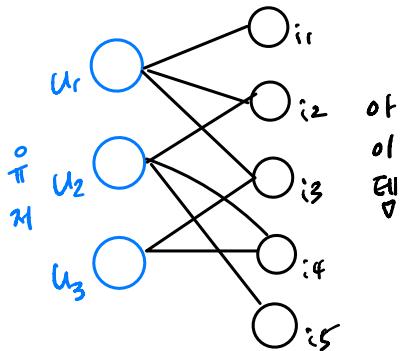
Neural CF Layer : 유저 및 아이템 implicit은 장르 별로 예측 가능에 대처하기 위해  
(자유로 높음) Multi-layer neural을 적용한 Neural CF에 활용된다  
(non-linear) 또래 Multi Layer Perceptron을 활용하였다.  
(flexible) 각각 Layers는 모델의 capability를 확장

Neural Matrix Factorization : Neural CF + Generalized MF  
 (기능구현)  
 ↗  
 ↗  
 ↗  
 ↗

But 유저, 아이템 각각 implicit하는 기준 빠트려서 예측을 끝낼 때까지  
유저-아이템 간의 상호작용을 차단하는데 부족.

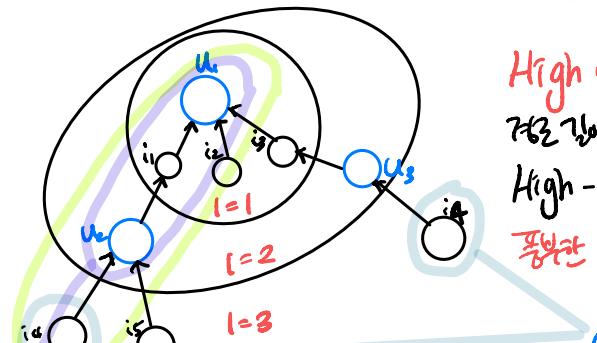
↓  
 Neural Graph Collaborative Filtering 등장

## 일반적인 유저 - 아이템 상호작용 그래프



User-item interaction graph

## 고차유연 그래프 형태



High order connectivity for  $U_1$

### High order Connectivity

각각 다른 노드에서  $U_1$ 에 도달하는 경로를 의미  
High-order connectivity에는 **Higher階層을 지닌  
복잡한 연결**이 포함.

- ④  $U_1$ 은  $i_4$ 보다  $i_5$ 에 더 가깝기 때문에 가능성이 높다.

$$(\#i_4 : 2, \#i_5 : 1)$$

- ①  $U_1$ 과  $U_2$ 가 상호작용함
- ②  $U_1$ 과  $U_2$ 가 행동유사성이 존재
- ③  $U_1$ 과 유사한  $U_2$ 가 이미 아이템  $i_4$ 를 선호  $\rightarrow$   $U_1$ 이 아이템  $i_4$ 를 선택할 가능성이 존재

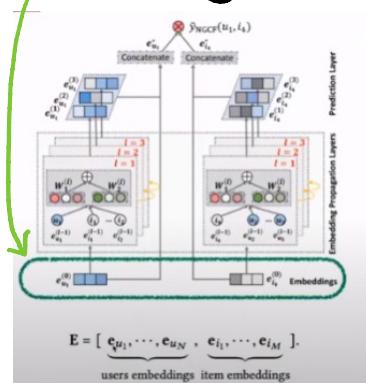
$\therefore$  그래프 네트워크를 통해서  
유저-아이템 간의 상호작용과 비슷한  
유저와의 유사성을 유저에게 노드로  
아이템을 추천하고 동시에 유저가  
어떤 아이템을 선호할지 예측가능.

# NGCF

## Embedding Layer :

(아프로젝트 사용)

일반적인 헤지 필터링 모델처럼 각각의 유저와 아이템에 대해  
인접한 유저와 아이템에 대한 정보를 학습하는 것.



전체 모델에서 유저와 아이템의 인접한 값은 end-to-end  
방식으로 학습됨.

## Embedding Propagation Layer \*

\* 그래프 구조에 따라 형상 노드를  
제작하고 유저 및 아이템 인접성을 계산하기 위해  
GNN의 마지막 단계 아키텍처를 구축함.

(형상 노드 : 유저(아이템)가 한 행동의 유사성을  
나타내는 유저-유저-아이템 상호작용에 참여한 모든)

First-Order Propagation : 유저가 아이템을 선택한 때,  
아이템 장바구니 유저에게 전달됨.

특정 유저가 아이템을 선택한 정확히 아이템 바로  
사용할 수 있음.

이를 통해 아이템과 유저의 유사성을 측정할 수 있음!

