Paper Review

Simplifying and Powering Graph Convolution Network for Recommendation

(LightGCN)

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, Meng Wang

2020 (SIGIR '20)

HTET ARKAR

School of Computer Science and Engineering

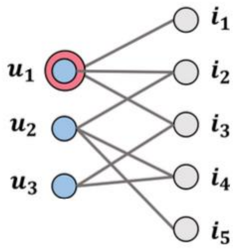
Chung-Ang University

Content

- Part I
  - Introduction
  - Proposed Method I
  - Methodology
- Part II
  - Problem
  - Ablation Study
  - Proposed Method II
  - Experiment
  - Conclusion

NGCF – Problem

- High-hop neighbors로 sub-graph 구조의 사용을 심화한 모델 (GCN으로부터 발전된 모델)
- **Ablation Study 결과 두가지를 발견**
  - Feature transformation과 nonlinear activation이 성능에 오히려 악영향을 끼치고 있음
  - 제거 후 상당한 성능 향상으로 이어짐
- Collaborative filtering에서는 user-item 사이의 one-hot ID로만 설명됨
  - Sematic 한 정보가 없어서 성능을 저하함



User-Item Interaction Graph

Solution

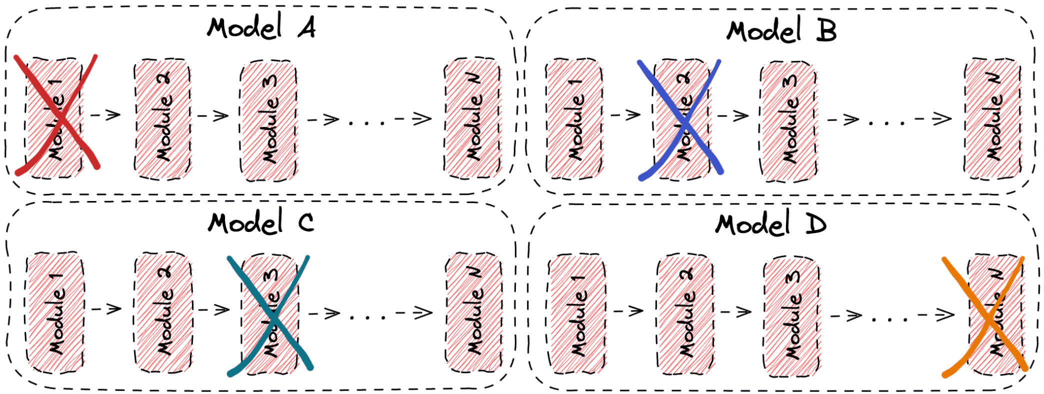
GCN을 단순화하여 추천에 더 간결하고 적합하게 하는 모델

GCN 에 가장 필수적인 neighborhood aggregation만을 사용

LightGCN

Ablation Study

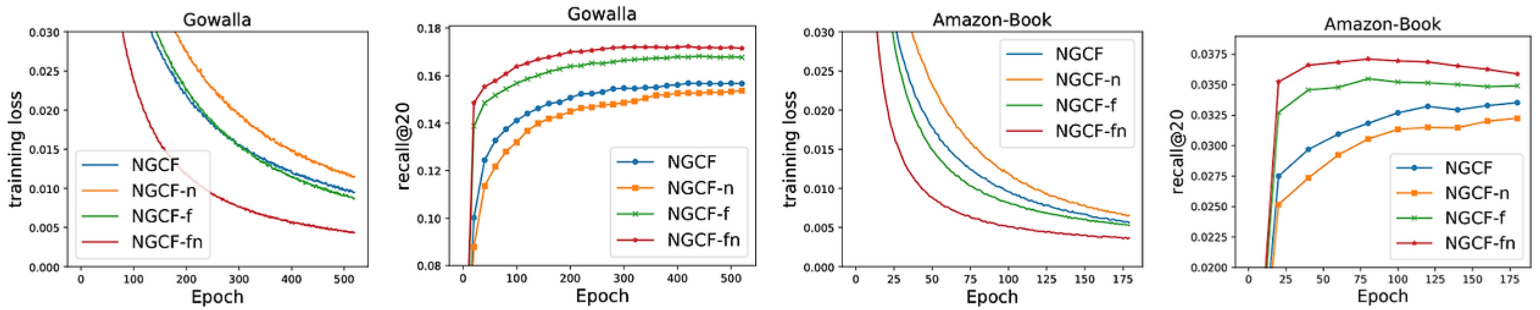
"Machine learning system의 building blocks을 제거해서 전체 성능에 미치는 효과에 대한 insight를 얻기 위한 과학적 실험"



