LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation

Cheolhee Jung

DMLAB

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, Meng Wang
"LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation"

Contents

- Background
- Problem definition
- Proposed method
- Experiment
- Conclusion
  - strong points
  - weak points
    - Improve or resolve those weak point

Background

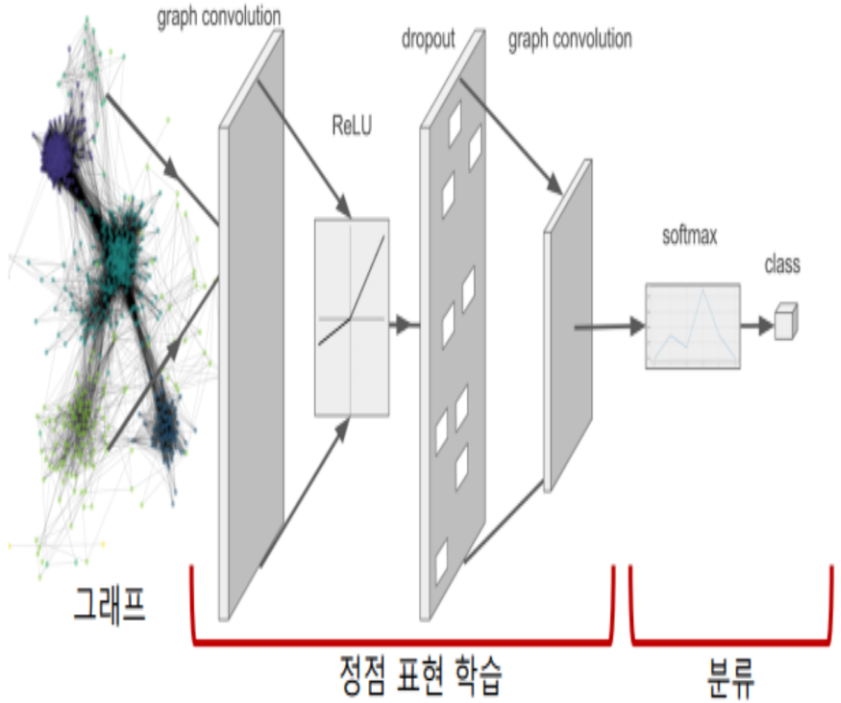
▷ User – Item data를 통해 추천 시스템을 만들 수 있다(CF)

▷ 잠재특성을 효율적으로 파악하기 위해 NGCF 제안

▷추천 시스템에서 NGCF 같은 GCN 기반 모델이
실제로는 필요하지 않는 설계 포함

Background

▷ What is Graph Neural Networks(GNN)



· **그래프 컨볼루션 (Graph Convolution)**:
그래프의 노드들이 자신의 특징과 이웃 노드들의 특징을 집계하여 새로운 임베딩을 생성 ➡ 지역적 정보 포착

