SIGIR 2023 - Reproducibility of Recommendation Systems based on message passing - Additional Material

This is the additional material associated with our submitted paper. This material contains the full results of our experiments of which, due to space reasons and for the sake of improving readability, only the most representative ones are reported in the paper. The results for each of the papers we analyze are reported in separate sections. The results for the ported methods are labeled with: *no early-stopping* if it uses the the optimal number of epochs reported in the original paper; *original early-stopping* if it uses the early-stopping criteria described in the original paper (e.g., evaluating every epoch and a patience of 50 epochs); *our early-stopping* if it uses our own early-stopping criteria, evaluating the model every 5 epochs and stopping if for 5 consecutive evaluation the model does not improve.

ACM Reference Format:
. 2023. SIGIR 2023 - Reproducibility of Recommendation Systems based on message passing - Additional Material. 1, 1 (March 2023), 32 pages. https://doi.org/0000001.0000001

Contents

Author's address:

XXXX-XXXX/2023/3-ART $15.00
https://doi.org/0000001.0000001

1 LESS IS MORE: REWEIGHTING IMPORTANT SPECTRAL GRAPH FEATURES FOR RECOMMENDATION

Peng et al. [4] analyzes the spectral properties of Graph Convolutional Networks and observe that the frequencies (i.e., eigenvalues) that contribute the most to the recommendation accuracy are both the highest and lowest ones, with the intermediate ones being less important. This effect is attributed to the different semantics of the two, with higher frequencies representing differences between users while the lower ones representing the commonalities. The article proposes *Graph Denoising Encoder* (GDE) which acts as a band-pass filter selecting high and low frequencies while removing intermediate ones. The proposed method is claimed to be substantially faster compared to LightGCN. The source code is publicly available[1].

1.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 1. All existing interactions are made implicit and assigned a value of 1.

Movielens: Is a movie recommendation dataset, the explicit ratings (1-5) are all transformed in implicit ratings of value 1.

CiteULike-a: Is a dataset collected from CiteULike, which is an online service providing users with a digital catalogue to save and share academic papers. If the user has saved the article in their library it will be associated to a rating of 1.

Pinterest: refers to the well known social network which allows users to save or pin an image to their board. If a user has pinned an image on the board it will be associated to a rating of 1.

Gowalla: A dataset of locations where a user can check in if hey have visited a certain location.

Dataset	Interactions	Items	Users	Sparsity
citeulike	210504	16980	5551	9.978E-01
gowalla	1027370	40981	29858	9.992E-01
movielens1m	1000209	3952	6040	9.581E-01
movielens100k	100000	1682	943	9.370E-01
pinterest	1000154	9836	37501	9.973E-01

Table 1. Dataset characteristics for GDE.

[1] https://github.com/tanatosuu/GDE

Hyperparameter	Described in	Value					
		All datasets	CiteULike	ML-1M	ML-100K	Pinterest	Gowalla
Embedding size	Paper	64	-	-	-	-	-
Regularization rate	Paper	0.01	-	-	-	-	-
Learning rate	Source code	0.03	0.02	7.5	2.0	0.85/0.12	0.03
Dropout rate	Source code	0.1	0.3	0.5	0.2	0.2	0.1
Epochs	Source code	400	200	90	50	>200	160
Batch size	Paper	256	-	-	-	-	-
β	Source code	-	5.0[a]	4.0	4.0[b]	4.0/5.0	5.0
Loss type	Source code	adaptive	adaptive	adaptive	bpr	adaptive	adaptive
Smooth ratio	Source code	0.1	0.3	0.05	0.2	0.2	0.1
Rough ratio	Source code	0.0	0.0	0.005	0.002	0.0	0.0
Feature type[c]	Source code	smoothed	smoothed	both	both	smoothed	smoothed

[a]The paper reports the optimal value should be 4.5
[b]The paper reports the optimal value should be 4.5
[c]If "smoothed" only the smooth features are used, if "both" rough features are used as well.

Table 2. Hyperparameter Values for GDE.

Table 3. Experimental results for the GDE method for the Citeulike dataset.

	@ 20	
	PREC REC	NDCG
TopPop	0.0525	0.0544
UserKNN CF cosine	**0.1003**	**0.1131**
ItemKNN CF cosine	**0.0997**	**0.1121**
RP3beta	**0.1028**	**0.1151**
GraphFilter CF	0.0973	0.1006
EASE R	0.0981	**0.1099**
SLIM ElasticNet	**0.1000**	**0.1116**
NegHOSLIM ElasticNet	0.0983	**0.1104**
MF BPR	0.0316	0.0371
IALS	**0.1143**	**0.1240**
GDE paper	**0.1224**	**0.1339**
GDE	0.0570	0.0551
GDE no earlystopping	0.0015	0.0014
GDE hyperopt	0.0991	0.1086

Table 4. Experimental results for the GDE method for the Movielens 1M dataset.

	@ 20	
	PREC REC	NDCG
TopPop	0.3838	0.4062
UserKNN CF cosine	0.4876	0.5184
ItemKNN CF cosine	0.4493	0.4830
RP3beta	0.5072	0.5361
GraphFilter CF	0.5247	0.5537
EASE R	0.4780	0.5062
SLIM ElasticNet	0.4644	0.4950
NegHOSLIM ElasticNet	0.4612	0.4930
MF BPR	0.2848	0.2918
IALS	0.5147	0.5415
GDE paper	**0.5423**	**0.5715**
GDE	0.5357	0.5658
GDE no earlystopping	0.5356	0.5636
GDE hyperopt	0.5291	0.5564

Table 5. Experimental results for the GDE method for the Movielens 100k dataset.

	@ 20	
	PREC REC	NDCG
TopPop	0.4062	0.4292
UserKNN CF cosine	0.4912	0.5281
ItemKNN CF cosine	0.4289	0.4644
RP3beta	0.4820	0.5207
GraphFilter CF	0.4421	0.4747
EASE R	0.4567	0.4972
SLIM ElasticNet	0.4689	0.5047
NegHOSLIM ElasticNet	0.4654	0.4979
MF BPR	0.3905	0.4141
IALS	0.4194	0.4370
GDE paper	**0.5400**	**0.5731**
GDE	0.5196	0.5515
GDE no earlystopping	0.5293	0.5585
GDE hyperopt	0.4229	0.4516

Table 6. Experimental results for the GDE method for the Pinterest dataset.

