Reproducibility and Artifact Consistency of the SIGIR 2022 Recommender Systems Papers Based on Message Passing - Additional Online Material

MAURIZIO FERRARI DACREMA, Politecnico di Milano, Italy
MICHAEL BENIGNI, Politecnico di Milano, Italy
NICOLA FERRO, Università degli Studi di Padova, Italy

This is the additional online material associated with our paper "Reproducibility and Artifact Consistency of the SIGIR 2022 Recommender Systems Papers Based on Message Passing". This additional material contains the full results of the experiments of which, due to space reasons and for the sake of improving readability, only a selection is reported in the paper. The results for each of the papers we analyze are reported in separate sections.

ACM Reference Format:
Maurizio Ferrari Dacrema, Michael Benigni, and Nicola Ferro. 2025. Reproducibility and Artifact Consistency of the SIGIR 2022 Recommender Systems Papers Based on Message Passing - Additional Online Material. 1, 1 (July 2025), 62 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Contents

Authors' addresses: Maurizio Ferrari Dacrema, Politecnico di Milano, Italy, maurizio.ferrari@polimi.it; Michael Benigni, Politecnico di Milano, Italy, michael.benigni@polimi.it; Nicola Ferro, Università degli Studi di Padova, Italy, ferro@dei.unipd.it.

XXXX-XXXX/2025/7-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 BASELINES

Here we list all the 21 collaborative and 5 content-based and hybrid baseline algorithms used in each experiment, most of them are the same used by Ferrari Dacrema et al. [6]:

Non-personalized.

- **Random**: non-personalized method recommending random items the user has not yet interacted with.
- **TopPop**: non-personalized method recommending to all users the most popular items the user has not yet interacted with.
- **Global Effects**: leverages global, item and user biases to recommend items.

Nearest-Neighbor Collaborative and Content-Based.

- **UserKNN**: user-based nearest-neighbor algorithm [15], with cosine similarity and shrinkage [1].
- **ItemKNN**: item-based nearest-neighbor algorithm [16], with cosine similarity and shrinkage [1].
- **UserKNN CBF**: UserKNN computed on the user features.
- **UserKNN CFCBF**: UserKNN computed on the concatenation of the user profile and the user features. A hyperparameter controls the weight of the content-based part.
- **ItemKNN CBF <attribute>**: ItemKNN computed on the item features.
- **ItemKNN CFCBF <attribute>**: ItemKNN computed on the concatenation of the item interactions and the item features. A hyperparameter controls the weight of the content-based part.

Graph-based.

- **P$^3\alpha$**: graph based algorithms modeling random walk on the bipartite graph of users-items interactions.
- **RP$^3\beta$**: graph-based method that uses a two-steps random walk from users to items and vice-versa, where transition probabilities are computed from the normalized ratings [12].
- **GF-CF**: a graph-based method that is based on a low-pass filter and has a closed form solution [17].

Item-Based Machine Learning.

- **EASE^R**: An "embarrassingly shallow" linear model with strong connections with autoencoders and a closed form solution [18].[1]
- **SLIM**: item-based model that uses linear regression to compute the item similarity [11].
- **SLIM-BPR**: item-based model similar to SLIM that computes the item similarity optimizing the *Bayesian Personalized Ranking* (BPR) loss [14].
- **NegHOSLIM**: linear full-rank model similar to SLIM that includes higher-order interactions as input-features [19].
- **NegHOSLIM (EN)**: linear full-rank model similar to NegHOSLIM that optimizes the ElasticNet loss.[2]

Matrix Factorization.

- **MF-BPR**: matrix factorization method based on the *Bayesian Personalized Ranking* (BPR) loss [14].

[1] EASE^R has a high memory requirements and often exceeds the 64GB RAM available on our server.
[2] Due to the large memory requirement of the original NegHOSLIM we trained this version by using an ElasticNet loss which reduces memory requirement but sacrifices some effectiveness.

- **MF-WARP**: matrix factorization method based on the *Weighted Approximate-Rank Pairwise* loss (WARP).
- **SVDpp**: matrix factorization method for rating prediction accounting for user biases [9].[3]
- **PureSVD**: Matrix factorization method based on the truncated SVD decomposition of the user-item interaction matrix [3].[4]
- **NMF**: matrix factorization method that decomposes ratings matrix into two non-negative matrices [2].[5]
- **iALS**: matrix factorization method for ranking tasks based on alternating least-squares [8].

Other Machine Learning.

- **MultVAE**: variational autoencoder that assumes a multinomial likelihood for user-item interactions [10].
- **LightFM CF**: factorization machine method that uses only collaborative data.[6]
- **LightFM ItemHybrid <attribute>**: factorization machine method that uses a combination of collaborative and item features.
- **LightFM UserHybrid <attribute>**: factorization machine method that uses a combination of collaborative and user features.

Note that occasionally the results for **GF-CF**, **EASE**^{R}, **SLIM-BPR** and **NegHOSLIM** may be missing due to their memory requirements exceeding the 64GB available on our server.

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

Peng et al. [13] 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 original implementation is available on GitHub.[7]

2.1 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 collected from a social network where users check-in locations they visited.

[3]Note that to adapt SVDpp for the task of top-k recommendation we sample during training a certain quota of interactions that did not occur and attribute them a rating of zero. The specific quota is a hyperparameter.

[4]We use a standard SVD decomposition method provided in the `scikit-learn` package for Python.

[5]We use a standard NMF decomposition method provided in the `scikit-learn` package for Python.

[6]We use the LightFM library, https://github.com/lyst/lightfm

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

Dataset	Interactions	Items	Users	Sparsity
CiteULike	210504	16980	5551	0.9978
Gowalla	1027370	40981	29858	0.9992
MovieLens1M	1000209	3952	6040	0.9581
MovieLens100k	100000	1682	943	0.9370
Pinterest	1000154	9836	37501	0.9973

Table 1. Dataset statistics for GDE.

2.2 Results

The hyperparameter values used in our experiments are reported in Table 2 and the results for all the datasets and baseline algorithms are reported in Table 3 (CiteULike), 4 (MovieLens1M), 5 (MovieLens100k), 6 (Pinterest), and 7 (Gowalla).

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 that the optimal value should be 4.5.
[b]The paper reports that the optimal value should be 4.5.
[c]If the value is "smoothed" only the smooth features (low frequencies) are used, the value is "both" rough features (high frequencies) are used as well.

Table 2. Hyperparameter values for GDE.

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

	Cutoff 20		
	Recall (GDE)	Recall	NDCG
Random	0.0019	0.0010	0.0020
TopPop	0.0525	0.0446	0.0544
GlobalEffects	0.0019	0.0010	0.0017
UserKNN CF	**0.1003**	0.0552	**0.1131**
ItemKNN CF	**0.0997**	0.0545	**0.1121**
$\mathrm{P}^3\alpha$	**0.1032**	0.0572	**0.1156**
$\mathrm{RP}^3\beta$	**0.1028**	0.0571	**0.1151**
GF-CF	0.0973	**0.0608**	0.1006
EASE^R	0.0981	0.0541	**0.1099**
SLIM BPR	0.0850	0.0499	0.0937
SLIM	**0.1000**	0.0553	**0.1116**
NegHOSLIM	0.0980	0.0540	**0.1099**
NegHOSLIM (EN)	0.0983	0.0541	**0.1104**
MF-BPR	0.0316	0.0257	0.0371
MF-WARP	0.0197	0.0117	0.0220
SVDpp	0.0566	0.0439	0.0599
PureSVD	0.0607	0.0302	0.0670
NMF	0.0505	0.0246	0.0551
iALS	**0.1143**	**0.0667**	**0.1240**
LightFM CF	**0.1170**	**0.0760**	**0.1272**
MultVAE	<u>**0.1306**</u>	**0.0823**	<u>**0.1402**</u>
GDE paper	**0.1224**	-	**0.1339**
GDE our early-stopping	0.0570	0.0360	0.0551
GDE provided number of epochs	0.0015	0.0008	0.0014
GDE our hyperparameters	0.0991	0.0594	0.1086

Table 4. Experimental results for the GDE method for the MovieLens1M dataset.

	Cutoff 20		
	Recall (normalized)	Recall	NDCG
Random	0.0347	0.0053	0.0346
TopPop	0.3838	0.0863	0.4062
GlobalEffects	0.0376	0.0065	0.0399
UserKNN CF	0.4876	0.1172	0.5184
ItemKNN CF	0.4493	0.1067	0.4830
$P^3\alpha$	0.5033	0.1223	0.5326
$RP^3\beta$	0.5072	0.1230	0.5361
GF-CF	0.5247	0.1290	0.5537
$EASE^R$	0.4780	0.1105	0.5062
SLIM BPR	0.4764	0.1156	0.5087
SLIM	0.4644	0.1085	0.4950
NegHOSLIM	0.4693	0.1095	0.4996
NegHOSLIM (EN)	0.4612	0.1080	0.4930
MF-BPR	0.2848	0.0602	0.2918
MF-WARP	0.3586	0.0795	0.3756
SVDpp	0.4395	0.0992	0.4534
PureSVD	0.4753	0.1090	0.5029
NMF	0.4385	0.0969	0.4605
iALS	0.5147	0.1237	0.5415
LightFM CF	0.4856	0.1148	0.5069
MultVAE	0.5305	0.1330	0.5587
GDE paper	**0.5423**	-	**0.5715**
GDE our early-stopping	**0.5357**	**0.1344**	**0.5658**
GDE provided number of epochs	**0.5356**	**0.1349**	**0.5636**
GDE our hyperparameters	0.5291	0.1294	0.5564

Table 5. Experimental results for the GDE method for the MovieLens100k dataset.