· **비선형 활성화 함수** : 그래프 컨볼루션을 통해 생성된 임베딩에 비선형 변환을 적용 ➡ 이는 모델이 복잡한 패턴을 학습

· **드롭아웃 (Dropout)**: 오버피팅을 방지하기 위해 일부 노드의 연결을 임의로 끊는 과정

· **소프트맥스 (Softmax)**: 마지막 단계에서는 각 노드의 임베딩을 클래스의 확률 분포로 변환

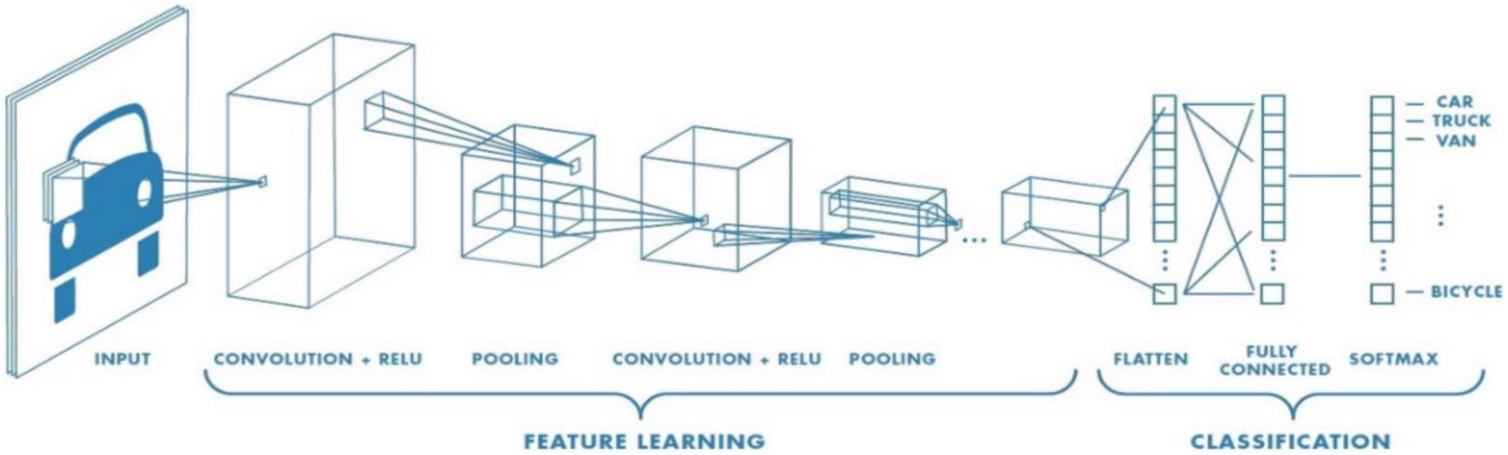
· **클래스 (Class)**: 최종적으로, 각 노드는 소프트맥스를 통해 가장 높은 확률을 가진 클래스로 분류

➡ **그래프 데이터의 복잡한 관계를 학습**

https://wonbae.github.io/2021-02-26-Graph9-GNN/

Background

▷ What is Graph Convolutional Networks (GCN)



그래프 구조에 대해 convolution 연산을 적용하는 GCN

특징

・특성변환

・비선형 활성화

https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Background

▷ What is NGCF?

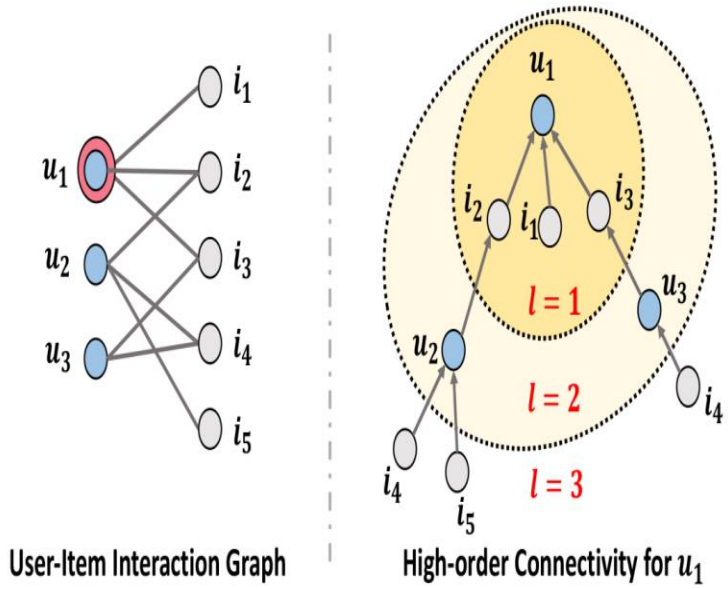


Figure 1: An illustration of the user-item interaction graph and the high-order connectivity. The node u_1 is the target user to provide recommendations for.