	@ 20	
	PREC REC	NDCG
TopPop	0.0174	0.0181
UserKNN CF cosine	0.0879	0.0948
ItemKNN CF cosine	0.0877	0.0944
RP3beta	0.0872	0.0941
GraphFilter CF	0.1007	0.1081
EASE R	0.0831	0.0898
SLIM ElasticNet	0.0852	0.0919
NegHOSLIM ElasticNet	0.0851	0.0917
MF BPR	0.0654	0.0699
IALS	0.1067	0.1146
GDE paper	**0.1147**	**0.1240**
GDE	0.0026	0.0024
GDE no earlystopping	0.0026	0.0024
GDE hyperopt	0.1082	0.1171

Table 7. Experimental results for the GDE method for the Gowalla dataset.

	@ 20	
	PREC REC	NDCG
TopPop	0.0421	0.0451
UserKNN CF cosine	0.1128	0.1304
ItemKNN CF cosine	0.1119	0.1288
RP3beta	0.1116	0.1285
GraphFilter CF	-	-
EASE R	-	-
SLIM ElasticNet	0.1057	0.1219
NegHOSLIM ElasticNet	0.1053	0.1214
MF BPR	0.0299	0.0319
IALS	0.1361	0.1531
GDE paper	**0.1449**	**0.1632**
GDE	0.0959	0.1077
GDE no earlystopping	0.1433	0.1627
GDE hyperopt	0.1282	0.1476

2 ARE GRAPH AUGMENTATIONS NECESSARY? SIMPLE GRAPH CONTRASTIVE LEARNING FOR RECOMMENDATION

Yu et al. [11] propose *Simple Graph Contrastive Learning* (SimGCL). The paper claims that in constrastive learning based recommendations the main contribution to the recommendation quality is not the graph augmentation (random edge dropout) but rather the constrastive learning loss function (InfoNCE). The InfoNCE loss effect is to increase the separation between positive and negative samples for each user. SimGCL uses random perturbations of the embeddings instead of graph augmentations. In practice, SimCL is a LightGCM [3] with random embedding perturbations, a regularizing loss and the aggregated user and item embeddings that start from layer 1 (excluding E0).

The original implementation is available on Github[2]

2.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 8

Douban Book: relations for the Douban book service, ratings are in the range 1-5. Ratings greater or equal to 4 are transformed in implicit interactions with value 1, the other ratings are removed.
Yelp2018: from LightGCN [3]
Amazon-Book: from LightGCN [3]

Dataset	Interactions	Items	Users	Sparsity
amazon book	2984108	91599	52643	9.994E-01
douban book	598420	22348	13025	9.979E-01
yelp2018	1561406	38048	31668	9.987E-01

Table 8. Dataset characteristics for SimGCL.

[2]https://github.com/Coder-Yu/QRec we use the pytorch implementation available from the authors here https://github.com/Coder-Yu/SELFRec

Hyperparameter	Described in	Value			
		All datasets	Douban Book	Yelp2018	Amazon-Book
λ (contrastive loss weight)	Paper[a]	-	0.2	0.5	2
τ (contrastive loss temperature)	Paper	0.2	-	-	-
ϵ (noise magnitude)	Paper[b]	0.1	-	-	-
Batch size	Paper	2048	-	-	-
Number of layers	Paper[c]	3	-	-	-
Learning rate	Paper	10^{-3}	-	-	-
Adaptive gradient	Paper	Adam	-	-	-
Embedding size	Paper	64	-	-	-
L_2 regularization	Paper	10^{-4}	-	-	-
Epochs	Paper[d]	-	25	11	10

[a]From a section discussing hyperparameter sensitivity
[b]From a section discussing hyperparameter sensitivity
[c]From a table comparing the result for different number of layers.
[d]From a section that discusses a plot showing when the models converge with Recall and BPR loss.

Table 9. Hyperparameter Values for SimGCL.

Table 10. Experimental results for the SimGCL method for the Amazon Book Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0051	0.0044
UserKNN CF cosine	**0.0616**	**0.0518**
ItemKNN CF cosine	**0.0741**	**0.0617**
RP3beta	**0.0750**	**0.0608**
GraphFilter CF	**0.0710**	**0.0585**
EASE R	-	-
SLIM ElasticNet	**0.0756**	**0.0600**
NegHOSLIM ElasticNet	**0.0737**	**0.0607**
MF BPR	0.0281	0.0220
IALS	0.0426	0.0342
SimGCL paper	0.0515	**0.0515**
SimGCL	0.0531	0.0421
SimGCL no earlystopping	0.0518	0.0418

Table 11. Experimental results for the SimGCL method for the Doubanbook Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0722	0.0582
UserKNN CF cosine	0.1686	**0.1575**
ItemKNN CF cosine	**0.1972**	**0.1908**
RP3beta	**0.2033**	**0.1841**
GraphFilter CF	**0.1788**	**0.1604**
EASE R	-	-
SLIM ElasticNet	**0.2250**	**0.2226**
NegHOSLIM ElasticNet	**0.1971**	**0.1833**
MF BPR	0.0916	0.0774
IALS	**0.1833**	**0.1668**
SimGCL paper	**0.1772**	**0.1583**
SimGCL	0.1753	0.1551
SimGCL no earlystopping	0.1640	0.1451

Table 12. Experimental results for the SimGCL method for the Yelp 2018 Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0124	0.0101
UserKNN CF cosine	0.0638	0.0534
ItemKNN CF cosine	0.0643	0.0536
RP3beta	0.0670	0.0558
GraphFilter CF	0.0708	0.0583
EASE R	-	-
SLIM ElasticNet	0.0649	0.0543
NegHOSLIM ElasticNet	0.0622	0.0521
MF BPR	0.0392	0.0322
IALS	0.0652	0.0541
SimGCL paper	0.0721	**0.0721**
SimGCL	**0.0735**	0.0606
SimGCL no earlystopping	**0.0721**	0.0598

3 LEARNING TO DENOISE UNRELIABLE INTERACTIONS FOR GRAPH COLLABORATIVE FILTERING

Tian et al. [5] presents *Robust Graph Collaborative Filtering* (RGCF) based on the LighgGCN message passing architecture. RGCF consists of two steps, first a graph denoising module removes interactions that are estimated as being noisy and assigns a reliability weight to the other ones. This step is performed via the cosine similarity of the learned embeddings. Then, a diversity preserving module builds new interaction graphs (i.e., adjacency matrix) based on the denoised one. A certain number of random user-item candidates are sampled, the prediction computed using the learned embeddings and those with high score (the paper calls it reliability) are added to the interaction graph. The model is trained with BPR with a second loss added to pull the representation of nodes learned with the augmented graphs close to each other, this is done with the contrastive loss InfoNCE. The source code is publicly available[3].

