MixGCF: An Improved Training Method for Graph Neural Network-based Recommender Systems

Tinglin Huang[†★], Yuxiao Dong[‡], Ming Ding[◇], Zhen Yang[◇], Wenzheng Feng[◇]
Xinyu Wang[†], Jie Tang[◇§]
[†]Zhejiang University, [‡]Facebook AI, [◇]Tsinghua University
tinglin.huang@zju.edu.cn,yuxiaod@fb.com,dm18@mails.tsinghua.edu.cn,zheny2751@gmail.com
fwz17@mails.tsinghua.edu.cn,wangxinyu@zju.edu.cn,jietang@tsinghua.edu.cn

Motivation

- The negative samples paly a decisive role in the performance of (GNN-based) recommendation models. However, existing works (e.g., PinSage, MCNS) only focus on improving negative sampling in the discrete graph space, ignoring GNN's unique neighborhood aggregation process in the embedding space.

- MixGCF synthesizes negative samples rather than directly sampling negatives from the data for improving GNN-based recommender systems.

Optimization with Negative Sampling

- For learning to rank task, we often assume that users prefer the observed (positive) items over all unobserved (negative) ones. Due to the large size of unobserved items, the learning objective is usually simplified by negative sampling as the BPR loss.

$$\max \prod_{v^+, v^- \sim f_S(u)} P_u(v^+ > v^- | \Theta)$$

where v^+ and v^- denote the positive and negative items, respectively, $P_u(a > b)$ represents user u prefers item a over b, $f_S(u)$ is the distribution of negative sampling.

An overview of MixGCF

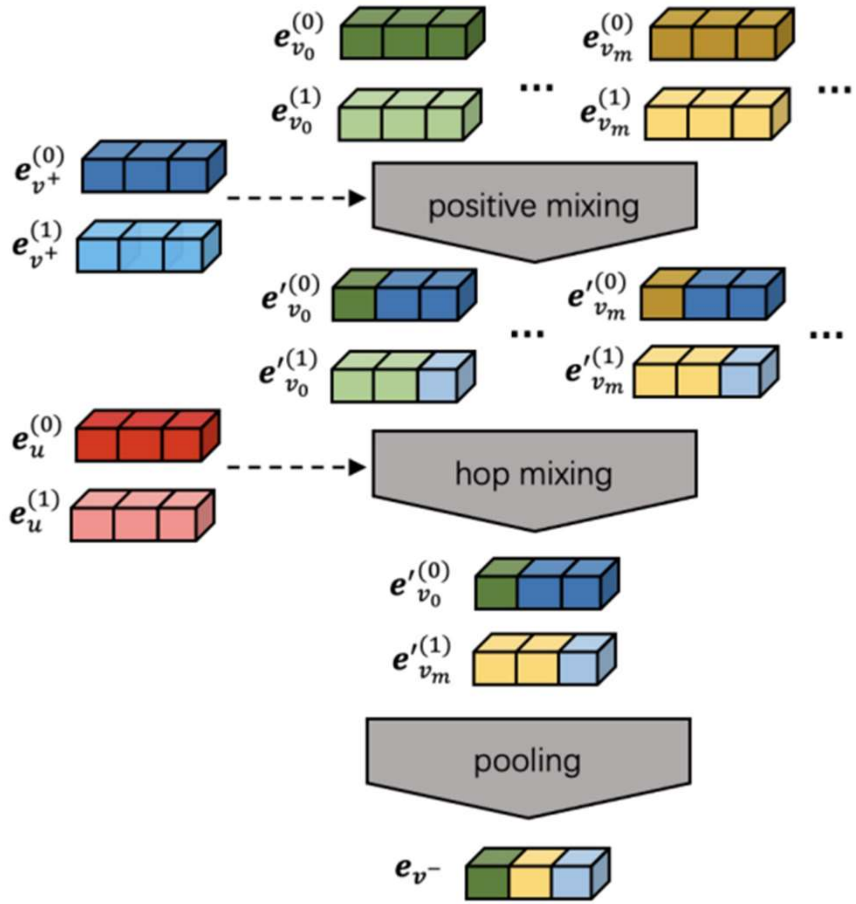


Figure 2: An overview of MixGCF, where $e^{(l)}$ denotes the l-th layer embedding of node e, and $e'^{(l)}$ denotes the l-th layer embedding generated by positive mixing.