전통적인 협업 필터링은 user-item interaction만을 고려하여 제품과 다른 사용자 간의 관계 즉, 고차-연결성을 고려하지 못함

Neural Graph Collaborative Filtering (NGCF)

= CF + Graph

➡ **고차-연결성(High-Order Connectivity) 극복**

예) $u_1 \leftarrow i_2 \leftarrow u_2 \leftarrow i_4$라는 그래프 상의 경로가 있을 때 그래프 구조에서는 이 경로를 모두 고려해 이웃 사용자들로부터 협력 신호를 제공 받아 u_1에게 i_4를 추천할 정보를 얻음

Background

▷ Points of Attention in Graph Structure

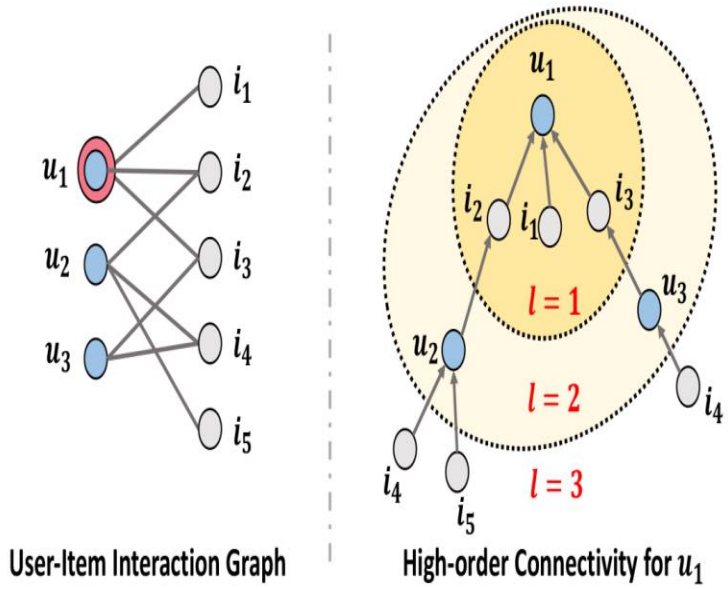


Figure 1: An illustration of the user-item interaction graph and the high-order connectivity. The node u_1 is the target user to provide recommendations for.

1. **Self-connection** : 각 노드는 자신의 정보도 중요하게 여겨야하지만, 그래프에서는 노드가 자기 자신을 가르치는 link가 없기 때문에 이를 모델에 추가해줘야 함,
NGCF는 각 레이어의 연산에서 노드 자신의 초기 임베딩을 포함시켜 self-connection을 구현함
(임베딩 업데이트 수식에서 확인)

2. **정규화** : 간선이 많이 연결된 노드는 임베딩 행렬에서 큰 값으로 이어질 수 있으며, 이는 학습과정에서 기울기 폭발, 소실을 일으킬 수 있음
이를 방지하기 위해 정규화를 사용하여 각 노드 임베딩을 조정함

"Neural Graph Collaborative Filtering" - Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, Tat-Seng Chua

Background

▷ NGCF 임베딩 업데이트(표기)

$e_u^{(k)}$: (1)k번째 레이어에서의 사용자 u의 임베딩

σ : 비선형 활성화 함수

W_1, W_2 : (2)특성 변환을 수행하는 학습 가능한 가중치 행렬
w_1 사용자 u의 현재 임베딩 $e_u^{(k)}$에 선형 변환 적용
w_2 사용자 u의 임베딩과 각 아이템 i의 임베딩 사이의 요소별 곱셈을 통해 얻은 벡터에 선형 변환을 적용

N_u : 사용자 u와 상호작용한 아이템의 집합

N_i : 아이템 i와 상호작용한 사용자의 집합

$\odot$: 요소별 곱셈 => 사용자와 아이템 간의 관계를 임베딩 벡터에 직접적으로 반영

(1) : 그래프 신경망 내에서 임베딩이 전파되는 순서 또는 깊이
(2) : 기본적으로 데이터의 원래 특성을 새로운 특성 공간으로 변환 하는 과정, 그래프 구조 내 노드를 벡터 형태로 변환

Background

▷ NGCF 임베딩 업데이트

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

σ 임베딩에 비선형성을 도입 => 더 복잡한 패턴을 학습할 수 있도록 함

$W_1 e_u^{(k)}$ 사용자 u의 현재 임베딩 $e_u^{(k)}$에 가중치 적용 => 노드 자신에 대한 정보를 다음 레이어에 전달(self-connection)