3.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 13

Amazon Book: only users and items with at least 15 interactions are retained.
Movielens 1M: ratings ≥ 4 are transformed into implicit interactions with value 1.
Yelp: only users and items with at least 15 interactions are retained.

Dataset	Interactions	Items	Users	Sparsity
Amazon Book	2517437	58051	58144	9.993E-01
Movielens 1M	836478	3883	6040	9.643E-01
Yelp	1730025	31731	45160	9.988E-01

Table 13. Dataset characteristics for RGCF.

Hyperparameter	Described in	Value All datasets
epochs	source code	500
K	source code	2
batch size	Paper	4096
embedding size	Paper	64
prune threshold beta	source code	0.02
contrastive loss temperature tau	source code	0.2
contrastive loss weight	source code	1e-06
augmentation ratio	source code	0.1
learning rate	source code	4e-5
l2 reg	Paper	1e-05
optimizer	Paper	Adam

Table 14. Hyperparameter Values for RGCF.

[3] https://github.com/ChangxinTian/RGCF

Table 15. Experimental results for the RGCF method for the Movielens 1M dataset.

	@ 10			
	REC	NDCG	HR	MRR
TopPop	0.0773	0.1213	0.4894	0.2433
UserKNN CF cosine	0.1939	0.2711	0.7733	0.4741
ItemKNN CF cosine	0.1811	0.2578	0.7441	0.4610
RP3beta	0.1824	0.2557	0.7560	0.4577
GraphFilter CF	**0.2076**	**0.2885**	**0.7897**	**0.4944**
EASE R	**0.2128**	**0.3015**	**0.7970**	**0.5082**
SLIM ElasticNet	**0.2057**	**0.2944**	**0.7870**	**0.5034**
NegHOSLIM ElasticNet	**0.2125**	**0.3001**	**0.7958**	**0.5059**
MF BPR	0.1500	0.2105	0.6971	0.3894
IALS	0.1938	**0.2759**	0.7707	**0.4783**
RGCF paper	0.1986	0.2565	0.7569	0.4429
RGCF original earlystopping	0.1882	0.2635	0.7653	0.4642
RGCF ours earlystopping	0.1966	0.2693	0.7792	0.4733
RGCF hyperopt	0.1995	0.2738	0.7809	0.4776

4 INMO: A MODEL-AGNOSTIC AND SCALABLE MODULE FOR INDUCTIVE COLLABORATIVE FILTERING

Wu et al. [8] presents *Inductive Embedding Module for collaborative filtering* (INMO), that aims to improve the effectiveness of matrix factorization models to recommend to new users. The paper focuses on matrix factorization models that are *transductive* (i.e., not model based, such as FunkSVD, BPRMF etc..) and proposes a *inductive* (could we say it is model-based, to some extent?) representation of the user and item embeddings as a function of the embeddings of a selected subset of template user and items. INMO can in principle have fewer parameters. INMO includes an annealing process for a normalization hyperparameter. The source code is publicly available in a google drive folder linked to Github [4].

4.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 16

Amazon Book: ratings ≥ 4 are transformed into implicit interactions with value 1, then a 10-core is computed

Gowalla: no details provided

Yelp2018: ratings ≥ 4 are transformed into implicit interactions with value 1, then a 10-core is computed

Dataset	Interactions	Items	Users	Sparsity
Amazon Book	2780441	96421	109730	9.997E-01
Gowalla	900713	40988	29858	9.993E-01
Yelp2018	1680930	42706	75173	9.995E-01

Table 16. Dataset characteristics for INMO.

Hyperparameter	Described in	Value			
		All datasets	Amazon Book	Gowalla	Yelp2018
embedding size	Source code	64	-	-	-
batch size	Source code	2048	-	-	-
K	Source code	3	-	-	-
optimizer	Source code	Adam	-	-	-
epochs	Source code	1000 (max)	-	-	-
learning rate	Source code	10^{-3}	-	-	-
template loss weight	Source code	10^{-2}	-	-	-
λ_2	Source code	0.0	-	-	-
dropout rate	Source code	-	0.0	0.3	0.3
feature ratio	Source code	-	1.0	1.0	0.7
normalization decay	Source code	0.99	-	-	-
template node ranking	Source code	cardinality	-	-	-

Table 17. Hyperparameter Values for INMO.

[4] https://github.com/WuYunfan/igcn_cf

Table 18. Experimental results for the INMO method for the Gowalla dataset.

	@ 20 REC	PREC	NDCG
TopPop	0.0303	0.0083	0.0208
UserKNN CF cosine	0.1834	0.0493	0.1376
ItemKNN CF cosine	0.1908	0.0508	0.1431
RP3beta	**0.2029**	**0.0548**	**0.1523**
GraphFilter CF	**0.2014**	0.0525	**0.1483**
EASE R	-	-	-
SLIM ElasticNet	**0.2037**	**0.0574**	**0.1573**
NegHOSLIM ElasticNet	0.1934	**0.0526**	**0.1478**
MF BPR	0.1308	0.0350	0.0979
IALS	0.1820	0.0491	0.1362
INMO paper	**0.2017**	**0.0536**	**0.1541**
INMO original earlystopping	0.1972	0.0526	0.1465
INMO ours earlystopping	0.1957	0.0524	0.1461

Table 19. Experimental results for the INMO method for the Amazon Book dataset.