	Cutoff 20		
	Recall (normalized)	Recall	NDCG
Random	0.0511	0.0122	0.0522
TopPop	0.4062	0.1301	0.4292
GlobalEffects	0.0255	0.0042	0.0238
UserKNN CF	0.4912	0.1734	0.5281
ItemKNN CF	0.4289	0.1570	0.4644
$P^3\alpha$	0.5002	0.1788	0.5348
$RP^3\beta$	0.4820	0.1667	0.5207
GF-CF	0.4421	0.1495	0.4747
$EASE^R$	0.4567	0.1518	0.4972
SLIM BPR	0.5011	0.1791	0.5346
SLIM	0.4689	0.1587	0.5047
NegHOSLIM	0.4505	0.1500	0.4904
NegHOSLIM (EN)	0.4654	0.1573	0.4979
MF-BPR	0.3905	0.1308	0.4141
MF-WARP	0.3646	0.1216	0.3867
SVDpp	0.4111	0.1431	0.4329
PureSVD	0.4583	0.1597	0.4920
NMF	0.4061	0.1301	0.4292
iALS	0.4194	0.1558	0.4370
LightFM CF	0.4685	0.1651	0.5037
MultVAE	0.5054	0.1805	0.5393
GDE paper	**0.5400**	-	**0.5731**
GDE our early-stopping	**0.5196**	**0.1902**	**0.5515**
GDE provided number of epochs	**0.5293**	**0.1930**	**0.5585**
GDE our hyperparameters	0.4229	0.1432	0.4516

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

	Cutoff 20		
	Recall (normalized)	Recall	NDCG
Random	0.0024	0.0020	0.0023
TopPop	0.0174	0.0144	0.0181
GlobalEffects	0.0028	0.0023	0.0030
UserKNN CF	0.0879	0.0758	0.0948
ItemKNN CF	0.0877	0.0755	0.0944
$P^3\alpha$	0.0885	0.0762	0.0954
$RP^3\beta$	0.0872	0.0749	0.0941
GF-CF	0.1007	0.0874	0.1081
$EASE^R$	0.0831	0.0713	0.0898
SLIM BPR	0.0827	0.0712	0.0889
SLIM	0.0852	0.0732	0.0919
NegHOSLIM	0.0799	0.0691	0.0857
NegHOSLIM (EN)	0.0851	0.0732	0.0917
MF-BPR	0.0654	0.0561	0.0699
MF-WARP	0.0619	0.0531	0.0647
SVDpp	0.0878	0.0761	0.0932
PureSVD	0.0706	0.0612	0.0762
NMF	0.0700	0.0608	0.0744
iALS	0.1067	0.0925	0.1146
LightFM CF	0.1013	0.0877	0.1084
MultVAE	0.1063	0.0920	0.1143
GDE paper	**0.1147**	-	**0.1240**
GDE our early-stopping	0.0026	0.0022	0.0024
GDE provided number of epochs	0.0026	0.0022	0.0024
GDE our hyperparameters	**0.1082**	**0.0940**	**0.1171**

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

	Cutoff 20		
	Recall (GDE)	Recall	NDCG
Random	0.0008	0.0005	0.0008
TopPop	0.0421	0.0298	0.0451
GlobalEffects	0.0005	0.0004	0.0005
UserKNN CF	0.1128	0.0748	0.1304
ItemKNN CF	0.1119	0.0741	0.1288
$P^3\alpha$	0.1153	0.0754	0.1326
$RP^3\beta$	0.1116	0.0737	0.1285
GF-CF	-	-	-
$EASE^R$	-	-	-
SLIM BPR	0.0958	0.0623	0.1089
SLIM	0.1057	0.0692	0.1219
NegHOSLIM	-	-	-
NegHOSLIM (EN)	0.1053	0.0690	0.1214
MF-BPR	0.0299	0.0202	0.0319
MF-WARP	0.0347	0.0201	0.0375
SVDpp	0.0945	0.0636	0.1016
PureSVD	0.0682	0.0445	0.0780
NMF	0.0568	0.0373	0.0655
iALS	0.1361	0.0963	0.1531
LightFM CF	0.1346	0.0949	0.1496
MultVAE	0.1362	0.0962	0.1540
GDE paper	**0.1449**	-	**0.1632**
GDE our early-stopping	0.0959	0.0704	0.1077
GDE provided number of epochs	**0.1433**	**0.1036**	**0.1627**
GDE our hyperparameters	0.1282	0.0910	0.1476

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

Yu et al. [26] 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 (e.g., random edge dropout) but rather the constrastive learning loss function (i.e., InfoNCE). The effect of the InfoNCE loss 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 [7] with random embedding perturbations, a regularizing loss and the aggregated user and item embeddings that start from layer 1, therefore excluding layer zero (i.e., $E^{(0)}$). The original implementation is available on GitHub.[8]

3.1 Datasets

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

- **DoubanBook:** Is a dataset of relations for the Douban Book service, with ratings 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:** Is a business reviews dataset. The split is the same used in LightGCN [7], see Section 10.
- **Amazon-Book:** Is a dataset of book purchases on Amazon. The split is the same used in LightGCN [7], see Section 10.

Dataset	Interactions	Items	Users	Sparsity
Amazon-Book	2984108	91599	52643	0.9994
DoubanBook	598420	22348	13025	0.9979
Yelp2018	1561406	38048	31668	0.9987

Table 8. Dataset statistics for SimGCL.

3.2 Results

The hyperparameter values used in our experiments are reported in Table 9 and the results for all the datasets and baseline algorithms are reported in Table 10 (Amazon-Book Original Split), 11 (Amazon-Book Our Split), 12 (DoubanBook), 13 (Yelp2018 Original Split), and 14 (Yelp2018 Our Split).

[8] 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	DoubanBook	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 Split dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0002	0.0002
TopPop	0.0051	0.0044
GlobalEffects	0.0004	0.0003
UserKNN CF	**<u>0.0616</u>**	**<u>0.0518</u>**
ItemKNN CF	**<u>0.0741</u>**	**<u>0.0617</u>**
$P^3\alpha$	**<u>0.0690</u>**	**<u>0.0550</u>**
$RP^3\beta$	**<u>0.0750</u>**	**<u>0.0608</u>**
GF-CF	**<u>0.0710</u>**	**<u>0.0585</u>**
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	**<u>0.0756</u>**	**<u>0.0600</u>**
NegHOSLIM	-	-
NegHOSLIM (EN)	**<u>0.0737</u>**	**<u>0.0607</u>**
MF-BPR	0.0281	0.0220
MF-WARP	0.0276	0.0217
SVDpp	0.0398	0.0301
PureSVD	0.0403	0.0335
NMF	0.0351	0.0296
iALS	0.0426	0.0342
LightFM CF	0.0452	0.0341
MultVAE	**<u>0.0593</u>**	**<u>0.0467</u>**
SimGCL paper	**0.0515**	**0.0414**
SimGCL our early-stopping	0.0507	0.0402
SimGCL provided number of epochs	0.0506	0.0402

Table 11. Experimental results for the SimGCL method for the Amazon Book Our Split dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0002	0.0002
TopPop	0.0093	0.0079
GlobalEffects	0.0002	0.0002
UserKNN CF	**0.1381**	**0.1333**
ItemKNN CF	**0.1707**	**0.1652**
$P^3\alpha$	**0.1746**	**0.1658**
$RP^3\beta$	**0.1687**	**0.1629**
GF-CF	**0.1530**	**0.1443**
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	**0.1871**	**0.1816**
NegHOSLIM	-	-
NegHOSLIM (EN)	**0.1865**	**0.1806**
MF-BPR	0.0603	0.0518
MF-WARP	0.0623	0.0543
SVDpp	0.0910	0.0791
PureSVD	0.0870	0.0844
NMF	0.0682	0.0656
iALS	**0.1166**	0.1012
LightFM CF	0.0784	0.0668
MultVAE	**0.1442**	**0.1325**
SimGCL paper	0.1157	0.1043
SimGCL provided number of epochs	0.1160	0.1047