$\sum_{i\in N_u}$ 사용자 u가 상호 작용한 모든 아이템 i에 대한 합산

$\frac{1}{\sqrt{|N_u||N_i|}}$ 정규화 인자 => 그래프의 다양한 부분이 동등하게 임베딩 학습 과정에 기여

$W_1 e_i^{(k)}$ 각 아이템 i의 현재 임베딩에 대한 선형 변환 => 사용자 u와 상호 작용한 아이템의 정보 갱신

$W_2(e_i^{(k)} \odot e_u^{(k)})$ 요소별 곱셈은 사용자와 아이템 간의 상호작용을 더 직접적으로 모델링합니다.

각 사용자의 임베딩은 그들의 이웃(상호작용한 아이템들)으로부터 정보를 수집하고,
그 결과는 사용자의 새로운 임베딩에 반영

Background

▷ What is Problem?

Table 1: Performance of NGCF and its three variants.

	Gowalla		Amazon-Book	
	recall	ndcg	recall	ndcg
NGCF	0.1547	0.1307	0.0330	0.0254
NGCF-f	0.1686	0.1439	0.0368	0.0283
NGCF-n	0.1536	0.1295	0.0336	0.0258
NGCF-fn	0.1742	0.1476	0.0399	0.0303

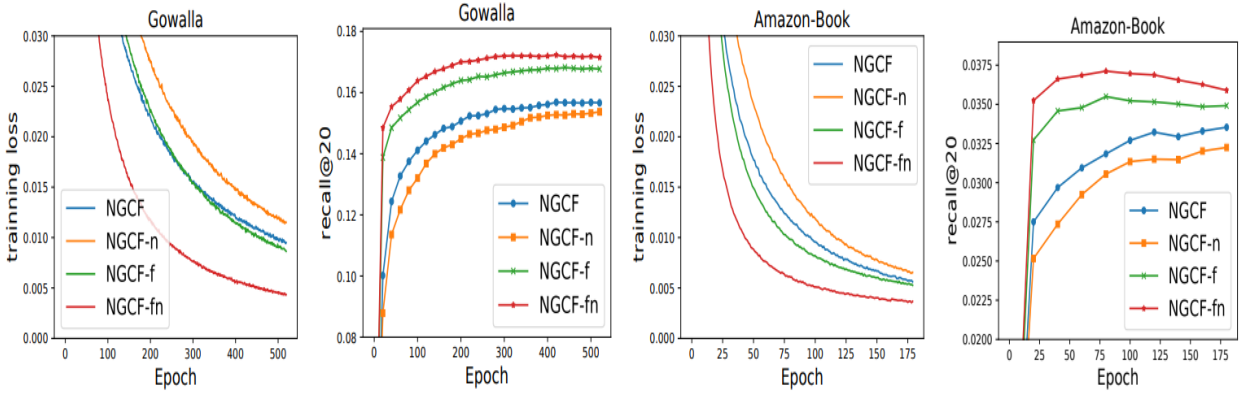


Figure 1: Training curves (training loss and testing recall) of NGCF and its three simplified variants.

평가 지표
Recall : 시스템이 얼마나 많은 관련 아이템을 찾아냈는지
Ndcg : 추천된 아이템들이 사용자에게 얼마나 적합한지를 순위에 따라 가중치를 두어 평가