	@ 20 REC	PREC	NDCG
TopPop	0.0114	0.0024	0.0069
UserKNN CF cosine	**0.1661**	**0.0353**	**0.1193**
ItemKNN CF cosine	**0.1880**	**0.0420**	**0.1379**
RP3beta	**0.1946**	**0.0418**	**0.1402**
GraphFilter CF	**0.1726**	**0.0364**	**0.1222**
EASE R	-	-	-
SLIM ElasticNet	**0.2006**	**0.0445**	**0.1451**
NegHOSLIM ElasticNet	**0.1947**	**0.0425**	**0.1408**
MF BPR	0.0876	0.0178	0.0597
IALS	**0.1447**	0.0290	**0.0941**
INMO paper	**0.1428**	**0.0301**	**0.0986**
INMO original earlystopping	0.1391	0.0297	0.0931
INMO ours earlystopping	0.1391	0.0297	0.0934

Table 20. Experimental results for the INMO method for the Yelp 2018 dataset.

	@ 20		
	REC	PREC	NDCG
TopPop	0.0171	0.0035	0.0102
UserKNN CF cosine	0.0846	0.0188	0.0545
ItemKNN CF cosine	0.0901	0.0205	0.0584
RP3beta	0.0907	0.0204	0.0583
GraphFilter CF	-	-	-
EASE R	-	-	-
SLIM ElasticNet	0.0870	0.0201	0.0571
NegHOSLIM ElasticNet	-	-	-
MF BPR	0.0539	0.0123	0.0332
IALS	0.0994	0.0220	0.0635
INMO paper	0.1026	0.0225	0.0651
INMO original earlystopping	0.1022	0.0224	0.0646
INMO ours earlystopping	**0.1032**	**0.0226**	**0.0651**

5 HYPERGRAPH CONTRASTIVE COLLABORATIVE FILTERING

Xia et al. [9] presents *Hypergraph Contrastive Collaborative Filtering* (HCCF), based on the LightGCN paradigm adds several components: besides the message passing done on the user-item adjacency matrix as in LightGCN, but with the addition of a nonlinear aggregation function, HCCF incorporates one layer of message passing done on a hypergraph whose adjacency matrix is learnable and decomposed as the product of two lower dimensionality matrices. There is an additional step called Hierarchical Hypergraph Mapping which does the usual message passing but on the learned hypergraph adjacency matrix. The model is trained with contrastive learning using the InfoNCE loss, the goal is to push the embeddings learned via the message passing on the user-item adjacency matrix to be close to those obtained by learning the low dimensional approximation of the hypergraph adjacency. The source code is publicly available[5].

5.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 21

Yelp: preprocessed with 10-cores
Movielens 10M: preprocessed with 10-cores
Amazon-book: preprocessed with 20-cores

Dataset	Interactions	Items	Users	Sparsity
Yelp	1527326	24734	29601	9.979E-01
Movielens 10M	9998816	10196	69878	9.860E-01
Amazon-book	3200224	77801	78578	9.995E-01

Table 21. Dataset characteristics for HCCF.

[5] https://github.com/akaxlh/HCCF

Hyperparameter	Described in	Value			
		All datasets	Movielens 10M	Yelp	Amazon Book
epochs	Paper	100	-	-	-
sgd mode	Paper	Adam	-	-	-
learning rate	Paper	10^{-3}	-	-	-
embedding size	Paper	32	-	-	-
learning rate decay	Paper	0.96	-	-	-
GNN layers (K)	Paper	2	-	-	-
hyperedge size	Paper	128	-	-	-
hypergraph mapping layers (C)	Paper	3	-	-	-
batch size	source code	256	-	-	2048 [a]
dropout	source code	0.5	0.0	-	-
contrastive loss weight (λ_1)	source code	10^{-4}	10^{-6}	10^{-4}	10^{-7}
λ_2	source code	10^{-5}	-	10^{-3}	10^{-2}
contrastive loss temperature (τ)	Paper / source code	1.0	0.1	1.0	0.1
leaky relu slope	Paper	0.5	-	-	-

[a]The paper states the optimal value is 256 but in the experiments we use 2048 for Amazon due to the very large computational cost of this model. On Amazon Book a batch size of 256 results in a training time of 45 minutes per epoch, hence a total of 3 days.

Table 22. Hyperparameter Values for HCCF.

Table 23. Experimental results for the HCCF method for the Yelp 2018 dataset.

	@ 20		@ 40	
	REC	NDCG	REC	NDCG
TopPop	0.0123	0.0109	0.0208	0.0141
UserKNN CF cosine	**0.0973**	**0.0848**	**0.1527**	**0.1049**
ItemKNN CF cosine	**0.1054**	**0.0922**	**0.1646**	**0.1134**
RP3beta	**0.1082**	**0.0947**	**0.1687**	**0.1165**
GraphFilter CF	**0.1115**	**0.0966**	**0.1752**	**0.1196**
EASE R	-	-	-	-
SLIM ElasticNet	**0.1062**	**0.0935**	**0.1642**	**0.1144**
NegHOSLIM ElasticNet	**0.1032**	**0.0908**	**0.1602**	**0.1112**
MF BPR	0.0601	0.0507	0.0990	0.0650
IALS	**0.1090**	**0.0948**	**0.1706**	**0.1171**
HCCF paper	0.0607	0.0510	0.1007	0.0658
HCCF	0.0812	0.0699	0.1324	0.0885
HCCF no earlystopping	0.0810	0.0692	0.1325	0.0881

Table 24. Experimental results for the HCCF method for the Amazon Book dataset.

	@ 20		@ 40	
	REC	NDCG	REC	NDCG
TopPop	0.0097	0.0073	0.0157	0.0093
UserKNN CF cosine	**0.1476**	**0.1278**	**0.1973**	**0.1434**
ItemKNN CF cosine	**0.1726**	**0.1502**	**0.2209**	**0.1650**
RP3beta	**0.1773**	**0.1510**	**0.2377**	**0.1702**
GraphFilter CF	**0.1580**	**0.1344**	**0.2131**	**0.1520**
EASE R	-	-	-	-
SLIM ElasticNet	**0.1885**	**0.1626**	**0.2458**	**0.1803**
NegHOSLIM ElasticNet	-	-	-	-
MF BPR	**0.0687**	**0.0532**	**0.1012**	**0.0640**
IALS	**0.1185**	**0.0928**	**0.1744**	**0.1113**
HCCF paper	0.0344	0.0258	0.0561	0.0330
HCCF	0.0619	0.0467	0.1002	0.0593
HCCF no earlystopping	0.0612	0.0461	0.0981	0.0582

Table 25. Experimental results for the HCCF method for the Movielens 10M dataset.