Positive Mixing

- For each pair (u, v^+), we could selected M candidate negative samples by uniformly sampling. Denote $\mathcal{E} = \left\{e_{v_m}^{(l)}\right\}$ as the candidate negative embedding set of size $M \times (L+1)$. $e_v^{(l)}$ is the item v embedding in layer l. Thus, the positive mixing operation is formalized as:

$$e_{v_m}^{\prime(l)} = \alpha^{(l)} e_{v^+}^{(l)} + \left(1 - \alpha^{(l)}\right) e_{v_m}^{(l)}, \alpha^{(l)} \in (0,1)$$

where $\alpha^{(l)}$ is the mixing coefficient, which can be uniformly sampled from (0, 1) . Thus, we could obtain corresponding candidate negative items embedding $\mathcal{E}' = \left\{e_{v_m}^{\prime(l)}\right\} \in \mathbb{R}^{M \times (L+1)}$ enhanced by positive mixing.

Hop Mixing

- For layer l, hop mixing operation is to sample one candidate negative embedding $e'^{(l)}_{v_x}$ $(1 \leq x \leq M)$ from $\mathcal{E}'^{(l)}$, which contains all the l-th layer embedding of the candidate negative items in M.

$$e'^{(l)}_{v_x} = \underset{e'^{(l)}_{v_m} \in \mathcal{E}'}{\arg\, max}\, f_Q(u, l) \cdot e'^{(l)}_{v_m}$$

where $\cdot$ is the inner product, $f_Q(u, l)$ is a query mapping that returns an embedding related to the target user u for the l-th hop. For sum based pooling, $f_Q(u, l) = e_u$. For concat based pooling, $f_Q(u, l) = e^{(l)}_u$. e_u is the user embedding.

Hop Mixing

- The idea of hop mixing is then to combine all the L+1 embeddings $e_{v_m}^{\prime(l)}$ selected by layer to generate the representation e_{v^-} of the (fake) negative v^- via pooling operation

$$e_{v^-} = f_{pool}(e_{v_x}^{\prime(0)}, \dots, e_{v_y}^{\prime(L)})$$

where $e_{v_x}^{\prime(0)}$ denotes the l-th layer embedding of v_x that is sampled at layer l, and $f_{pool}(\cdot)$ is the pooling operation, which can be sum-based pooling or concat-based pooling.

Optimization with MixGCF

- The BRP loss function is applied for MixGCF optimization

$$\mathcal{L}_{BRP} = \sum_{\substack{(u,v^+)\in\mathcal{O}^+ \\ e_{v^-}\sim f_{MixGCF}(u,v^+)}} ln\sigma(e_u \cdot e_{v^-} - e_u \cdot e_{v^+})$$

where $\sigma(\cdot)$ is the sigmoid function, $\mathcal{O}^+$ is the set of the positive feedback, and $e_{v^-}\sim f_{MixGCF}(u, v^+)$ represents that the instance (embedding) e_{v^-} is synthesized by the proposed MixGCF method.

For sum-based pooling, $e_u \cdot e_{v^-} = \sum_{l=0}^{L} \lambda\, e_u \cdot e_{v^-}^{l}$.

For concat-based pooling, $e_u \cdot e_{v^-} = \sum_{l=0}^{L} e_u^{(l)} \cdot e_{v^-}^{l}$.