Table 12. Experimental results for the SimGCL method for the DoubanBook dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0006	0.0005
TopPop	0.0722	0.0582
GlobalEffects	0.0001	0.0001
UserKNN CF	**0.1686**	**0.1575**
ItemKNN CF	<u>**0.1972**</u>	<u>**0.1908**</u>
$P^3\alpha$	<u>**0.2089**</u>	<u>**0.1981**</u>
$RP^3\beta$	<u>**0.2033**</u>	<u>**0.1841**</u>
GF-CF	<u>**0.1788**</u>	<u>**0.1604**</u>
$EASE^R$	<u>**0.2094**</u>	<u>**0.1994**</u>
SLIM BPR	<u>**0.1785**</u>	<u>**0.1667**</u>
SLIM	<u>**0.2250**</u>	<u>**0.2226**</u>
NegHOSLIM	-	-
NegHOSLIM (EN)	<u>**0.1971**</u>	<u>**0.1833**</u>
MF-BPR	0.0916	0.0774
MF-WARP	0.0825	0.0685
SVDpp	0.1620	0.1336
PureSVD	0.1420	0.1388
NMF	0.1313	0.1313
iALS	<u>**0.1833**</u>	<u>**0.1668**</u>
LightFM CF	**0.1699**	0.1385
MultVAE	<u>**0.1885**</u>	<u>**0.1694**</u>
SimGCL paper	**0.1772**	**0.1583**
SimGCL our early-stopping	0.1685	0.1492
SimGCL provided number of epochs	0.1629	0.1445

Table 13. Experimental results for the SimGCL method for the Yelp2018 Original Split dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0005	0.0004
TopPop	0.0124	0.0101
GlobalEffects	0.0006	0.0004
UserKNN CF	0.0638	0.0534
ItemKNN CF	0.0643	0.0536
$P^3\alpha$	0.0661	0.0548
$RP^3\beta$	0.0670	0.0558
GF-CF	0.0708	0.0583
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	0.0649	0.0543
NegHOSLIM	-	-
NegHOSLIM (EN)	0.0622	0.0521
MF-BPR	0.0392	0.0322
MF-WARP	0.0386	0.0317
SVDpp	0.0575	0.0471
PureSVD	0.0532	0.0446
NMF	0.0492	0.0408
iALS	0.0652	0.0541
LightFM CF	0.0597	0.0485
MultVAE	**0.0731**	**0.0602**
SimGCL paper	**0.0721**	**0.0601**
SimGCL our early-stopping	0.0716	0.0592
SimGCL provided number of epochs	0.0719	0.0594

Table 14. Experimental results for the SimGCL method for the Yelp 2018 Our Split dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0004	0.0003
TopPop	0.0162	0.0132
GlobalEffects	0.0003	0.0003
UserKNN CF	0.0929	0.0798
ItemKNN CF	0.0986	0.0850
$P^3\alpha$	0.0991	0.0845
$RP^3\beta$	0.0991	0.0848
GF-CF	0.1038	0.0880
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	0.1015	0.0887
NegHOSLIM	-	-
NegHOSLIM (EN)	0.0988	0.0858
MF-BPR	0.0540	0.0446
MF-WARP	0.0522	0.0431
SVDpp	0.0804	0.0670
PureSVD	0.0805	0.0695
NMF	0.0723	0.0615
iALS	0.1047	0.0895
LightFM CF	0.0929	0.0774
MultVAE	**0.1123**	**0.0954**
SimGCL paper	0.1073	0.0921
SimGCL provided number of epochs	0.1074	0.0920

4 LEARNING TO DENOISE UNRELIABLE INTERACTIONS FOR GRAPH COLLABORATIVE FILTERING

Tian et al. [20] 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. RGCF 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 original implementation is available on GitHub.[9]

4.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 15.

Amazon-Book: Is a dataset of book purchases on Amazon. Only users and items with at least 15 interactions are retained, this corresponds to the 15-cores subgraph.

MovieLens1M: Is a movie recommendation dataset. Ratings ≥ 4 are transformed into implicit interactions with value 1.

Yelp: Is a business reviews dataset. Only users and items with at least 15 interactions are retained, this corresponds to the 15-cores subgraph.

Dataset	Interactions	Items	Users	Sparsity
Amazon-Book	2517437	58051	58144	0.9993
MovieLens1M	836478	3883	6040	0.9643
Yelp	1730025	31731	45160	0.9988

Table 15. Dataset statistics for RGCF.

4.2 Results

The hyperparameter values used in our experiments are reported in Table 16 and the results for all the datasets and baseline algorithms are reported in Table 17 (MovieLens1M).

[9] https://github.com/ChangxinTian/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 16. Hyperparameter values for RGCF.

Table 17. Experimental results for the RGCF method for the MovieLens1M dataset.

	Cutoff 10			
	Recall	NDCG	HR	MRR
Random	0.0030	0.0047	0.0421	0.0120
TopPop	0.0773	0.1213	0.4894	0.2433
GlobalEffects	0.0007	0.0013	0.0117	0.0032
UserKNN CF	0.1939	0.2711	0.7733	0.4741
ItemKNN CF	0.1811	0.2578	0.7441	0.4610
$P^3\alpha$	0.1852	0.2565	0.7600	0.4561
$RP^3\beta$	0.1824	0.2557	0.7560	0.4577
GF-CF	**0.2076**	**0.2885**	**0.7897**	**0.4944**
$EASE^R$	**0.2128**	**0.3015**	**0.7970**	**0.5082**
SLIM BPR	0.1947	0.2667	0.7785	0.4716
SLIM	**0.2057**	**0.2944**	**0.7870**	**0.5034**
NegHOSLIM	**0.2141**	**0.3026**	**0.7972**	**0.5096**
NegHOSLIM (EN)	**0.2125**	**0.3001**	**0.7958**	**0.5059**
MF-BPR	0.1500	0.2105	0.6971	0.3894
MF-WARP	0.1406	0.2000	0.6778	0.3764
SVDpp	0.1867	0.2602	0.7555	0.4516
PureSVD	0.1942	0.2748	0.7733	0.4756
NMF	0.1773	0.2441	0.7448	0.4244
iALS	0.1938	0.2759	0.7707	0.4783
LightFM CF	0.1844	0.2478	0.7667	0.4455
MultVAE	**0.2029**	**0.2812**	**0.7858**	**0.4845**
ItemKNN CBF ICM genres	0.0316	0.0445	0.2705	0.1012
ItemKNN CBF ICM year	-	-	-	-
UserKNN CBF	0.0817	0.1285	0.5087	0.2556
ItemKNN CFCBF ICM genres	0.1759	0.2507	0.7405	0.4526
ItemKNN CFCBF ICM year	0.1819	0.2583	0.7494	0.4598
UserKNN CFCBF	0.1887	0.2623	0.7641	0.4654
LightFM ItemHybrid ICM genres	0.1625	0.2405	0.7170	0.4283
LightFM ItemHybrid ICM year	0.1510	0.2200	0.6931	0.4116
LightFM UserHybrid	0.1921	**0.2799**	0.7663	**0.4815**
RGCF paper	**0.1986**	0.2565	0.7569	0.4429
RGCF original early-stopping	0.1887	0.2620	0.7634	0.4625
RGCF our early-stopping	0.1970	0.2710	0.7787	0.4758
RGCF our hyperparameters	0.1981	0.2763	0.7807	0.4813

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

Wu et al. [23] 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., memory-based, such as SVDpp, MF-BPR etc.) and proposes an *inductive* representation (i.e., model-based) of the user and item embeddings as a function of the embeddings of a selected subset of template user and items. Due to this, the number of learnable parameters used in INMO can be lower compared to memory-based matrix factorization models. INMO includes an annealing process for normalization as a hyperparameter. The original implementation is available on GitHub and the datasets are available in a Google Drive folder.[10]

5.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 18.

Amazon-Book: Is a dataset of book purchases on Amazon. Ratings ≥ 4 are transformed into implicit interactions with value 1, then a 10-core subgraph selection is applied.

Gowalla: Is a dataset collected from a social network where users check-in locations they visited. No details are provided on the preprocessing.

Yelp2018: Is a business reviews dataset. Ratings ≥ 4 are transformed into implicit interactions with value 1, then a 10-core subgraph selection is applied.

Dataset	Interactions	Items	Users	Sparsity
Amazon-Book	2780441	96421	109730	0.9997
Gowalla	900713	40988	29858	0.9993
Yelp2018	1680930	42706	75173	0.9995

Table 18. Dataset statistics for INMO.

5.2 Results

The hyperparameter values used in our experiments are reported in Table 19 and the results for all the datasets and baseline algorithms are reported in Table 20 (Gowalla), 21 (Amazon-Book), and 22 (Yelp2018).

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

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 19. Hyperparameter values for INMO.

Table 20. Experimental results for the INMO method for the Gowalla dataset.