	@ 20		@ 40	
	REC	NDCG	REC	NDCG
TopPop	0.1363	0.1903	0.2114	0.2022
UserKNN CF cosine	**0.3503**	**0.4448**	**0.4700**	**0.4595**
ItemKNN CF cosine	**0.2816**	**0.3645**	0.3884	0.3790
RP3beta	**0.2886**	**0.3761**	0.3960	**0.3895**
GraphFilter CF	**0.3342**	**0.4210**	**0.4484**	**0.4354**
EASE R	-	-	-	-
SLIM ElasticNet	**0.3387**	**0.4422**	**0.4578**	**0.4563**
NegHOSLIM ElasticNet	-	-	-	-
MF BPR	**0.2849**	0.3569	0.3989	0.3759
IALS	**0.3368**	**0.4232**	**0.4593**	**0.4426**
HCCF paper	0.2048	0.2467	0.3081	0.2717
HCCF	0.2780	0.3577	0.4004	0.3792
HCCF no earlystopping	0.2716	0.3544	0.3935	0.3752

6 HAKG: HIERARCHY-AWARE KNOWLEDGE GATED NETWORK FOR RECOMMENDATION

Du et al. [1] presents *Hierarchy-Aware Knowledge Gated Network* (HAKG), which aims to combine graphs obtained with collaborative interactions as well as knowledge-based. The goal of the paper is to exploit the hierarchical structure of knowledge graphs as well as the "higher order" relations in collaborative data, hence it is not sufficient to use a Euclidean space and therefore the embeddings are represented in hyperbolic space. The paper proposes a hierarchy-aware modeling strategy which includes an aggregation function for hyperbolic embeddings and a constraint on the angles generated by embedding involved aiming at better preserving the hierarchical structure. The aggregation function is computed in Euclidean space, so the embeddings are converted from hyperbolic to Euclidean space, then aggregated, and then converted back to Hyperbolic space. Knowledge-based and collaborative embeddings are separate (dual embeddings) and are fused with a "learnable gating fusion unit", which learns a weight matrix. The final prediction is computed with the cosine similarity of embeddings. The source code is publicly vailable[6].

6.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 26

Alibaba-iFashion: processed with 10-core
Yelp2018: processed with 10-core
Last-FM: processed with from KGAT including the knowledge base [7] "we take the subset of the dataset where the timestamp is from Jan, 2015 to June, 2015. We use the 10-core setting"

The two-hop neighbor entities of items in KG are used to construct the item knowledge graph for each dataset. All existing relations are considered as hierarchical. 10-core applied on the entities in the KG as well.

Dataset	Interactions	Items	Users	Sparsity
Alibaba-iFashion	1781093	30040	114737	9.995E-01
Yelp2018	1183610	45538	45919	9.994E-01
last-fm	1542856	48123	23566	9.986E-01

Table 26. Dataset characteristics for HAKG.

[6] https://github.com/zealscott/HAKG

Hyperparameter	Described in	Value			
		All datasets	Alibaba-iFashion	Yelp2018	Last-FM
embedding size	Paper	64	-	-	-
optimizer	Paper	Adam	-	-	-
batch size	Paper	4096	-	-	-
weight of angle loss w	Source code	$5 \cdot 10^{-3}$	-	-	-
learning rate	Source code	-	10^{-4}	$5 \cdot 10^{-4}$	10^{-4}
GNN layers	Source code	-	3	2	3
negative samples $\lvert M_u \rvert$	Paper	-	200	400	400
margin m	Paper	-	0.6	0.8	0.7

Table 27. Hyperparameter Values for HAKG.

Table 28. Experimental results for the HAKG method for the Alibaba iFashion Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0312	0.0167
UserKNN CF cosine	0.1090	0.0700
ItemKNN CF cosine	0.1264	**0.0818**
RP3beta	0.1247	**0.0807**
GraphFilter CF	0.1182	0.0742
EASE R	0.1262	**0.0819**
SLIM ElasticNet	**0.1276**	**0.0832**
NegHOSLIM ElasticNet	0.1259	**0.0822**
MF BPR	0.0761	0.0460
IALS	**0.1268**	**0.0807**
HAKG paper	**0.1319**	**0.0848**
HAKG original earlystopping	0.1263	0.0790
HAKG ours earlystopping	0.1266	0.0791

Table 29. Experimental results for the HAKG method for the Yelp 2018 Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0174	0.0110
UserKNN CF cosine	0.0715	0.0477
ItemKNN CF cosine	0.0727	0.0485
RP3beta	0.0733	0.0485
GraphFilter CF	0.0752	0.0492
EASE R	-	-
SLIM ElasticNet	0.0739	0.0494
NegHOSLIM ElasticNet	0.0692	0.0465
MF BPR	0.0484	0.0310
IALS	0.0764	0.0495
HAKG paper	0.0778	0.0501

Table 30. Experimental results for the HAKG method for the Last-FM Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0229	0.0198
UserKNN CF cosine	**0.1720**	0.1695
ItemKNN CF cosine	**0.1836**	**0.1838**
RP3beta	**0.2012**	**0.2014**
GraphFilter CF	**0.1806**	**0.1729**
EASE R	-	-
SLIM ElasticNet	**0.2070**	**0.2078**
NegHOSLIM ElasticNet	**0.2049**	**0.2058**
MF BPR	0.1281	0.1250
IALS	**0.1750**	0.1645
HAKG paper	0.1008	0.0931
HAKG ours earlystopping	0.1712	0.1702

7 GRAPH TREND FILTERING NETWORKS FOR RECOMMENDATION

Fan et al. [2] presents *Graph Trend Filtering Networks for Recommendation* (GTN), which proposes a method to adaptively capture the reliability of interactions. This is done with a new *smoothness* constraint on the embeddings, which in practice penalizes the occurrence of interactions between users and items with very different embeddings. The paper then proposes to use the Proximal Alternating Predictor-Corrector method and formulates an iterative solver requiring three steps. The source code is publicly available[7].