Ablation Study

☐ NGCF-fn

- Such lower training loss successfully transfers to better recommendation accuracy



(a) Training loss on Gowalla (b) Testing recall on Gowalla (c) Training loss on Amazon-Book (d) Testing recall on Amazon-Book

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

- ☐ NGCF-f : removing the feature matrices, W_1 and W_2
- ☐ NGCF-n : removing nonlinear activation function, $\sigma(\cdot)$
- ☐ NGCF-fn : removing both

Proposed Method

□ LightGCN

- Feature transformations, nonlinear activation, self-connection을 제거함
- Layer Combination을 통해 유저와 아이템의 점수를 계산함
- 유저가 구매하지 않은 아이템 중 상위의 점수에 있는 k개의 아이템을 유저에게 추천

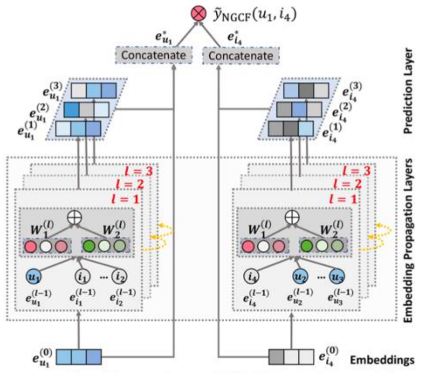
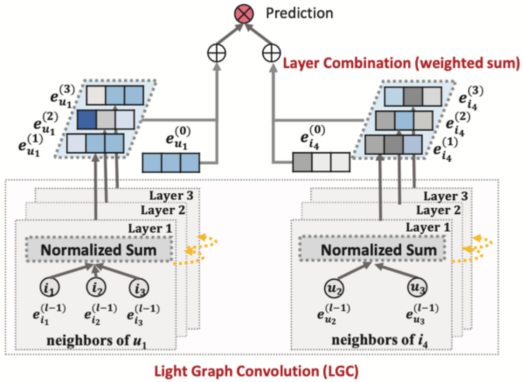


Figure 2: An illustration of NGCF model architecture



Proposed Method

LightGCN

- Performing two essential components
  - (1) Light graph convolution
    - Adopting simple weighted sum aggregator

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

  - (2-1) Layer combination to get final representations

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

    - $\alpha_k \geq 0$: hyper-parameter/ model parameter (here – setting uniformly : 1/(K + 1))

  - (2-1) Model Prediction -> $\hat{y}_{ui} = \mathbf{e}_u^T \mathbf{e}_i$ (used as ranking score)

LightGCN

- Layer combination한 결과를 사용하는 이유
  - 레이어 수가 늘어나면 임베딩들이 over-smoothing 됨
    - 마지막 layer만을 사용하는 것은 문제가 존재
  - 포괄적인(comprehensive) representation을 추출할 수 있음
    - 각각의 layer에서 서로 다른 semantic을 포착한
      - First layer – Smoothness on users and items that have interactions
      - Second layer – Smoothness on users(items) that have overlap on interacted items(user)
  - Self-connected의 효과를 포착할 수 있음
    - 서로 다른 layer의 embedding을 가중합(weighted sum)을 통해 결합함으로써