NGCF-n : σ가 없는 버전
NGCF-f : feature transformation이 없는 버전
NGCF-fn : 둘 다 없는 버전

feature transformation(w_1, w_2), 비선형 활성화(σ)가 성능에 악영향 즉 불필요한 연산

Problem definition

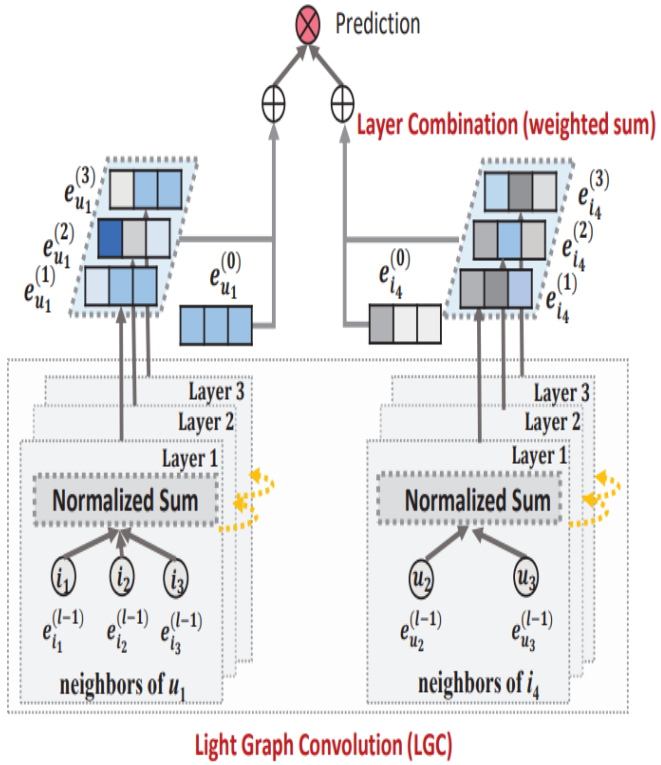
▷ NGCF의 불필요한 복잡성

NGCF가 GCN에서 상속 받은 두 가지 설계
1) 특성 변환 2) 비선형 활성화가 협력 필터링에 긍정적인 영향을 주지 않음

필요하지 않은 복잡한 설계 사용하지 말자

Proposed method

▷Light Graph Convolution (LGC)



Normalized sum

- 그래프 내 각 노드가 이웃 노드들의 정보를 수집하는 방식 정의

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

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

Weighted sum

-여러 레이어에 걸쳐 얻은 임베딩을 통합하여 최종 임베딩 생성

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

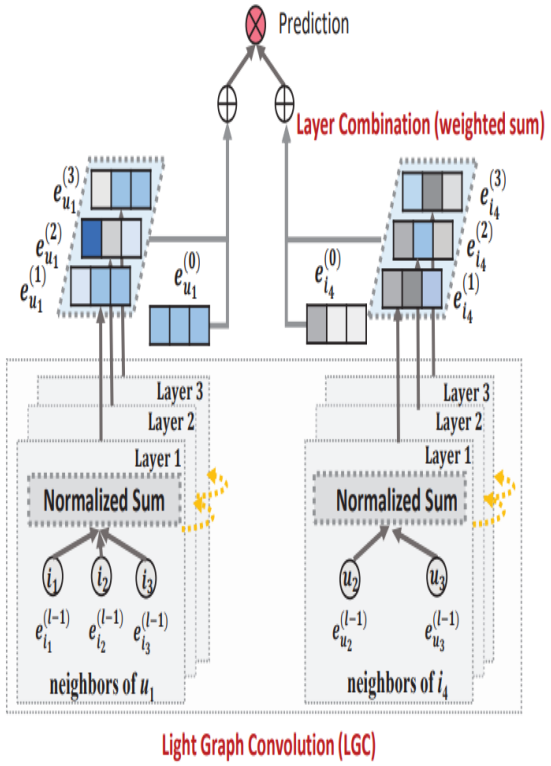
예측 결과

- final embedding을 서로 내적하여 얻음 $\hat{y}_{u_i} = e_u^T e_i$

➡ **특성변환과 비선형 활성화를 제거하여 기존 GCN 간소화**

Proposed method

▷Light Graph Convolution (LGC)



Light GCN은 모델의 첫 번째 레이어의 임베딩 파라미터e_u^0 e_i^0만 학습

·**간소화된 학습 과정**: 첫 번째 레이어의 임베딩만을 학습함으로써, 모델의 매개변수 수와 학습 과정이 간소화

·**효율적인 정보 전파**: LightGCN의 나머지 레이어는 학습된 첫 번째 레이어의 임베딩을 기반으로 정보를 전파

·(1)**오버스무딩 문제 해결**: 고차 레이어의 임베딩을 직접 학습하지 않고 첫 번째 레이어만을 학습함으로써, LightGCN은 네트워크가 너무 깊어질 때 발생할 수 있는 오버스무딩(over-smoothing) 문제를 완화

·**자체 연결 효과의 내재화**: LightGCN에서는 자체 연결을 명시적으로 추가하지 않지만, 첫 번째 레이어의 임베딩이 최종 임베딩에 중요한 기여를 하기 때문에, 별도의 자체 연결을 추가할 필요가 없음 (self-connection).