7.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 31

Gowalla: from LightGCN [3]
Yelp 2018: from LightGCN [3]
Amazon Book: from LightGCN [3]
LastFM: from KGAT [7] "we take the subset of the dataset where the timestamp is from Jan, 2015 to June, 2015. We use the 10-core setting"

Dataset	Interactions	Items	Users	Sparsity
Gowalla	1027370	40981	29858	9.992E-01
Yelp 2018	1561406	38048	31668	9.987E-01
Amazon Book	2984108	91599	52643	9.994E-01
LastFM	1542856	48123	23566	9.986E-01

Table 31. Dataset characteristics for GTN.

Hyperparameter	Described in	Value				
		All datasets	Gowalla	Yelp 2018	Amazon Book	LastFM
embedding size	Source code	256	-	-	-	-
optimizer	Paper	Adam	-	-	-	-
batch size	Source code	2048	-	-	-	-
epochs	Paper	1000	-	-	-	-
learning rate	Source code	10^{-3}	-	-	-	-
GNN layers	Paper	-	3	3	2	3
embedding smoothness weight[a]	Paper	3	-	-	-	-
l2 regularization	Source code	10^{-4}	-	-	-	-
dropout rate LightGCN[b]	Source code	0.4	-	-	-	-
dropout rate GTN[c]	Source code	0.1	-	-	-	-
ogb	Paper	True	-	-	-	-
incnorm_para	Paper	True	-	-	-	-

[a]This is called *lambda2*
[b]This is called *keep_prob* and is 0.6, hence dropout is 0.4.
[c]This is called *prop_dropout*

Table 32. Hyperparameter Values for GTN.

[7]https://github.com/wenqifan03/GTN-SIGIR2022

Table 33. Experimental results for the GTN method for the Yelp 2018 Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0124	0.0101
UserKNN CF cosine	**0.0637**	**0.0533**
ItemKNN CF cosine	**0.0622**	**0.0514**
RP3beta	**0.0672**	**0.0558**
GraphFilter CF	**0.0693**	**0.0568**
EASE R	-	-
SLIM ElasticNet	**0.0646**	**0.0541**
NegHOSLIM ElasticNet	**0.0590**	**0.0492**
MF BPR	0.0382	0.0313
IALS	**0.0667**	**0.0546**
GTN paper	**0.0679**	**0.0554**
GTN	0.0516	0.0422

Table 34. Experimental results for the GTN method for the Amazon Book Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0051	0.0044
UserKNN CF cosine	**0.0616**	**0.0518**
ItemKNN CF cosine	**0.0750**	**0.0624**
RP3beta	**0.0701**	**0.0585**
GraphFilter CF	**0.0710**	**0.0585**
EASE R	-	-
SLIM ElasticNet	**0.0757**	**0.0600**
NegHOSLIM ElasticNet	**0.0754**	**0.0609**
MF BPR	0.0254	0.0203
IALS	**0.0451**	**0.0347**
GTN paper	**0.0450**	**0.0346**
GTN	0.0291	0.0228

Table 35. Experimental results for the GTN method for the Gowalla Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0416	0.0317
UserKNN CF cosine	0.1699	0.1387
ItemKNN CF cosine	0.1559	0.1228
RP3beta	0.1811	0.1490
GraphFilter CF	0.1843	0.1505
EASE R	-	-
SLIM ElasticNet	0.1767	0.1448
NegHOSLIM ElasticNet	-	-
MF BPR	0.1319	0.1060
IALS	-	-
GTN paper	0.1870	0.1588

Table 36. Experimental results for the GTN method for the Last-FM Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0229	0.0198
UserKNN CF cosine	**0.1720**	**0.1695**
ItemKNN CF cosine	**0.1836**	**0.1838**
RP3beta	**0.2012**	**0.2014**
GraphFilter CF	**0.1806**	**0.1729**
EASE R	-	-
SLIM ElasticNet	**0.2070**	**0.2078**
NegHOSLIM ElasticNet	**0.2049**	**0.2058**
MF BPR	**0.1281**	**0.1250**
IALS	**0.1750**	**0.1645**
GTN paper	0.0932	0.0857
GTN	0.1142	0.1110

8 KNOWLEDGE GRAPH CONTRASTIVE LEARNING FOR RECOMMENDATION

Yang et al. [10] presents *Knowledge Graph Contrastive Learning framework* (KGCL), aiming to reduce the impact of noisy knowledge bases, this is done with a knowledge graph augmentation schema that guides a contrastive learning process. KGCL uses a parameterized attention matrix on the concatenation of the user and item embeddings to calculate an estimation of relevance between the two. KGCL also uses TransE, which is a translation aware loss function aiming to ensure that the embedding of the head entity + the embedding of the relation is close to the embedding of the tail entity (i.e., $e_h + e_r \approx e_t$). The training is done with contrastive learning and multiple views are created with a graph augmentation scheme which aims to identify items that are less sensitive to structure (edges) variations, the contrastive learning process is also guided by the knowledge based.

The source code is publicly available[8].

8.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 37

Yelp 2018: 10-core, from HAKG [1]. The entities are collected in the same way as [7]
Amazon Book: 10-core. The entities are collected in the same way as [7]
MIND: The data and kwnoledge base are collected in the same way as [6] "one million users who had at least 5 news clicks during six weeks (i.e., October 12 to November 22, 2019) were randomly sampled. "

Dataset	Interactions	Items	Users	Sparsity
amazon-book	846434	24915	70679	9.995E-01
yelp2018	1183610	45538	45919	9.994E-01
MIND	2545327	48957	300000	9.998E-01

Table 37. Dataset characteristics for KGCL.