① Embedding Propagation Layer는 양방향 유저-아이템 간의 인접성 관계를 수행하여  
② Message Construction 과 Message Aggregation 두 가지 주요 작업으로 구성

① Message Construction : 유저(유저) 정답과 유저(아이템)에게 전달되는  
메시지를 계산. 아이템 유저 연결 정보 전달하는 것

$$m_{u \leftarrow i} = \frac{1}{\sqrt{|N_u| |N_i|}} (W_1 e_i + W_2 (e_i \odot e_u))$$

아이템 출력  
유저가 전달하는  
아이템 차지값

② Message Aggregation :  $e_u^{(0)} = \text{LeakyReLU}(m_{u \leftarrow u} + \sum_{i \in N_u} m_{u \leftarrow i})$

<span style="color: blue;">first-hop 출력 전파되는 아이 템 유저의 정보</span>	<span style="color: blue;">유저 차지 연결 고려하여 다른 유저의 정보</span>	<span style="color: blue;">( -hop 이웃도 전달하는</span>
---	---	---

### ∴ Embedding Propagation Layer

: 유저와 아이템의 표현(representation)을 관련 갖기 위해  
각각 유저와 아이템의 차 연결 정보를 학습적으로 활용한다는 것

노드와 엣지 만듬  $\rightarrow$  임베딩 표현체계에 따른 통제  
다음 레이어로 인코딩

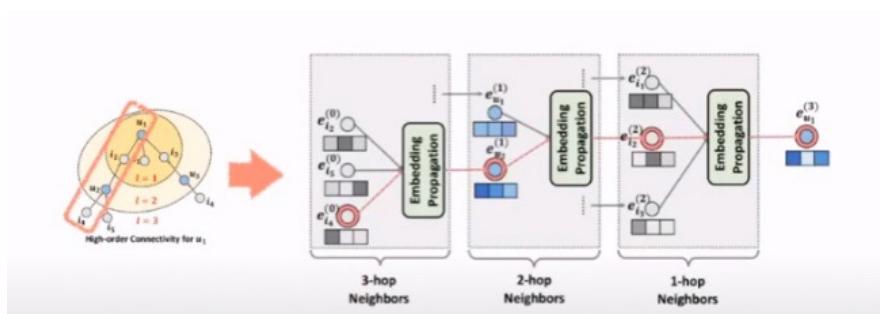
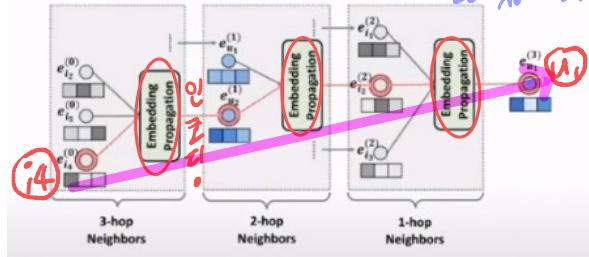
High-order Propagation : 임베딩 표현체계에서 이미지가 여러 번  $\frac{1}{k}$ 에 합성됨  
 $\Rightarrow$  상위 연결 정보 탐색 가능

# High-order Propagation

High-order는 유저-아이템간의 관련성을 추적하기 위해 헝겊 시그널을 암호화하는  
기술 중요

cf. Latent factor을 추적하는 문제이기 때문에

"신생유저"와 노동과 같은 다른 사용자들은 암호 없음.



High-order Propagation을 통해 헝겊 시그널을 암호화·전파 과정에서  
암호 있고 그걸儿 미리 예상적으로 암호화하는 것을 알 수 있다.

또한 암버디! 전파레이어에는 넣어주면 헝겊 시그널이 표현되는 과정에參與하게 되면

2<sup>nd</sup> 차원 베이트릭스로 Propagation Rule은

$$L = D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \text{ and } A = \begin{bmatrix} 0 & R \\ R^T & 0 \end{bmatrix}$$

이렇게 구조화된다

모든 유저와 아이템에 대한 정보를 효율적으로 동시에 암호화할 수 있고

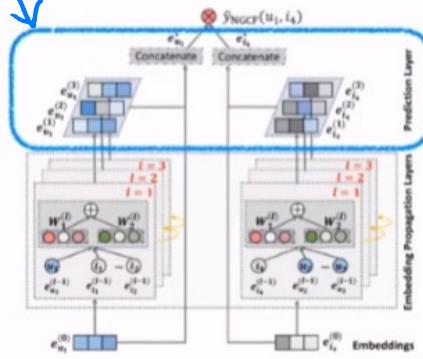
또한 사용자 정보를 별도로 넣어주면 헝겊 시그널이 표현되는 과정에參與하게 되면