(1) : 오버스무딩이란 너무 많은 레이어를 통과하면서 노드 간의 임베딩이 구별하기 어려울 정도로 비슷해지는 현상

Proposed method

▷LGC Matrix Form

·Notation

$A = \begin{bmatrix} 0 & R^T \\ R & 0 \end{bmatrix}$ **Adjacency Matrix** : 사용자와 아이템간의 관계, R은 사용자 아이템 상호작용 행렬

Diagonal Degree Matrix : D는 대각 차수 행렬로, 각 노드의 연결 정도를 나타냄
D의 원소 Dii는 인접행렬 A의 i번째 행벡터에 존재하는 nonzero entry 개수
(노드에 연결된 edge수)

$\boldsymbol{E^{(k+1)}}$ **:** (k+1)th layer의 임베딩

·LGC 식

$\mathbf{E}^{(k+1)} = (\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})\mathbf{E}^{(k)}$, 연결된 edge수로 정규화

·Embedding Matrix for Prediction

$$\mathbf{E} = \alpha_0\mathbf{E}^{(0)} + \alpha_1\mathbf{E}^{(1)} + \alpha_2\mathbf{E}^{(2)} + \ldots + \alpha_K\mathbf{E}^{(K)}$$

$$= \alpha_0\mathbf{E}^{(0)} + \alpha_1\tilde{\mathbf{A}}\mathbf{E}^{(0)} + \alpha_2\tilde{\mathbf{A}}^2\mathbf{E}^{(0)} + \ldots + \alpha_K\tilde{\mathbf{A}}^K\mathbf{E}^{(0)},$$

Proposed method

▷ Self – connection in Simplified GCN

$$\mathbf{E}^{(k+1)} = (\mathbf{D}+\mathbf{I})^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})(\mathbf{D}+\mathbf{I})^{-\frac{1}{2}}\mathbf{E}^{(k)},$$

$\mathbf{I}$: 자체 연결을 포함하기 위한 단위행렬

$$\begin{aligned}\mathbf{E}^{(K)} &= (\mathbf{A}+\mathbf{I})\mathbf{E}^{(K-1)} = (\mathbf{A}+\mathbf{I})^{K}\mathbf{E}^{(0)} \\ &= \binom{K}{0}\mathbf{E}^{(0)} + \binom{K}{1}\mathbf{A}\mathbf{E}^{(0)} + \binom{K}{2}\mathbf{A}^{2}\mathbf{E}^{(0)} + \ldots + \binom{K}{K}\mathbf{A}^{K}\mathbf{E}^{(0)}. \\ &= \alpha_0\mathbf{E}^{(0)} + \alpha_1\tilde{\mathbf{A}}\mathbf{E}^{(0)} + \alpha_2\tilde{\mathbf{A}}^{2}\mathbf{E}^{(0)} + \ldots + \alpha_K\tilde{\mathbf{A}}^{K}\mathbf{E}^{(0)},\end{aligned}$$

$\boldsymbol{E}^{(k)}$: 마지막 레이어 임베딩

$\tilde{\boldsymbol{A}}$: 정규화된 인접 행렬

LGC는 self – connection 포함

Proposed method

▷ APPNP과 연관성

APPNP : Personalized PageRank에서 영감을 받아 오버스무딩을 해결하는 GCN 변형 모델

APPNP는 각 전파 레이어에 시작 특성(즉, 0번째 레이어 임베딩)을 보완하여 로컬 정보를 보존

·APPNP의 propagation layer

$$\mathbf{E}^{(k+1)} = \beta\mathbf{E}^{(0)} + (1-\beta)\tilde{\mathbf{A}}\mathbf{E}^{(k)}$$