[8] https://github.com/yuh-yang/KGCL-SIGIR22

Hyperparameter	Described in	Value			
		All datasets	Amazon Books	Yelp 2018	MIND
embedding size	Paper	64	-	-	-
learning date	Paper	10^{-3}	-	-	$5 \cdot 10^{-4}$
batch size	Paper	2048	-	-	-
self supervised loss weight λ_1	Paper	0.1	-	-	0.06[a]
contrastive loss temperature τ	Paper	0.2	-	-	-
optimizer	source code	Adam	-	-	-
epochs	source code	1000	-	-	-
GNN layers K	source code	3	-	-	-
GNN dropout rate	source code	0.2	0.2	0.2	0.4
entities per head	source code	10	-	-	6
knowledge graph dropout rate	source code	0.5	-	-	0.5
user interaction dropout rate	source code	0.001	0.05	0.1	0.4
mix_ratio[b]	source code	-	0.75[c]	-	0.6[d]
uicontrast[e]	source code	-	"WEIGHTED"	"WEIGHTED"	"WEIGHTED-MIX"
l2 regularization	source code	10^{-4}[f]	-	-	10^{-3}
learning rate milestones	source code	-	[1500, 2500][g]	[1500, 2500][h]	[5, 10]
min number of epochs[i]	source code	-	15	25	1
earlystopping patience	source code	-	5	5	3

[a]source code
[b]Seems to be used to add random samples as part of the user interaction dropout process, only when *uicontrast* is "weighted-mix"
[c]Useless hyperparameter, uicontrast weighted does not use it.
[d]Defined as 1-ui_p_drop
[e]I suppose this could be how the graph augmentations are generated for the contrastive learning part, but the hyperparameter values are not described in the paper.
[f]The TransR learning part had a hardcoded l2 regularization weight of 10^{-3}, the ported version uses the one provided as hyperparameter.
[g]Useless setting, the epochs never reach 1500
[h]Useless setting, the epochs never reach 1500
[i]While I understand how the convergence of different methods may require different strategies to early-stop training (e.g., one may need more patience) I find oddly specific that for two datasets the patience is 5 while for MIND it is 3.

Table 38. Hyperparameter Values for KGCL.

Table 39. Experimental results for the KGCL method for the Amazon Book Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0287	0.0123
UserKNN CF cosine	**0.1658**	**0.0944**
ItemKNN CF cosine	**0.1653**	**0.0974**
RP3beta	**0.1706**	**0.0983**
GraphFilter CF	**0.1712**	**0.0973**
EASE R	-	-
SLIM ElasticNet	**0.1742**	**0.1031**
NegHOSLIM ElasticNet	**0.1740**	**0.1028**
MF BPR	0.1143	0.0637
IALS	**0.1676**	**0.0908**
KGCL paper	**0.1496**	**0.0793**
KGCL	0.1466	0.0791

Table 40. Experimental results for the KGCL method for the Yelp 2018 Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0174	0.0110
UserKNN CF cosine	0.0715	**0.0477**
ItemKNN CF cosine	0.0727	**0.0485**
RP3beta	**0.0733**	**0.0485**
GraphFilter CF	**0.0752**	**0.0492**
EASE R	-	-
SLIM ElasticNet	**0.0739**	**0.0494**
NegHOSLIM ElasticNet	0.0692	0.0465
MF BPR	0.0484	0.0310
IALS	**0.0764**	**0.0495**
KGCL paper	**0.0756**	**0.0493**
KGCL	0.0730	0.0475

Table 41. Experimental results for the KGCL method for the MIND Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0894	0.0437
UserKNN CF cosine	0.0972	0.0509
ItemKNN CF cosine	**0.1225**	**0.0647**
RP3beta	**0.1187**	**0.0621**
GraphFilter CF	-	-
EASE R	-	-
SLIM ElasticNet	**0.1287**	**0.0686**
NegHOSLIM ElasticNet	**0.1281**	**0.0681**
MF BPR	0.0888	0.0435
IALS	**0.1130**	**0.0600**
KGCL paper	**0.1073**	**0.0551**
KGCL	0.1010	0.0531

A LIGHTGCN: SIMPLIFYING AND POWERING GRAPH CONVOLUTION NETWORK FOR RECOMMENDATION

In He et al. [3] LightGCN is proposed, a graph-based collaborative filtering method in which the user and item embeddings are propagated according to the graph adjacency matrix. The source code is publicly available[9].

A.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 42

Amazon Book: processed extracting its 10-core.
Gowalla: processed extracting its 10-core.
Yelp2018: processed extracting its 10-core.

Dataset	Interactions	Items	Users	Sparsity
Amazon Book	2984108	91599	52643	9.994E-01
Gowalla	1027370	40981	29858	9.992E-01
Yelp2018	1561406	38048	31668	9.987E-01

Table 42. Dataset characteristics for LightGCN.

Hyperparameter	Described in	Value			
		All datasets	Amazon Book	Gowalla	Yelp2018
embedding size	Paper	64	-	-	-
optimizer	Paper	Adam	-	-	-
learning rate	Paper	10^{-3}	-	-	-
batch size	Paper	1024	2048	-	-
l2 reg	Paper	10^{-4}	-	-	-
dropout	Source code	0.0	-	-	-
epochs	Paper	1000 (max)	-	-	-
GNN layers K	Paper	3	-	-	-
α_k	Paper	$\frac{1}{1+K}$	-	-	-

Table 43. Hyperparameter Values for LightGCN.

[9]https://github.com/gusye1234/LightGCN-PyTorch

Table 44. Experimental results for the LightGCN method for the Gowalla dataset.

	@ 20	
	REC	NDCG
TopPop	0.0416	0.0317
UserKNN CF cosine	0.1699	0.1387
ItemKNN CF cosine	0.1559	0.1228
RP3beta	**0.1811**	0.1490
GraphFilter CF	**0.1843**	0.1505
EASE R	-	-
SLIM ElasticNet	0.1767	0.1448
NegHOSLIM ElasticNet	-	-
MF BPR	0.1319	0.1060
IALS	-	-
LightGCN paper	**0.1830**	**0.1550**
LightGCN original earlystopping	0.1798	0.1536
LightGCN ours earlystopping	0.1775	0.1521

Table 45. Experimental results for the LightGCN method for the Amazon Book Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0051	0.0044
UserKNN CF cosine	**0.0616**	**0.0518**
ItemKNN CF cosine	**0.0750**	**0.0624**
RP3beta	**0.0701**	**0.0585**
GraphFilter CF	**0.0710**	**0.0585**
EASE R	-	-
SLIM ElasticNet	**0.0757**	**0.0600**
NegHOSLIM ElasticNet	**0.0754**	**0.0609**
MF BPR	0.0254	0.0203
IALS	**0.0451**	**0.0347**
LightGCN paper	0.0406	0.0313
LightGCN original earlystopping	0.0407	0.0315
LightGCN ours earlystopping	0.0408	0.0316

Table 46. Experimental results for the LightGCN method for the Yelp 2018 Original dataset.