	Cutoff 20		
	Recall	Precision	NDCG
Random	0.0005	0.0002	0.0004
TopPop	0.0303	0.0083	0.0208
GlobalEffects	0.0003	0.0001	0.0002
UserKNN CF	0.1834	0.0493	0.1376
ItemKNN CF	0.1908	0.0508	0.1431
$P^3\alpha$	<u>**0.2065**</u>	<u>**0.0558**</u>	<u>**0.1548**</u>
$RP^3\beta$	<u>**0.2029**</u>	<u>**0.0548**</u>	**0.1523**
GF-CF	**0.2014**	**0.0525**	**0.1483**
$EASE^R$	-	-	-
SLIM BPR	0.1906	0.0502	0.1437
SLIM	<u>**0.2037**</u>	<u>**0.0574**</u>	<u>**0.1573**</u>
NegHOSLIM	-	-	-
NegHOSLIM (EN)	0.1934	**0.0526**	**0.1478**
MF-BPR	0.1308	0.0350	0.0979
MF-WARP	0.1351	0.0350	0.0970
SVDpp	0.1691	0.0433	0.1258
PureSVD	0.1289	0.0377	0.0984
NMF	0.1295	0.0361	0.0958
iALS	0.1820	0.0491	0.1362
LightFM CF	0.1832	0.0493	0.1360
MultVAE	<u>**0.2079**</u>	<u>**0.0555**</u>	<u>**0.1563**</u>
INMO paper	**0.2017**	**0.0536**	**0.1541**
INMO original early-stopping	0.1961	0.0523	0.1456
INMO our early-stopping	0.1961	0.0523	0.1455

Table 21. Experimental results for the INMO method for the Amazon-Book dataset.

	Cutoff 20		
	Recall	Precision	NDCG
Random	0.0002	0.0001	0.0001
TopPop	0.0114	0.0024	0.0069
GlobalEffects	0.0001	0.0000	0.0001
UserKNN CF	**0.1661**	**0.0353**	**0.1193**
ItemKNN CF	**0.1880**	**0.0420**	**0.1379**
$P^3\alpha$	**0.1933**	**0.0416**	**0.1376**
$RP^3\beta$	**0.1946**	**0.0418**	**0.1402**
GF-CF	**0.1726**	**0.0364**	**0.1222**
$EASE^R$	-	-	-
SLIM BPR	-	-	-
SLIM	**0.2006**	**0.0445**	**0.1451**
NegHOSLIM	-	-	-
NegHOSLIM (EN)	**0.1947**	**0.0425**	**0.1408**
MF-BPR	0.0876	0.0178	0.0597
MF-WARP	0.0875	0.0178	0.0590
SVDpp	0.1160	0.0225	0.0741
PureSVD	0.0852	0.0204	0.0619
NMF	0.0589	0.0140	0.0415
iALS	**0.1447**	0.0290	**0.0941**
LightFM CF	0.1370	0.0293	0.0920
MultVAE	**0.1751**	**0.0374**	**0.1241**
INMO paper	**0.1428**	**0.0301**	**0.0986**
INMO original early-stopping	0.1394	0.0298	0.0934
INMO our early-stopping	0.1395	0.0297	0.0932

Table 22. Experimental results for the INMO method for the Yelp2018 dataset.

	Cutoff 20		
	Recall	Precision	NDCG
Random	0.0004	0.0001	0.0002
TopPop	0.0171	0.0035	0.0102
GlobalEffects	0.0002	0.0001	0.0001
UserKNN CF	0.0846	0.0188	0.0545
ItemKNN CF	0.0901	0.0205	0.0584
$P^3\alpha$	0.0892	0.0194	0.0564
$RP^3\beta$	0.0907	0.0204	0.0583
GF-CF	0.0979	0.0213	0.0620
$EASE^R$	-	-	-
SLIM BPR	-	-	-
SLIM	0.0870	0.0201	0.0571
NegHOSLIM	-	-	-
NegHOSLIM (EN)	0.0861	0.0194	0.0557
MF-BPR	0.0539	0.0123	0.0332
MF-WARP	0.0451	0.0106	0.0290
SVDpp	0.0835	0.0180	0.0507
PureSVD	0.0638	0.0151	0.0410
NMF	0.0638	0.0146	0.0397
iALS	0.0994	0.0220	0.0635
LightFM CF	0.0887	0.0196	0.0555
MultVAE	**0.1048**	**0.0231**	**0.0670**
INMO paper	**0.1026**	0.0225	**0.0651**
INMO original early-stopping	0.1025	0.0226	0.0647
INMO our early-stopping	0.1025	0.0225	0.0646

6 HYPERGRAPH CONTRASTIVE COLLABORATIVE FILTERING

Xia et al. [24] 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 original implementation is available on GitHub.[11]

6.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 23.

- **Yelp:** Is a business reviews dataset. The preprocessing is a 10-cores subgraph selection.
- **MovieLens10M:** Is a movie recommendation dataset. The preprocessing is a 10-cores subgraph selection.
- **Amazon-Book:** Is a dataset of book purchases on Amazon. The preprocessing is a 20-cores subgraph selection.

Dataset	Interactions	Items	Users	Sparsity
Yelp	1527326	24734	29601	0.9979
MovieLens10M	9998816	10196	69878	0.9860
Amazon-Book	3200224	77801	78578	0.9995

Table 23. Dataset statistics for HCCF.

6.2 Results

The hyperparameter values used in our experiments are reported in Table 24 and the results for all the datasets and baseline algorithms are reported in Table 25 (Yelp2018), 26 (Amazon-Book), and 27 (MovieLens10M).

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

Hyperparameter	Described in	Value			
		All datasets	MovieLens10M	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-Book 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 24. Hyperparameter values for HCCF.

Table 25. Experimental results for the HCCF method for the Yelp2018 dataset.

	Cutoff 20		Cutoff 40	
	Recall	NDCG	Recall	NDCG
Random	0.0009	0.0007	0.0016	0.0010
TopPop	0.0123	0.0109	0.0208	0.0141
GlobalEffects	0.0008	0.0006	0.0014	0.0008
UserKNN CF	**0.0973**	**0.0848**	**0.1527**	**0.1049**
ItemKNN CF	**0.1054**	**0.0922**	**0.1646**	**0.1134**
$P^3\alpha$	**0.1054**	**0.0921**	**0.1640**	**0.1132**
$RP^3\beta$	**0.1082**	**0.0947**	**0.1687**	**0.1165**
GF-CF	**0.1115**	**0.0966**	**0.1752**	**0.1196**
$EASE^R$	-	-	-	-
SLIM BPR	-	-	-	-
SLIM	**0.1062**	**0.0935**	**0.1642**	**0.1144**
NegHOSLIM	-	-	-	-
NegHOSLIM (EN)	**0.1032**	**0.0908**	**0.1602**	**0.1112**
MF-BPR	0.0601	0.0507	0.0990	0.0650
MF-WARP	0.0569	0.0480	0.0945	0.0618
SVDpp	**0.0928**	**0.0788**	**0.1504**	**0.0998**
PureSVD	**0.0863**	**0.0757**	**0.1360**	**0.0935**
NMF	**0.0761**	**0.0657**	**0.1229**	**0.0826**
iALS	**0.1090**	**0.0948**	**0.1706**	**0.1171**
LightFM CF	**0.0959**	**0.0822**	**0.1540**	**0.1034**
MultVAE	**0.1172**	**0.1029**	**0.1825**	**0.1264**
HCCF paper	0.0607	0.0510	0.1007	0.0658
HCCF our early-stopping	0.0609	0.0525	0.1011	0.0672
HCCF provided number of epochs	0.0536	0.0459	0.0899	0.0592

Table 26. Experimental results for the HCCF method for the Amazon-Book dataset.

	Cutoff 20		Cutoff 40	
	Recall	NDCG	Recall	NDCG
Random	0.0002	**0.0002**	0.0005	0.0003
TopPop	**0.0097**	**0.0073**	**0.0157**	**0.0093**
GlobalEffects	0.0001	0.0001	0.0003	0.0002
UserKNN CF	**0.1476**	**0.1278**	**0.1973**	**0.1434**
ItemKNN CF	**0.1726**	**0.1502**	**0.2209**	**0.1650**
$P^3\alpha$	**0.1705**	**0.1427**	**0.2300**	**0.1617**
$RP^3\beta$	**0.1773**	**0.1510**	**0.2377**	**0.1702**
GF-CF	**0.1580**	**0.1344**	**0.2131**	**0.1520**
$EASE^R$	-	-	-	-
SLIM BPR	-	-	-	-
SLIM	**0.1885**	**0.1626**	**0.2458**	**0.1803**
NegHOSLIM	-	-	-	-
NegHOSLIM (EN)	**0.1811**	**0.1558**	**0.2363**	**0.1728**
MF-BPR	**0.0687**	**0.0532**	**0.1012**	**0.0640**
MF-WARP	**0.0713**	**0.0557**	**0.1050**	**0.0669**
SVDpp	**0.0980**	**0.0717**	**0.1482**	**0.0885**
PureSVD	**0.0820**	**0.0718**	**0.1177**	**0.0827**
NMF	**0.0662**	**0.0572**	**0.0974**	**0.0670**
iALS	**0.1185**	**0.0928**	**0.1744**	**0.1113**
LightFM CF	**0.1166**	**0.0880**	**0.1719**	**0.1062**
MultVAE	**0.1474**	**0.1222**	**0.2038**	**0.1405**
HCCF paper	**0.0344**	**0.0258**	**0.0561**	**0.0330**
HCCF our early-stopping	0.0002	0.0001	0.0004	0.0002
HCCF provided number of epochs	0.0002	0.0001	0.0008	0.0003

Table 27. Experimental results for the HCCF method for the MovieLens10M dataset.