The training process with MixGCF

Algorithm 1: The training process with MixGCF

Input: Training set $\{(u, v^+)\}$, Recommender f_{GNN}, Number of negative candidate M, Number of aggregation layers L.

for $t = 1, 2, \cdots, T$ **do**
    Sample a mini-batch of positive pairs $\{(u, v^+)\}$.
    Initialize loss $\mathcal{L} = 0$.
    // Negative Sampling via MixGCF.
    for *each* (u, v^+) *pair* **do**
        Get the aggregated embeddings of each node by f_{GNN}.
        Get the set of candidate negative embeddings $\mathcal{E}$ by uniformly sampling M negatives.
        Get the updated set of negative candidate $\mathcal{E}'$ by (5).
        Synthesize a hard negative $\mathbf{e}_{v^-}$ based on $\mathcal{E}'$ by (6).
        $\mathcal{L} = \mathcal{L} + \ln \sigma(\mathbf{e}_u \cdot \mathbf{e}_{v^-} - \mathbf{e}_u \cdot \mathbf{e}_{v^+})$.
    end
    Update θ by descending the gradients $\nabla_\theta \mathcal{L}$.
end

$$e'^{(l)}_{v_m} = \alpha^{(l)} e^{(l)}_{v^+} + \left(1 - \alpha^{(l)}\right) e^{(l)}_{v_m}, \alpha^{(l)} \in (0,1) \quad (5)$$

$$e_{v^-} = f_{pool}\left(e'^{(0)}_{v_x}, \ldots, e'^{(L)}_{v_y}\right) \quad (6)$$

Experiments

Table 2: Overall Performance Comparison.

	Alibaba		Yelp2018		Amazon	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
LightGCN+RNS	0.0584	0.0275	0.0628	0.0515	0.0398	0.0177
LightGCN+DNS	0.0737	0.0343	0.0695	0.0571	0.0449	0.0211
LightGCN+IRGAN	0.0605	0.0280	0.0641	0.0527	0.0412	0.0185
LightGCN+AdvIR	0.0583	0.0273	0.0624	0.0510	0.0401	0.0185
LightGCN+MCNS	0.0632	0.0284	0.0658	0.0529	0.0423	0.0192
LightGCN+MixGCF	**0.0763***	**0.0357***	**0.0713***	**0.0589***	**0.0460***	**0.0216***
NGCF+RNS	0.0426	0.0197	0.0577	0.0469	0.0294	0.0123
NGCF+DNS	0.0453	0.0207	0.0650	0.0529	0.0312	0.0130
NGCF+IRGAN	0.0435	0.0200	0.0615	0.0502	0.0283	0.0120
NGCF+AdvIR	0.0440	0.0203	0.0614	0.0500	0.0318	0.0134
NGCF+MCNS	0.0430	0.0200	0.0625	0.0501	0.0313	0.0136
NGCF+MixGCF	**0.0544***	**0.0262***	**0.0688***	**0.0566***	**0.0350***	**0.0154***
PinSage+RNS	0.0196	0.0085	0.0410	0.0328	0.0193	0.0080
PinSage+DNS	0.0405	0.0183	0.0590	0.0488	0.0217	0.0088
PinSage+IRGAN	0.0200	0.0090	0.0422	0.0343	0.0248	0.0088
PinSage+AdvIR	0.0196	0.0090	0.0387	0.0313	0.0243	0.0087
PinSage+MCNS	0.0212	0.0095	0.0432	0.0349	0.0202	0.0088
PinSage+MixGCF	**0.0489***	**0.0226***	**0.0632***	**0.0525***	**0.0273***	**0.0124***