# Model Prediction Layer

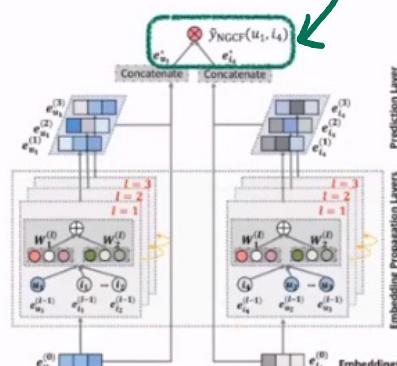
서로 다른 아이템에서 나온 표현은 같은 다른 메시지를 강조하기 때문에  
유저 속도 벡터에 서로 다른 가중치를 한다.

다른 아이템을 Concatenate하여 유저 (or 아이템)을 위한 최종 임베디팅 구성을 한다.

$$\mathbf{e}_u^* = \mathbf{e}_u^{(0)} \| \cdots \| \mathbf{e}_u^{(L)}, \quad \mathbf{e}_i^* = \mathbf{e}_i^{(0)} \| \cdots \| \mathbf{e}_i^{(L)},$$



# Output Layer



타깃 아이템에 대한 유저의 속도도 추정 가능

$$\hat{y}_{NGCF}(u, i) = \mathbf{e}_u^{*T} \mathbf{e}_i^*$$

(다음화)

이 과정을 통해 유저는 속도가 가장 높은,  
아이템을 추천 받게 됨! (그래프 네트워크 유저  
가장 가까운 유저들 가중치가  
높은 뉴스로 유저의 속도가  
추정됨)

# NGCF

data:

Table 1: Statistics of the datasets.

Dataset	#Users	#Items	#Interactions	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

평가지표: recall, ndcg

recall@K: 현재 유저가 관심 있는 아이템 중 추천된 아이템의 비율을 평가하는 지표  
ndcg@K: 추천 높이에 기여도를 주어도 비율을 평가하는 지표

비교 모델: MF, NeuralMF, 다양한 SOTA 모델들

성능비교 #1

Table 2: Overall Performance Comparison.

	Gowalla		Yelp2018*		Amazon-Book	
	recall	ndcg	recall	ndcg	recall	ndcg
MF	0.1291	0.1109	0.0433	0.0354	0.0250	0.0196
NeuMF	0.1399	0.1212	0.0451	0.0363	0.0258	0.0200
CMN	0.1405	0.1221	0.0457	0.0369	0.0267	0.0218
HOP-Rec	0.1399	0.1214	0.0517	0.0428	0.0309	0.0232
GC-MC	0.1395	0.1204	0.0462	0.0379	0.0288	0.0224
PinSage	0.1380	0.1196	0.0471	0.0393	0.0282	0.0219
NGCF-3	0.1569*	0.1327*	0.0579*	0.0477*	0.0337*	0.0261*
%Improv.	11.68%	8.64%	11.97%	11.29%	9.61%	12.50%
p-value	2.01e-7	3.03e-3	5.34e-3	4.62e-4	3.48e-5	1.26e-4

여기서 NGCF는 고차항을 활용해  
NGCF는 high-order 차원을  
활용할 수 있다. (다양한 연령층)

But, CMN, GC-MC는  
first-order 차원 활용함.

→ 따라서 NGCF는 다양한 차원의 중요성을  
알 수 있음.

성능비교 #2

Table 2: Overall Performance Comparison.

	Gowalla		Yelp2018*		Amazon-Book	
	recall	ndcg	recall	ndcg	recall	ndcg
MF	0.1291	0.1109	0.0433	0.0354	0.0250	0.0196
NeuMF	0.1399	0.1212	0.0451	0.0363	0.0258	0.0200
CMN	0.1405	0.1221	0.0457	0.0369	0.0267	0.0218
HOP-Rec	0.1399	0.1214	0.0517	0.0428	0.0309	0.0232
GC-MC	0.1395	0.1204	0.0462	0.0379	0.0288	0.0224
PinSage	0.1380	0.1196	0.0471	0.0393	0.0282	0.0219
NGCF-3	0.1569*	0.1327*	0.0579*	0.0477*	0.0337*	0.0261*
%Improv.	11.68%	8.64%	11.97%	11.29%	9.61%	12.50%
p-value	2.01e-7	3.03e-3	5.34e-3	4.62e-4	3.48e-5	1.26e-4

여기서 NGCF는 차원을 활용해 NGCF는  
multi-grained 차원을 활용하는 차원  
PinSage는 차원의 계층화의 차원을 활용함.