$\boldsymbol{\beta}$: $E^{(0)}$반영 비율
$\tilde{\boldsymbol{A}}$: 정규화된 인접 행렬

Proposed method

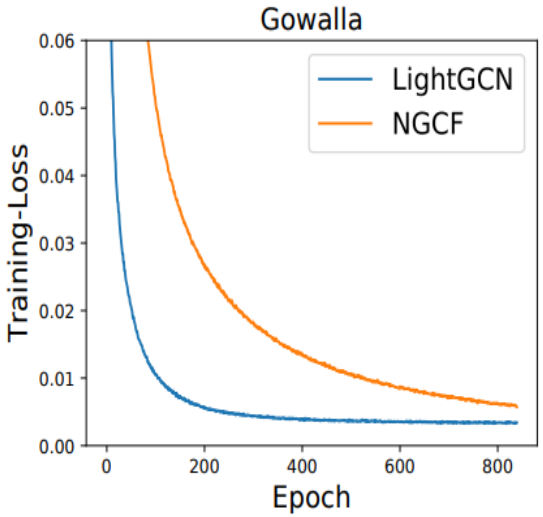
▷ 두 번째 레이어의 Embedding Smoothness

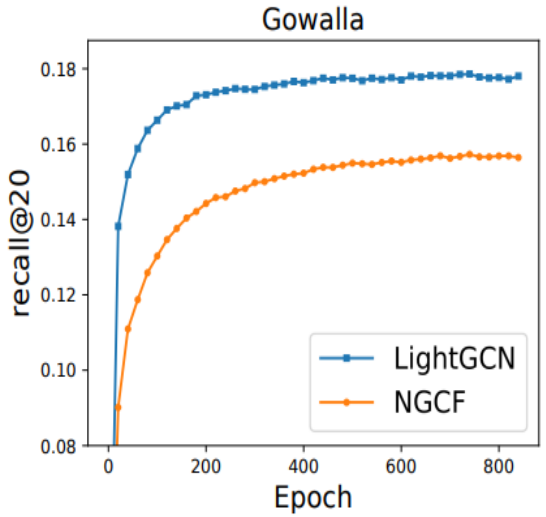
$$\mathbf{e}_u^{(2)} = \sum_{i \in \mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}} \mathbf{e}_i^{(1)} = \sum_{i \in \mathcal{N}_u} \frac{1}{|\mathcal{N}_i|} \sum_{v \in \mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_v|}} \mathbf{e}_v^{(0)}. \tag{13}$$

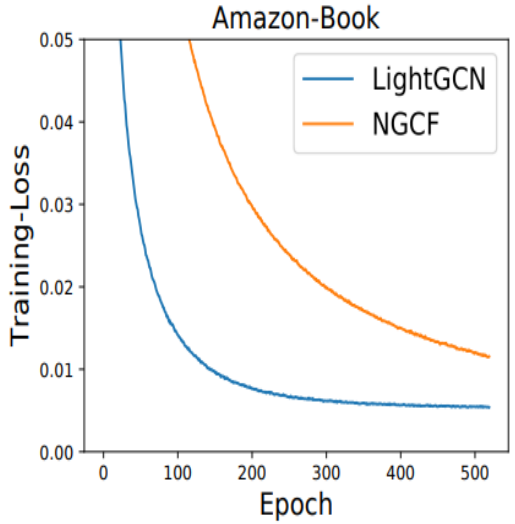
$$c_{v->u} = \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_v|}} \sum_{i \in \mathcal{N}_u \cap \mathcal{N}_v} \frac{1}{|\mathcal{N}_i|}.$$

User v가 target user u와 같은 item에 상호 작용한 기록이 있다면
두 유저간 smoothness 강도는 coefficient로 표현