	Cutoff 20		Cutoff 40	
	Recall	NDCG	Recall	NDCG
Random	0.0019	0.0032	0.0039	0.0038
TopPop	0.1363	0.1903	0.2114	0.2022
GlobalEffects	0.0001	0.0003	0.0002	0.0003
UserKNN CF	**0.3503**	**0.4448**	**0.4700**	**0.4595**
ItemKNN CF	0.2816	0.3645	0.3884	0.3790
$P^3\alpha$	0.2576	0.3263	0.3521	0.3391
$RP^3\beta$	0.2886	**0.3761**	0.3960	0.3895
GF-CF	**0.3342**	**0.4210**	**0.4484**	**0.4354**
$EASE^R$	-	-	-	-
SLIM BPR	-	-	-	-
SLIM	**0.3387**	**0.4422**	**0.4578**	**0.4563**
NegHOSLIM	-	-	-	-
NegHOSLIM (EN)	**0.3430**	**0.4430**	**0.4630**	**0.4582**
MF-BPR	0.2849	0.3569	0.3989	0.3759
MF-WARP	0.2823	0.3529	0.3964	0.3724
SVDpp	**0.3391**	**0.4171**	**0.4672**	**0.4381**
PureSVD	**0.3090**	**0.4032**	**0.4212**	**0.4166**
NMF	0.2800	0.3627	0.3853	0.3765
iALS	**0.3368**	**0.4232**	**0.4593**	**0.4426**
LightFM CF	**0.3310**	**0.4183**	**0.4528**	**0.4367**
MultVAE	**0.3563**	**0.4291**	**0.4840**	**0.4547**
HCCF paper	0.2048	0.2467	0.3081	0.2717
HCCF our early-stopping	0.2904	0.3754	0.4086	0.3945
HCCF provided number of epochs	0.2714	0.3605	0.3911	0.3798

7 HAKG: HIERARCHY-AWARE KNOWLEDGE GATED NETWORK FOR RECOMMENDATION

Du et al. [4] 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. The paper claims that it is not sufficient to use a Euclidean space for this purpose, 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 the embeddings involved, aiming at better preserving their 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 original implementation is available on GitHub.[12]

7.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 28.

Alibaba-iFashion: Is a datasets of outfits for garment recommendation. The data is preprocessed with 10-cores subgraph selection.

Yelp2018: Is a business reviews dataset. The data is preprocessed with 10-cores subgraph selection.

Last-FM: Is a dataset for song recommendation. The split is the same used in KGAT, including the knowledge base [22]. The preprocessing filters the data retaining the interactions from Jan 2015 to June 2015, followed by 10-cores subgraph selection.

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

Dataset	Interactions	Items	Users	Sparsity
Alibaba-iFashion	1781093	30040	114737	0.9995
Yelp2018	1183610	45538	45919	0.9994
Last-FM	1542856	48123	23566	0.9986

Table 28. Dataset statistics for HAKG.

7.2 Results

The hyperparameter values used in our experiments are reported in Table 29 and the results for all the datasets and baseline algorithms are reported in Table 30 (Alibaba-iFashion), 31 (Yelp2018), and 32 (Last-FM).

[12] 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 29. Hyperparameter values for HAKG.

Table 30. Experimental results for the HAKG method for the Alibaba iFashion dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0007	0.0004
TopPop	0.0312	0.0167
GlobalEffects	0.0002	0.0001
UserKNN CF	0.1090	0.0700
ItemKNN CF	**0.1264**	**0.0818**
$P^3\alpha$	0.1219	0.0779
$RP^3\beta$	0.1247	**0.0807**
GF-CF	0.1182	0.0742
$EASE^R$	0.1262	**0.0819**
SLIM BPR	0.1208	0.0776
SLIM	**0.1276**	**0.0832**
NegHOSLIM	-	-
NegHOSLIM (EN)	0.1259	**0.0822**
MF-BPR	0.0761	0.0460
MF-WARP	0.0773	0.0466
SVDpp	0.1183	0.0740
PureSVD	0.0687	0.0434
NMF	0.0716	0.0429
iALS	**0.1268**	**0.0807**
LightFM CF	0.1183	0.0737
MultVAE	<u>**0.1388**</u>	<u>**0.0898**</u>
ItemKNN CBF	0.0099	0.0057
ItemKNN CFCBF	**0.1273**	**0.0820**
LightFM ItemHybrid	0.0553	0.0328
HAKG paper	**0.1319**	**0.0848**
HAKG original early-stopping	0.1261	0.0787
HAKG our early-stopping	0.1263	0.0789

Table 31. Experimental results for the HAKG method for the Yelp2018 dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0004	0.0002
TopPop	0.0174	0.0110
GlobalEffects	0.0006	0.0003
UserKNN CF	0.0715	0.0477
ItemKNN CF	0.0727	0.0485
$P^3\alpha$	0.0728	0.0480
$RP^3\beta$	0.0733	0.0485
GF-CF	0.0752	0.0492
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	0.0739	0.0494
NegHOSLIM	-	-
NegHOSLIM (EN)	0.0692	0.0465
MF-BPR	0.0484	0.0310
MF-WARP	0.0443	0.0280
SVDpp	0.0613	0.0389
PureSVD	0.0546	0.0364
NMF	0.0510	0.0339
iALS	0.0764	0.0495
LightFM CF	0.0705	0.0454
MultVAE	<u>0.0799</u>	<u>0.0521</u>
ItemKNN CBF	0.0272	0.0175
ItemKNN CFCBF	0.0743	0.0492
LightFM ItemHybrid	0.0516	0.0331
HAKG paper	0.0778	0.0501

Table 32. Experimental results for the HAKG method for the Last-FM dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0004	0.0005
TopPop	0.0229	0.0198
GlobalEffects	0.0004	0.0003
UserKNN CF	**0.1720**	**0.1695**
ItemKNN CF	**0.1836**	**0.1838**
$P^3\alpha$	**0.1979**	**0.1994**
$RP^3\beta$	**0.2012**	**0.2014**
GF-CF	**0.1806**	**0.1729**
$EASE^R$	-	-
SLIM BPR	**0.1861**	**0.1877**
SLIM	**0.2070**	**0.2078**
NegHOSLIM	-	-
NegHOSLIM (EN)	**0.2049**	**0.2058**
MF-BPR	0.1281	0.1250
MF-WARP	0.1337	0.1322
SVDpp	**0.1745**	0.1660
PureSVD	0.1314	0.1369
NMF	0.1086	0.1146
iALS	**0.1750**	0.1645
LightFM CF	**0.1816**	**0.1716**
MultVAE	**0.1884**	**0.1838**
ItemKNN CBF	**0.1887**	**0.1797**
ItemKNN CFCBF	**0.1849**	**0.1838**
LightFM ItemHybrid	**0.1945**	**0.1865**
HAKG paper	0.1008	0.0931
HAKG original early-stopping	0.1655	0.1644
HAKG our early-stopping	0.1693	0.1687

8 GRAPH TREND FILTERING NETWORKS FOR RECOMMENDATION

Fan et al. [5] 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 original implementation is available on GitHub.[13]

8.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 33.

Gowalla: Is a dataset collected from a social network where users check-in locations they visited. The split is the same used in LightGCN [7], see Section 10.

Yelp2018: Is a business reviews dataset. The split is the same used in LightGCN [7], see Section 10.

Amazon-Book: Is a dataset of book purchases on Amazon. The split is the same used in LightGCN [7], see Section 10.

Last-FM: Is a dataset for song recommendation. The split is the same used in KGAT, including the knowledge base [22]. The preprocessing filters the data retaining the interactions from Jan 2015 to June 2015, followed by 10-cores subgraph selection.

Dataset	Interactions	Items	Users	Sparsity
Gowalla	1027370	40981	29858	0.9992
Yelp2018	1561406	38048	31668	0.9987
Amazon-Book	2984108	91599	52643	0.9994
Last-FM	1542856	48123	23566	0.9986

Table 33. Dataset statistics for GTN.

8.2 Results

The hyperparameter values used in our experiments are reported in Table 34 and the results for all the datasets and baseline algorithms are reported in Table 35 (Yelp2018), 36 (Amazon-Book), 37 (Gowalla), and 38 (Last-FM).

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

Hyperparameter	Described in	Value				
		All datasets	Gowalla	Yelp2018	Amazon-Book	Last-FM
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]In the source code it is called *lambda2*.
[b]In the source code it is called *keep_prob* and is 0.6, hence dropout is 0.4.
[c]In the source code is called *prop_dropout*.