Experiments

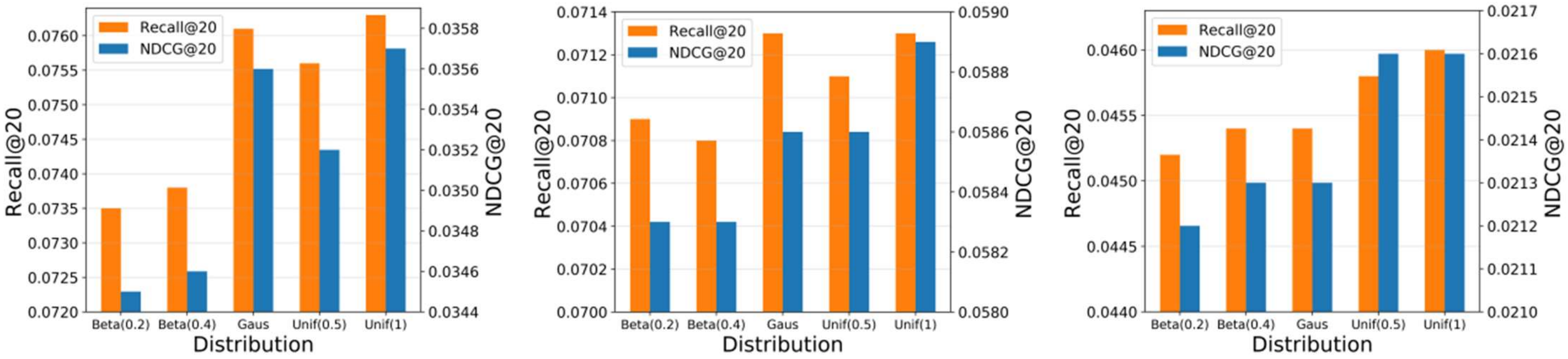
Table 4: Impact of the number of aggregation modules (L).

	Alibaba		Yelp2018		Amazon	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
LightGCN+MixGCF-1	0.0651	0.0309	0.0684	0.0564	0.0403	0.0193
LightGCN+MixGCF-2	0.0726	0.0335	0.0707	0.0582	0.0438	0.0209
LightGCN+MixGCF-3	0.0763	0.0357	0.0713	0.0589	0.0460	0.0216
NGCF+MixGCF-1	0.0484	0.0234	0.0647	0.0526	0.0320	0.0151
NGCF+MixGCF-2	0.0545	0.0262	0.0664	0.0542	0.0345	0.0153
NGCF+MixGCF-3	0.0544	0.0262	0.0688	0.0566	0.0350	0.0154
PinSage+MixGCF-1	0.0487	0.0231	0.0639	0.0526	0.0289	0.0130
PinSage+MixGCF-2	0.0472	0.0223	0.0627	0.0519	0.0278	0.0121
PinSage+MixGCF-3	0.0489	0.0226	0.0632	0.0525	0.0273	0.0124

Table 5: Impact of the size of candidate set (M).

		Alibaba		Yelp2018		Amazon	
		Recall	NDCG	Recall	NDCG	Recall	NDCG
LightGCN+MixGCF	M=8	0.0697	0.0311	0.0664	0.0547	0.0443	0.0203
	M=16	0.0728	0.0339	0.0684	0.0562	**0.0460***	**0.0216***
	M=32	**0.0763***	**0.0357***	0.0703	0.0579	0.0455	0.0215
	M=64	0.0744	0.0355	**0.0713***	**0.0589***	0.0430	0.0206
NGCF+MixGCF	M=8	0.0468	0.0201	0.0627	0.0512	0.0350	0.0147
	M=16	0.0518	0.0237	0.0658	0.0539	0.0333	0.0144
	M=32	0.0532	0.0253	0.0682	0.0560	0.0347	0.0154
	M=64	**0.0544***	**0.0262***	**0.0688***	**0.0566***	**0.0350***	**0.0154***
PinSage+MixGCF	M=8	0.0178	0.0075	0.0495	0.0402	0.0204	0.0072
	M=16	0.0388	0.0173	0.0546	0.0448	0.0207	0.0084
	M=32	0.0435	0.0195	0.0608	0.0501	0.0238	0.0106
	M=64	**0.0489***	**0.0226***	**0.0632***	**0.0525***	**0.0273***	**0.0124***

Experiments



(a) Alibaba (b) Yelp2018 (c) Amazon

Figure 4: Performance comparison over different distributions, i.e., Beta, Gaussian and Uniform distribution, of random coefficient ($\alpha^{(l)}$) on three datasets.