LightGCN

☐ Matrix form of LightGCN

- user-item interaction matrix : $\mathbf{R} \in \mathrm{R}^{M \times N}$
- Adjacency matrix : $A = \begin{bmatrix} \mathbf{0} & \mathbf{R} \\ \mathbf{R}^T & \mathbf{0} \end{bmatrix}$
- $E^{(0)} \in \mathrm{R}^{(M+N) \times T}$ (T : embedding size)
- $\mathbf{E}^{(k+1)} = (\mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}) \mathbf{E}^{(k)}$, **D** : *(M+N) x (M+N) Degree matrix*
- Final embedding matrix : $\mathbf{E} = \alpha_0 \mathbf{E}^{(0)} + \alpha_1 \mathbf{E}^{(1)} + \alpha_2 \mathbf{E}^{(2)} + \ldots + \alpha_K \mathbf{E}^{(K)}$
  $= \alpha_0 \mathbf{E}^{(0)} + \alpha_1 \tilde{\mathbf{A}} \mathbf{E}^{(0)} + \alpha_2 \tilde{\mathbf{A}}^2 \mathbf{E}^{(0)} + \ldots + \alpha_K \tilde{\mathbf{A}}^K \mathbf{E}^{(0)}$
  - ☐ $\tilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}$: Systematically normalized matrix

LightGCN

- Self-connection in SGCN (Simplified GCN)
  - By removing nonlinearities and collapsing weight matrices to one weight matrix

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

    - $\mathbf{I} \in R^{(M+N)\times(M+N)}$: identity matrix *(added on A to include self-connections)*
    - $(D + I)^{-1/2}$ terms for simplicity, since they only re-scale embeddings.

$$\begin{aligned}\mathbf{E}^{(K)} &= (\mathbf{A}+\mathbf{I})\mathbf{E}^{(K-1)} = (\mathbf{A}+\mathbf{I})^{K}\mathbf{E}^{(0)} \\ &= \binom{K}{0}\mathbf{E}^{(0)} + \binom{K}{1}\mathbf{A}\mathbf{E}^{(0)} + \binom{K}{2}\mathbf{A}^{2}\mathbf{E}^{(0)} + \ldots + \binom{K}{K}\mathbf{A}^{K}\mathbf{E}^{(0)}\end{aligned}$$

- LightGCN fully recovers the self-connection effect by layer combination

LightGCN

- **Alleviate Over-smoothing (APPNP)**
  - Connecting GCN with personalized PageRank
  - Propagating long range without the risk of over-smoothing

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

$$\begin{aligned}
\mathbf{E}^{(K)} &= \beta\mathbf{E}^{(0)} + (1-\beta)\tilde{\mathbf{A}}\mathbf{E}^{(K-1)}, \\
&= \beta\mathbf{E}^{(0)} + \beta(1-\beta)\tilde{\mathbf{A}}\mathbf{E}^{(0)} + (1-\beta)^2\tilde{\mathbf{A}}^2\mathbf{E}^{(K-2)} \\
&= \beta\mathbf{E}^{(0)} + \beta(1-\beta)\tilde{\mathbf{A}}\mathbf{E}^{(0)} + \beta(1-\beta)^2\tilde{\mathbf{A}}^2\mathbf{E}^{(0)} + \ldots + (1-\beta)^K\tilde{\mathbf{A}}^K\mathbf{E}^{(0)}
\end{aligned}$$

- **LightGCN shares the strength of APPNP in combination over-smoothing**

LightGCN

□ Model Training

- ■ Trainable parameter : only the embeddings of the 0-th layer
- ■ *Bayesian Personalized Ranking (BPR)* loss 를 사용

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

- □ A pairwise loss
- □ Observed/unobserved user-item interaction 사이의 상대적 우선순위 고려
- □ 유자의 선호를 더 반영하는 observed interaction 에 unobserved interaction 보다 높은 점수 부여

Experiments

- LightGCN closely follows the setting of the NGCF work

Table 3: Performance comparison between NGCF and LightGCN at 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%)	0.0530(+15.02%)	0.0406(+17.92%)	0.0313(+18.92%)

*The scores of NGCF on Gowalla and Amazon-Book are directly copied from Table 3 of the NGCF paper (https://arxiv.org/abs/1905.08108)