Table 34. Hyperparameter values for GTN.

Table 35. Experimental results for the GTN method for the Yelp2018 dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0005	0.0004
TopPop	0.0124	0.0101
GlobalEffects	0.0006	0.0004
UserKNN CF	0.0637	0.0533
ItemKNN CF	0.0622	0.0514
$P^3\alpha$	0.0661	0.0548
$RP^3\beta$	0.0672	<u>0.0558</u>
GF-CF	**<u>0.0693</u>**	**<u>0.0568</u>**
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	0.0646	0.0541
NegHOSLIM	-	-
NegHOSLIM (EN)	0.0590	0.0492
MF-BPR	0.0382	0.0313
MF-WARP	0.0415	0.0339
SVDpp	0.0606	0.0492
PureSVD	0.0537	0.0448
NMF	0.0517	0.0424
iALS	0.0667	0.0546
LightFM CF	0.0592	0.0482
MultVAE	**<u>0.0719</u>**	**<u>0.0590</u>**
GTN paper	0.0679	0.0554
GTN our early-stopping	<u>0.0679</u>	<u>0.0559</u>

Table 36. Experimental results for the GTN method for the Amazon-Book dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0002	0.0002
TopPop	0.0051	0.0044
GlobalEffects	0.0004	0.0003
UserKNN CF	**0.0616**	**0.0518**
ItemKNN CF	**0.0750**	**0.0624**
$P^3\alpha$	**0.0696**	**0.0561**
$RP^3\beta$	**0.0701**	**0.0585**
GF-CF	**0.0710**	**0.0585**
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	**0.0757**	**0.0600**
NegHOSLIM	-	-
NegHOSLIM (EN)	**0.0754**	**0.0609**
MF-BPR	0.0254	0.0203
MF-WARP	0.0288	0.0230
SVDpp	0.0379	0.0293
PureSVD	0.0403	0.0336
NMF	0.0341	0.0287
iALS	0.0451	0.0347
LightFM CF	**0.0501**	**0.0384**
MultVAE	**0.0553**	**0.0435**
GTN paper	0.0450	0.0346
GTN our early-stopping	0.0496	0.0384

Table 37. Experimental results for the GTN method for the Gowalla dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0005	0.0003
TopPop	0.0416	0.0317
GlobalEffects	0.0007	0.0004
UserKNN CF	0.1699	0.1387
ItemKNN CF	0.1559	0.1228
$P^3\alpha$	0.1838	0.1526
$RP^3\beta$	0.1811	0.1490
GF-CF	0.1843	0.1505
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	0.1767	0.1448
NegHOSLIM	-	-
NegHOSLIM (EN)	0.1723	0.1410
MF-BPR	0.1319	0.1060
MF-WARP	0.1266	0.0992
SVDpp	0.1611	0.1298
PureSVD	0.1135	0.0917
NMF	0.1278	0.1063
iALS	0.1669	0.1370
LightFM CF	0.1783	0.1468
MultVAE	<u>**0.1873**</u>	0.1539
GTN paper	**0.1870**	**0.1588**
GTN our early-stopping	0.1849	**0.1565**

Table 38. Experimental results for the GTN method for the Last-FM dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0004	0.0005
TopPop	0.0229	0.0198
GlobalEffects	0.0004	0.0003
UserKNN CF	0.1720	0.1695
ItemKNN CF	**0.1836**	**0.1838**
$P^3\alpha$	**0.1979**	**0.1994**
$RP^3\beta$	**0.2012**	**0.2014**
GF-CF	**0.1806**	0.1729
$EASE^R$	-	-
SLIM BPR	**0.1861**	**0.1877**
SLIM	**0.2070**	**0.2078**
NegHOSLIM	-	-
NegHOSLIM (EN)	**0.2049**	**0.2058**
MF-BPR	0.1281	0.1250
MF-WARP	0.1337	0.1322
SVDpp	0.1745	0.1660
PureSVD	0.1314	0.1369
NMF	0.1086	0.1146
iALS	0.1750	0.1645
LightFM CF	**0.1816**	0.1716
MultVAE	**0.1884**	**0.1838**
ItemKNN CBF	**0.1887**	**0.1797**
ItemKNN CFCBF	**0.1849**	**0.1838**
LightFM ItemHybrid	**0.1945**	**0.1865**
GTN paper	0.0932	0.0857
GTN our early-stopping	0.1773	0.1776

9 KNOWLEDGE GRAPH CONTRASTIVE LEARNING FOR RECOMMENDATION

Yang et al. [25] 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 original implementation is available on GitHub.[14]

9.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 39.

Yelp2018: Is a business reviews dataset. The preprocessing is a 10-cores subgraph selection. The split is the same used in HAKG [4]. The entities are collected in the same way as KGAT [22]

Amazon-Book: Is a dataset of book purchases on Amazon. The preprocessing is a 10-cores subgraph selection. The entities are collected in the same way as KGAT [22]

MIND: is a news recommendation dataset. The data and knowledge base are collected in the same way as [21], by randomly sampling one million users who had at least 5 news clicks during six weeks (i.e., October 12 to November 22, 2019).

Dataset	Interactions	Items	Users	Sparsity
Amazon-Book	846434	24915	70679	0.9995
Yelp2018	1183610	45538	45919	0.9994
MIND	2545327	48957	300000	0.9998

Table 39. Dataset statistics for KGCL.

9.2 Results

The hyperparameter values used in our experiments are reported in Table 40 and the results for all the datasets and baseline algorithms are reported in Table 41 (Amazon-Book), 42 (Yelp2018), and 43 (MIND).

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

Hyperparameter	Described in	Value			
		All datasets	Amazon-Book	Yelp2018	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]For this dataset the source code uses a different value compared to the paper.
[b]This hyperparameter appears to be used to add random samples as part of the user interaction dropout process, only when *uicontrast* is "weighted-mix".
[c]This hyperparameter is never used because it is not used when uicontrast is "weighted".
[d]Defined as 1-ui_p_drop.
[e]This hyperparameter could impact how the graph augmentations are generated for the contrastive learning part, but the 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]This hyperparameter has no impact because the epochs never reach 1500.
[h]This hyperparameter has no impact because the epochs never reach 1500.
[i]The patience and minimum number of epochs are different across the datasets, but the paper does not describe how were those values determined.

Table 40. Hyperparameter values for KGCL.

Table 41. Experimental results for the KGCL method for the Amazon-Book dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0007	0.0003
TopPop	0.0287	0.0123
GlobalEffects	0.0004	0.0002
UserKNN CF	**0.1658**	**0.0944**
ItemKNN CF	**0.1653**	**0.0974**
$P^3\alpha$	**0.1683**	**0.0958**
$RP^3\beta$	**0.1706**	**0.0983**
GF-CF	**0.1712**	**0.0973**
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	**0.1742**	**0.1031**
NegHOSLIM	-	-
NegHOSLIM (EN)	**0.1740**	**0.1028**
MF-BPR	0.1143	0.0637
MF-WARP	0.1103	0.0602
SVDpp	**0.1569**	**0.0848**
PureSVD	0.1059	0.0620
NMF	0.0953	0.0548
iALS	**0.1676**	**0.0908**
LightFM CF	**0.1573**	**0.0830**
MultVAE	**0.1707**	**0.0953**
ItemKNN CBF	0.1125	0.0640
ItemKNN CFCBF	**0.1698**	**0.0992**
LightFM ItemHybrid	0.1414	0.0762
KGCL paper	**0.1496**	0.0793
KGCL our early-stopping	0.1478	0.0794

Table 42. Experimental results for the KGCL method for the Yelp2018 dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0004	0.0002
TopPop	0.0174	0.0110
GlobalEffects	0.0006	0.0003
UserKNN CF	0.0715	0.0477
ItemKNN CF	0.0727	**0.0485**
$P^3\alpha$	0.0728	**0.0480**
$RP^3\beta$	**0.0733**	**0.0485**
GF-CF	**0.0752**	**0.0492**
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	**0.0739**	**<u>0.0494</u>**
NegHOSLIM	-	-
NegHOSLIM (EN)	0.0692	0.0465
MF-BPR	0.0484	0.0310
MF-WARP	0.0443	0.0280
SVDpp	0.0613	0.0389
PureSVD	0.0546	0.0364
NMF	0.0510	0.0339
iALS	**<u>0.0764</u>**	**<u>0.0495</u>**
LightFM CF	0.0705	0.0454
MultVAE	**<u>0.0799</u>**	**<u>0.0521</u>**
ItemKNN CBF	0.0272	0.0175
ItemKNN CFCBF	**0.0743**	**0.0492**
LightFM ItemHybrid	0.0516	0.0331
KGCL paper	**0.0756**	**0.0493**
KGCL our early-stopping	0.0729	0.0477