Experiment







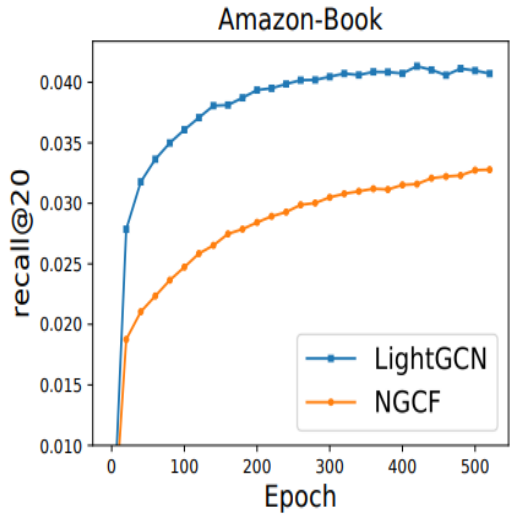
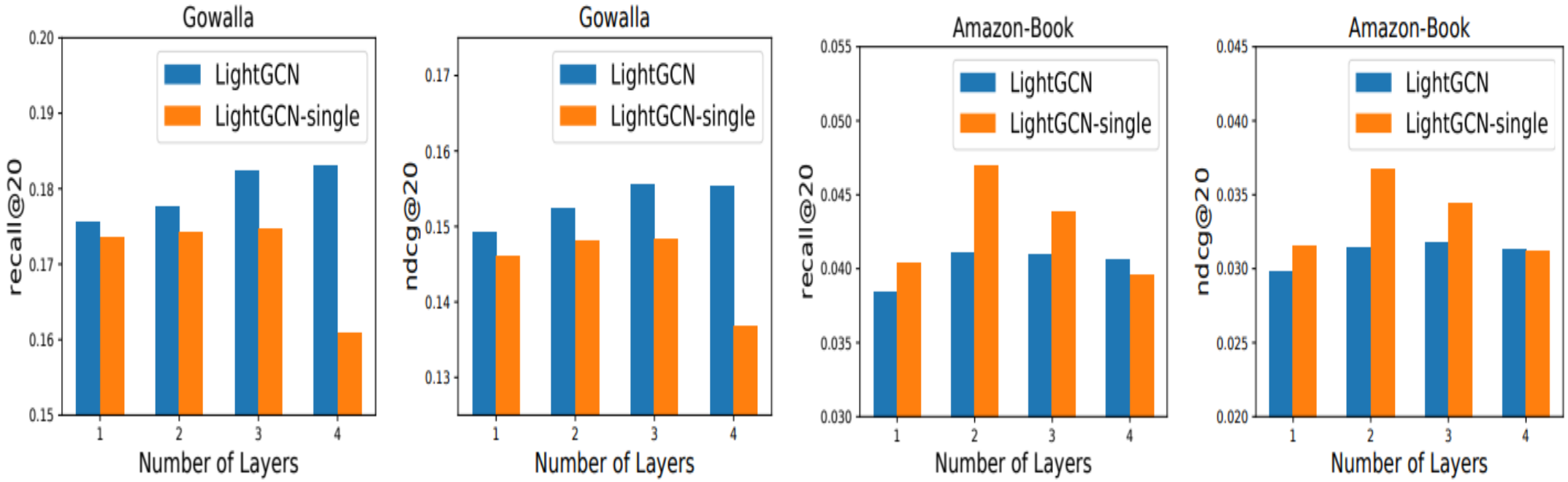


Table 4: The comparison of overall performance among LightGCN and competing methods.

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**

Experiment



・LightGCN – single : 마지막 Layer의 output만을 최종 임베딩 값으로 사용

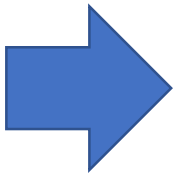
Conclusion

▷ Strong points

NGCF 구조를 단순화하여 추천 성능 향상 및 효율성 개선

▷ Weak points

LightGCN은 암시적 데이터를 통한 추천 시스템이므로 선호도에 대한 평가를 반영하지 못한다



LightGCN의 간소화된 구조를 가지고 가면서 명시적 데이터에 적용할 수 있도록 확장할 수 없을까?

Thank you!