	@ 20	
	REC	NDCG
TopPop	0.0124	0.0101
UserKNN CF cosine	**0.0637**	**0.0533**
ItemKNN CF cosine	0.0622	**0.0514**
RP3beta	**0.0672**	**0.0558**
GraphFilter CF	**0.0693**	**0.0568**
EASE R	-	-
SLIM ElasticNet	**0.0646**	**0.0541**
NegHOSLIM ElasticNet	0.0590	0.0492
MF BPR	0.0382	0.0313
IALS	**0.0667**	**0.0546**
LightGCN paper	**0.0649**	**0.0530**
LightGCN original earlystopping	0.0618	0.0506
LightGCN ours earlystopping	0.0626	0.0513

B HYPERPARAMETER RANGE

Algorithm	Hyperparameter	Range	Type	Distribution
UserKNN, ItemKNN cosine	topK	5 - 1000	Integer	uniform
	shrink	0 - 1000	Integer	uniform
	similarity	cosine	Categorical	
	normalize[a]	True, False	Categorical	
	feature weighting	none, TF-IDF, BM25	Categorical	
RP3beta	topK	5 - 1000	Integer	uniform
	alpha	0 - 2	Real	uniform
	beta	0 - 2	Real	uniform
	normalize similarity[??]	True, False	Categorical	
GraphFilter CF	topK	5 - 5000	Integer	uniform
	alpha	10^{-3} - 10^{+3}	Real	log-uniform
	num factors	1 - 350	Integer	uniform

[a]The *normalize* hyperparameter in KNNs refers to the use of the denominator when computing the similarity.

Table 47. Hyperparameter values for our KNN baselines.

Algorithm	Hyperparameter	Range	Type	Distribution
EASE R	l2 norm	10^{0} - 10^{+7}	Real	log-uniform
SLIMElasticNet	topK	5 - 1000	Integer	uniform
	l1 ratio	10^{-5} - 10^{0}	Real	log-uniform
	alpha	10^{-3} - 10^{0}	Real	uniform
NegHOSLIM ElasticNet	feature pairs n	1 - 1000	Integer	uniform
	topK	5 - 1000	Integer	uniform
	l1 ratio	10^{-5} - 10^{0}	Real	log-uniform
	alpha	10^{-3} - 10^{0}	Real	uniform
MF BPR	num factors	1 - 200[a]	Integer	uniform
	epochs	1 - 1500	Integer	early-stopping
	sgd mode	sgd, adam, adagrad	Categorical	
	batch size	2^{0} - 2^{10}	Integer	log-uniform
	positive reg	10^{-5} - 10^{-2}	Real	log-uniform
	negative reg	10^{-5} - 10^{-2}	Real	log-uniform
	learning rate	10^{-4} - 10^{-1}	Real	log-uniform
IALS	num factors	1 - 200[a]	Integer	uniform
	epochs	1 - 500[b]	Integer	early-stopping
	confidence scaling	linear, log	Categorical	
	alpha	10^{-3} - $5 \cdot 10^{+1}$ [c]	Real	log-uniform
	epsilon	10^{-3} - 10^{+1} [c]	Real	log-uniform
	reg	10^{-5} - 10^{-2}	Real	log-uniform

[a]The number of factors is lower due to the algorithm being slower.
[b]The number of epochs is lower due to the algorithm being slower, but converging in a lower number of epochs.
[c]The maximum value of this hyperparameter had been suggested in the article proposing the algorithm.

Table 48. Hyperparameter values for our ML baselines.

REFERENCES

[1] Yuntao Du, Xinjun Zhu, Lu Chen, Baihua Zheng, and Yunjun Gao. 2022. HAKG: Hierarchy-Aware Knowledge Gated Network for Recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 1390–1400. https://doi.org/10.1145/3477495.3531987

[2] Wenqi Fan, Xiaorui Liu, Wei Jin, Xiangyu Zhao, Jiliang Tang, and Qing Li. 2022. Graph Trend Filtering Networks for Recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 112–121. https://doi.org/10.1145/3477495.3531985

[3] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, Jimmy X. Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu (Eds.). ACM, 639–648. https://doi.org/10.1145/3397271.3401063

[4] Shaowen Peng, Kazunari Sugiyama, and Tsunenori Mine. 2022. Less is More: Reweighting Important Spectral Graph Features for Recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 1273–1282. https://doi.org/10.1145/3477495.3532014

[5] Changxin Tian, Yuexiang Xie, Yaliang Li, Nan Yang, and Wayne Xin Zhao. 2022. Learning to Denoise Unreliable Interactions for Graph Collaborative Filtering. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 122–132. https://doi.org/10.1145/3477495.3531889

[6] Yu Tian, Yuhao Yang, Xudong Ren, Pengfei Wang, Fangzhao Wu, Qian Wang, and Chenliang Li. 2021. Joint Knowledge Pruning and Recurrent Graph Convolution for News Recommendation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 51–60. https://doi.org/10.1145/3404835.3462912

[7] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge Graph Attention Network for Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 950–958. https://doi.org/10.1145/3292500.3330989

[8] Yunfan Wu, Qi Cao, Huawei Shen, Shuchang Tao, and Xueqi Cheng. 2022. INMO: A Model-Agnostic and Scalable Module for Inductive Collaborative Filtering. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 91–101. https://doi.org/10.1145/3477495.3532000

[9] Lianghao Xia, Chao Huang, Yong Xu, Jiashu Zhao, Dawei Yin, and Jimmy X. Huang. 2022. Hypergraph Contrastive Collaborative Filtering. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 70–79. https://doi.org/10.1145/3477495.3532058

[10] Yuhao Yang, Chao Huang, Lianghao Xia, and Chenliang Li. 2022. Knowledge Graph Contrastive Learning for Recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 1434–1443. https://doi.org/10.1145/3477495.3532009

[11] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are Graph Augmentations Necessary?: Simple Graph Contrastive Learning for Recommendation. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai (Eds.). ACM, 1294–1303. https://doi.org/10.1145/3477495.3531937

Received February 2007; revised March 2009; accepted June 2009