Table 43. Experimental results for the KGCL method for the MIND dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0004	0.0002
TopPop	0.0894	0.0437
GlobalEffects	0.0009	0.0003
UserKNN CF	0.0972	0.0509
ItemKNN CF	**0.1225**	**0.0647**
$P^3\alpha$	**0.1189**	**0.0621**
$RP^3\beta$	**0.1187**	**0.0621**
GF-CF	**0.1017**	0.0524
$EASE^R$	-	-
SLIM BPR	**0.1193**	**0.0627**
SLIM	**0.1287**	**0.0686**
NegHOSLIM	-	-
NegHOSLIM (EN)	**0.1281**	**0.0681**
MF-BPR	0.0888	0.0435
MF-WARP	0.0842	0.0389
SVDpp	0.0989	0.0489
PureSVD	0.0892	0.0436
NMF	0.0894	0.0437
iALS	**0.1130**	**0.0600**
LightFM CF	**0.1044**	0.0522
MultVAE	**0.1321**	**0.0700**
ItemKNN CBF	0.0051	0.0024
ItemKNN CFCBF	**0.1076**	**0.0556**
LightFM ItemHybrid	0.0688	0.0320
KGCL paper	**0.1073**	**0.0551**
KGCL our early-stopping	0.1006	0.0531

10 LIGHTGCN: SIMPLIFYING AND POWERING GRAPH CONVOLUTION NETWORK FOR RECOMMENDATION

He et al. [7] proposes LightGCN, a graph-based collaborative filtering method in which the user and item embeddings are propagated according to the graph adjacency matrix. LightGCN is presented as a "light" model based on message-passing, compared to previous more complex architectures. The original implementation is available on Github.[15]

10.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 44

- **Amazon Book:** Is a dataset of book purchases on Amazon. The preprocessing applies a 10-cores subgraph selection.
- **Gowalla:** Is a dataset collected from a social network where users check-in locations they visited. The preprocessing applies a 10-cores subgraph selection.
- **Yelp2018:** Is a business reviews dataset. The preprocessing applies a 10-cores subgraph selection.

Dataset	Interactions	Items	Users	Sparsity
Amazon Book	2984108	91599	52643	0.9994
Gowalla	1027370	40981	29858	0.9992
Yelp2018	1561406	38048	31668	0.9987

Table 44. Dataset statistics for LightGCN.

10.2 Results

The hyperparameter values used in our experiments are reported in Table 45 and the results for all the datasets and baseline algorithms are reported in Table 46 (Gowalla), 47 (Amazon-Book Original Split), 48 (Amazon-Book Our Split), 49 (Yelp 2018 Original Split), and 50 (Yelp 2018 Our Split).

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 45. Hyperparameter values for LightGCN.

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

Table 46. Experimental results for the LightGCN method for the Gowalla dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0005	0.0003
TopPop	0.0416	0.0317
GlobalEffects	0.0007	0.0004
UserKNN CF	0.1699	0.1387
ItemKNN CF	0.1559	0.1228
$P^3\alpha$	**<u>0.1838</u>**	0.1526
$RP^3\beta$	**0.1811**	0.1490
GF-CF	**<u>0.1843</u>**	0.1505
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	0.1767	0.1448
NegHOSLIM	-	-
NegHOSLIM (EN)	0.1723	0.1410
MF-BPR	0.1319	0.1060
MF-WARP	0.1266	0.0992
SVDpp	0.1611	0.1298
PureSVD	0.1135	0.0917
NMF	0.1278	0.1063
iALS	0.1669	0.1370
LightFM CF	0.1783	0.1468
MultVAE	**<u>0.1873</u>**	**0.1539**
LightGCN paper	**0.1830**	**0.1550**
LightGCN original early-stopping	0.1798	0.1536
LightGCN our early-stopping	0.1772	0.1519

Table 47. Experimental results for the LightGCN method for the Amazon Book Original Split dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0002	0.0002
TopPop	0.0051	0.0044
GlobalEffects	0.0004	0.0003
UserKNN CF	**0.0616**	**0.0518**
ItemKNN CF	**0.0750**	**0.0624**
$P^3\alpha$	**0.0696**	**0.0561**
$RP^3\beta$	**0.0701**	**0.0585**
GF-CF	**0.0710**	**0.0585**
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	**0.0757**	**0.0600**
NegHOSLIM	-	-
NegHOSLIM (EN)	**0.0754**	**0.0609**
MF-BPR	0.0254	0.0203
MF-WARP	0.0288	0.0230
SVDpp	0.0379	0.0293
PureSVD	0.0403	**0.0336**
NMF	0.0341	0.0287
iALS	**0.0451**	**0.0347**
LightFM CF	**0.0501**	**0.0384**
MultVAE	**0.0553**	**0.0435**
LightGCN paper	0.0406	0.0313
LightGCN original early-stopping	0.0407	0.0315
LightGCN our early-stopping	0.0409	0.0317

Table 48. Experimental results for the LightGCN method for the Amazon Book Our Split dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0002	0.0002
TopPop	0.0093	0.0079
GlobalEffects	0.0002	0.0002
UserKNN CF	**0.1396**	**0.1357**
ItemKNN CF	**0.1719**	**0.1680**
$P^3\alpha$	**0.1688**	**0.1596**
$RP^3\beta$	**0.1652**	**0.1583**
GF-CF	**0.1543**	**0.1470**
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	**0.1880**	**0.1838**
NegHOSLIM	-	-
NegHOSLIM (EN)	**0.1778**	**0.1730**
MF-BPR	0.0589	0.0518
MF-WARP	0.0555	0.0485
SVDpp	0.0986	0.0818
PureSVD	0.0872	0.0853
NMF	0.0675	0.0655
iALS	**0.1157**	**0.1020**
LightFM CF	**0.1257**	**0.1117**
MultVAE	**0.1373**	**0.1272**
LightGCN original early-stopping	0.0997	0.0862
LightGCN our early-stopping	0.0968	0.0840

Table 49. Experimental results for the LightGCN method for the Yelp 2018 Original Split dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0005	0.0004
TopPop	0.0124	0.0101
GlobalEffects	0.0006	0.0004
UserKNN CF	**0.0637**	**0.0533**
ItemKNN CF	**0.0622**	**0.0514**
$P^3\alpha$	**0.0661**	**0.0548**
$RP^3\beta$	**0.0672**	**0.0558**
GF-CF	**0.0693**	**0.0568**
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	**0.0646**	**0.0541**
NegHOSLIM	-	-
NegHOSLIM (EN)	0.0590	0.0492
MF-BPR	0.0382	0.0313
MF-WARP	0.0415	0.0339
SVDpp	0.0606	0.0492
PureSVD	0.0537	0.0448
NMF	0.0517	0.0424
iALS	**0.0667**	**0.0546**
LightFM CF	0.0592	0.0482
MultVAE	**0.0719**	**0.0590**
LightGCN paper	**0.0649**	**0.0530**
LightGCN original early-stopping	0.0618	0.0506
LightGCN our early-stopping	0.0621	0.0510

Table 50. Experimental results for the LightGCN method for the Yelp 2018 Our Split dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0007	0.0005
TopPop	0.0160	0.0130
GlobalEffects	0.0004	0.0003
UserKNN CF	0.0937	**0.0811**
ItemKNN CF	**0.1002**	**0.0868**
$P^3\alpha$	**0.1002**	**0.0864**
$RP^3\beta$	**0.1029**	**0.0891**
GF-CF	**0.1043**	**0.0893**
$EASE^R$	-	-
SLIM BPR	-	-
SLIM	**0.0996**	**0.0866**
NegHOSLIM	-	-
NegHOSLIM (EN)	0.0904	0.0778
MF-BPR	0.0525	0.0435
MF-WARP	0.0511	0.0429
SVDpp	0.0841	0.0702
PureSVD	0.0792	0.0689
NMF	0.0701	0.0600
iALS	**0.1037**	**0.0891**
LightFM CF	**0.0982**	**0.0837**
MultVAE	**0.1126**	**0.0971**
LightGCN original early-stopping	0.0942	0.0802
LightGCN our early-stopping	0.0943	0.0803

11 COMPARISON OF THE ANALYZED METHODS OF SIGIR 2022

This section reports the details of the experimental protocol of the comparative analysis between all SIGIR 2022 methods we analyze.

11.1 Datasets

The evaluation is performed on the datasets described and processed as follows, their statistics are reported in Table 51:

Amazon Book: Is a dataset of book purchases on Amazon. The preprocessing applies a 10-cores subgraph selection. The entities for KGCL and HAKG are collected in the same way as KGAT [22].

Yelp2018: Is a business reviews dataset. The preprocessing applies a 10-cores subgraph selection. The entities for KGCL and HAKG are collected in the same way as KGAT [22].

Dataset	Interactions	Items	Users	Sparsity
Amazon Book	846434	24915	70679	0.9995
Yelp2018	1183610	45538	45919	0.9994

Table 51. Datasets statistics for the comparison of the analyzed methods of SIGIR 2022.

11.2 Hyperparameter Ranges

In this section we report the hyperparameter ranges and distribution for all the GNN algorithms we analyze, see Table 52 for the purely collaborative models and Table 53 for those including a Knowledge Base. Notice that some hyperparameters are searched for all models and are labeled as *Common Hyperparameters* in Table 52. Overall, for each model there are between 9 and 16 hyperparameters.