→ 이를 통해 서로 다른 전파 메시지를  
포함에서 서로 다른 정보를 인코딩하는 것을  
알 수 있음.

# NGCF 설계 2

## Embedding Propagation Layer가太多的 가지는 경우

Table 3: Effect of embedding propagation layer numbers (L).

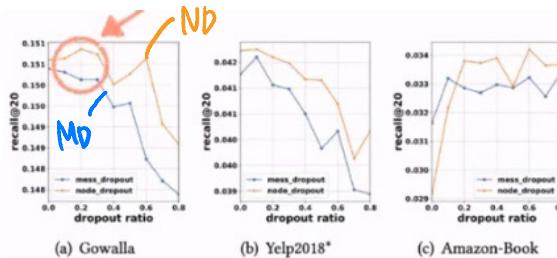
	Gowalla		Yelp2018*		Amazon-Book	
	recall	ndcg	recall	ndcg	recall	ndcg
NGCF 1	0.1556	0.1315	0.0543	0.0442	0.0313	0.0241
NGCF 2	0.1547	0.1307	0.0566	0.0465	0.0330	0.0254
NGCF 3	0.1569	0.1327	0.0579	0.0477	0.0337	0.0261
NGCF-4	0.1570	0.1327	0.0566	0.046	0.0344	0.0263

high

overfitting

# NGCF 설계 3

오버피팅 방지 방법을 하는 두 가지



NGCF에서 embedding 전파 계층이 많아질 때 recall과 ndcg가 크게 향상된다.

즉, 축적이 강화됨.

→ high-order connectivity 모델링이 우수함

random selection  
모든 노드를 선택

random selection  
부수적인 노드만 선택

Dropout의 효과

Node Dropout & Message Dropout 사용

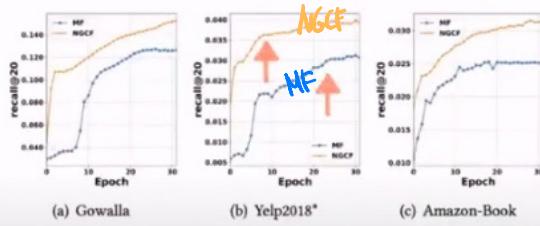
특정 유저(아이템)에 대해서는 모든 노드를 선택, 즉 노드 간의 연결 관계를 유지하면서도 특정 노드를 제거하는 것을 한다.

Node Dropout의 비중이 커 질 때.

→ Node Dropout이 GNN 오버피팅을 해결하는 중요한 역할을 할 수 있음.

# NGCF 실험 4

Epoch별 recall@20 대비 테스트 성능

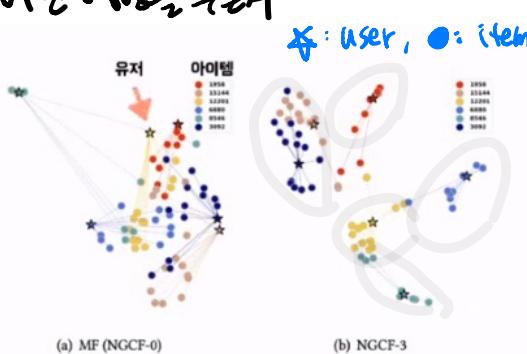


NGCF가 MF보다 빠르게 수렴

연결된 유저-아이템은 미니Batch화되어  
시작작용 사용을 최적화하기 때문  
→ NGCF의 모델 용량이 줄고 임베딩 공간에서  
유저와 전파를 수행하는 효과를 보여줌.

# NGCF 실험 5

Embedding Propagation Layer의 구조가 유저와商品 공간에서 표현 학습에  
영향 여부를 살펴보자



유저-아이템 연결을 임베딩 공간에서 바로  
기제로 부분에 임베드됩니다.

이를 통해 동일한 유저가 소비한 아이템(같은색상)을  
기반 포인트가 클러스터링을 형성하는 경향을  
알 수 있음.

NGCF 레이어가 그림처럼 고지아이템의  
影响力에 더 기제운 경향을 보임.

→ Embedding Propagation Layer가  
협업 NGCF를 통해 직접 주장할 수 있음

# 결론

## \* Embedding Propagation Layer

Message Construction at Message Aggregation 단계를 통해  
High-order connectivity 그래프를 만들 후, (GNN 적용)  
이웃들을 자신의 베이스로 표기하여  
유저-아이템 선호도를 추정할 수 있는 방안을 제안

∴ 내가 비슷한 취향의 > 친구의 거의 거의  
친구가 될 예상친구 본 예상 추천