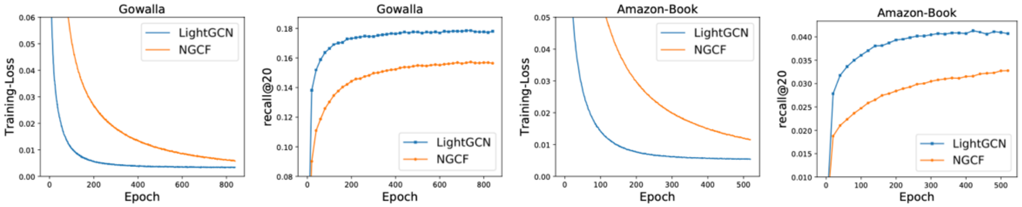


Figure 3: Training curves of LightGCN and NGCF, which are evaluated by training loss and testing recall per 20 epochs on Gowalla and Amazon-Book (results on Yelp2018 show exactly the same trend which are omitted for space).

- Increasing the # of layers can improve the performance of LightGCN
- LightGCN obtains lower training loss, but transfers to better testing accuracy

Experiments

- Performance comparison with other SOTA

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 consistently outperforms other methods on all data sets
- Hight effectiveness with simple yet reasonable designs

Experiments

- Comparison of LightGCN and LightGCN-single

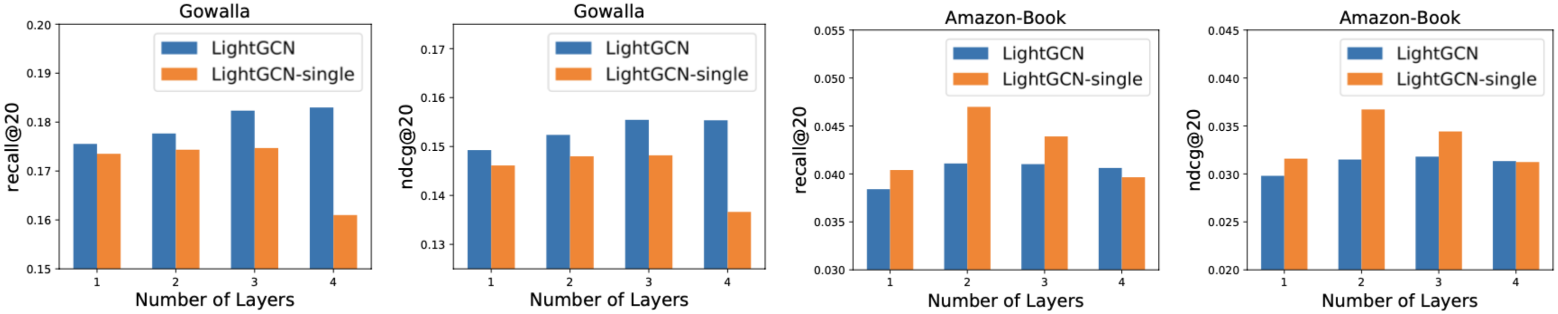


Figure 4: Results of LightGCN and the variant that does not use layer combination (i.e., LightGCN-single) at different layers on Gowalla and Amazon-Book (results on Yelp2018 shows the same trend with Amazon-Book which are omitted for space).

- Layer combination의 효과로 over-smoothing 문제를 잘 해결한다고 판단 할 수 있음
  - LightGCN-single : Layers의 수가 증가할수록 모델의 성능이 떨어짐 (over-smoothing이 발생)
  - LightGCN : Layers의 수가 증가할수록 일관성 있게 성능이 좋아짐

Conclusion

- **Problem**
  - Unnecessarily complicated design of GCNs for collaborative filtering
- **Solution**
  - LightGCN – beign simple
    - consists of two essential components
      - Light graph convolution
        - Discarding feature transformation and nonlinear activation
      - layer combination
        - Recovering the effect of self-connection and helpful to control over-smoothing

HTET ARKAR (hak3601@cau.ac.kr)