11.3 Results

The values of the optimized hyperparameters for our baselines are reported in Table 54 (Nearest-Neighbor Collaborative), Table 55 (Graph-based), Table 56 (Item-based Machine Learning), Table 57 (Matrix Factorization), and Table 58 (Autoencoder). The values of the optimized hyperparameters for the GNN algorithms we analyze are reported in Table 59 and 60 (Collaborative), and Table 61 (Knowledge Base).

The results for all the datasets and baseline algorithms are reported in 62 (Amazon-Book), and 63 (Yelp2018).

Algorithm	Hyperparameter	Range	Type	Distribution
Common Hyperparameters	epochs	1000	Categorical	-
	batch size	256, 512, 1024, 2048, 4096	Categorical	-
	learning rate	10^{-6} - 10^{-1}	Real	log-uniform
	embedding size	2 - 350	Integer	uniform
	optimizer	Adam	-	-
GDE	beta	10^{-1} - 10^{+2}	Real	log-uniform
	feature type	smoothed, both	Categorical	-
	smooth ratio	$\frac{1}{\text{min(n\_users, n\_items)-1}}$ - 0.1	Real	log-uniform
	rough ratio	$\frac{1}{\text{min(n\_users, n\_items)-1}}$ - 0.1	Real	log-uniform
	loss type	adaptive, bpr	Categorical	-
	dropout rate	0.1 - 0.9	Real	uniform
	regularization rate	10^{-6} - 10^{-1}	Real	log-uniform
GTN	GNN layers K	1 - 6	Integer	uniform
	embedding smoothness weight	1 - 15	Integer	uniform
	l2 reg	10^{-6} - 10^{-1}	Real	log-uniform
	dropout rate GTN	0.1 - 0.9	Real	uniform
	dropout rate LightGCN	0.1 - 0.9	Real	uniform
HCCF	GNN layers K	1 - 6	Integer	uniform
	HYP layers C	1 - 4	Integer	uniform
	hyperedge size	2, 350	Integer	uniform
	dropout rate	0.1 - 0.9	Real	uniform
	l2 reg	10^{-6} - 10^{-1}	Real	log-uniform
	contrastive loss temperature τ	10^{-2} - 10^{0}	Real	log-uniform
	contrastive loss weight	10^{-7} - 10^{-1}	Real	log-uniform
	leaky relu slope	0.01	-	-
INMO	GNN layers K	1 - 6	Integer	uniform
	l2 reg	10^{-6} - 10^{-1}	Real	log-uniform
	template loss weight	10^{-4} - 10^{-1}	Real	log-uniform
	template node ranking metric	degree, sort, page rank	Categorical	-
	dropout rate	0.1 - 0.9	Real	uniform
	template ratio	0.1 - 1.0	Real	uniform
	normalization decay	0.99	-	-
RGCF	GNN layers K	1 - 6	Integer	uniform
	prune threshold β	10^{-3} - 10^{0}	Real	log-uniform
	contrastive loss temperature τ	10^{-2} - 10^{0}	Real	log-uniform
	contrastive loss weight	10^{-7} - 10^{-1}	Real	log-uniform
	augmentation ratio	0.01 - 0.3	Real	log-uniform
	l2 reg	10^{-6} - 10^{-1}	Real	log-uniform
SimGCL	GNN layers K	1 - 6	Integer	uniform
	noise magnitude ϵ	10^{-2} - 10^{0}	Real	log-uniform
	contrastive loss temperature τ	10^{-2} - 10^{0}	Real	log-uniform
	contrastive loss weight	10^{-7} - 10^{-1}	Real	log-uniform
	l2 reg	10^{-6} - 10^{-1}	Real	log-uniform
LightGCN	GNN layers K	1 - 6	Integer	uniform
	l2 reg	10^{-6} - 10^{-1}	Real	log-uniform
	dropout rate	0.1 - 0.9	Real	uniform

Table 52. Hyperparameter ranges and distributions for the purely collaborative GNN models.

Algorithm	Hyperparameter	Range	Type	Distribution
HAKG	GNN layers K	1 - 6	Integer	uniform
	angle loss weight	10^{-6} - 10^{-1}	Real	log-uniform
	l2 reg	10^{-6} - 10^{-1}	Real	log-uniform
	add KB inverse relation	True, False	Categorical	-
	dropout rate node	0.1 - 0.9	Real	uniform
	dropout rate mess	0.1 - 0.9	Real	uniform
	dropout rate angle	0.1 - 0.9	Real	uniform
	n negative samples M	100, 500	Integer	uniform
	contrastive loss margin	0.1 - 0.9	Real	uniform
KGCL	GNN layers K	1 - 6	Integer	uniform
	contrastive loss temperature τ	10^{-2} - 10^{0}	Real	log-uniform
	GNN dropout rate	0.1 - 0.9	Real	uniform
	knowledge graph dropout rate	0.1 - 0.9	Real	uniform
	user interaction dropout rate	0.1 - 0.9	Real	uniform
	mix ratio	0.1 - 0.9	Real	uniform
	uicontrast	weighted, weighted-mix	Categorical	-
	entities per head	1, 20	Integer	uniform
	l2 reg	10^{-6} - 10^{-1}	Real	log-uniform
	self supervised loss weight	10^{-4} - 10^{-1}	Real	log-uniform

Table 53. Hyperparameter ranges and distributions for the GNN models that include a knowledge base.

Table 54. Selected hyperparameter values for our Nearest-Neighbor Collaborative baselines.

Algorithm	Hyperparameter	Yelp2018	Amazon-Book
UserKNN CF	topK	397	636
	shrink	0	1000
	similarity	cosine	cosine
	normalize	True	True
	feature weighting	TF-IDF	TF-IDF
ItemKNN CF	topK	996	1000
	shrink	993	1000
	similarity	cosine	cosine
	normalize	True	True
	feature weighting	TF-IDF	TF-IDF

Table 55. Selected hyperparameter values for our Graph-based baselines.

Algorithm	Hyperparameter	Yelp2018	Amazon-Book
P3alpha	topK	401	201
	alpha	0.6278	0.4115
	normalize similarity	False	True
$RP^3\beta$	topK	927	680
	alpha	0.3973	0.4485
	beta	0.2520	0.2321
	normalize similarity	False	False
GF-CF	-	-	-

Table 56. Selected hyperparameter values for our Item-based Machine Learning baselines.

Algorithm	Hyperparameter	Yelp2018	Amazon-Book
EASE^R	l2 norm	-	7.60E+01
SLIM	topK	1000	1000
	l1 ratio	1.15E-03	2.08E-04
	alpha	0.0010	0.0010
SLIM BPR	topK	880	733
	epochs	200	420
	symmetric	True	True
	sgd mode	adagrad	adagrad
	lambda i	1.93E-04	1.00E-05
	lambda j	1.00E-05	1.00E-02
	learning rate	4.50E-02	4.99E-03
NegHOSLIM	-	-	-
NegHOSLIM (EN)	feature pairs n	6	1
	topK	777	1000
	l1 ratio	3.38E-05	1.81E-04
	alpha	0.0273	0.0010

Table 57. Selected hyperparameter values for our Matrix Factorization baselines.

Algorithm	Hyperparameter	Yelp2018	Amazon-Book
MF-BPR	sgd mode	adagrad	adagrad
	epochs	540	1500
	num factors	200	200
	batch size	1024	1024
	positive reg	2.78E-04	2.55E-04
	negative reg	1.00E-02	1.00E-02
	learning rate	4.31E-02	1.00E-01
MF-WARP	sgd mode	adagrad	adagrad
	epochs	845	940
	num factors	196	200
	batch size	1024	1024
	neg item attempts	20	5
	positive reg	1.00E-02	3.57E-03
	negative reg	1.50E-03	1.00E-02
	learning rate	3.65E-02	6.61E-02
SVDpp	sgd mode	adam	adam
	epochs	495	445
	use bias	True	False
	batch size	128	8
	num factors	193	200
	item reg	1.19E-04	1.91E-03
	user reg	1.87E-04	1.00E-02
	learning rate	1.29E-03	1.56E-03
	negative quota	0.0324	0.0742
PureSVD	num factors	148	350
NMF	num factors	34	323
	solver beta loss	multiplicative update:kullback-leibler	multiplicative update:frobenius
	init type	random	nndsvda
iALS	num factors	200	200
	epochs	30	110
	confidence scaling	linear	linear
	alpha	20.3787	50.0000
	epsilon	10.0000	0.0125
	reg	1.00E-02	1.00E-05

Table 58. Selected hyperparameter values for our Autoencoder baselines.

Algorithm	Hyperparameter	Yelp2018	Amazon-Book
MultVAE	epochs	400	500
	learning rate	9.22E-05	3.68E-05
	sgd mode	rmsprop	rmsprop
	l2 reg	1.00E-06	4.89E-05
	dropout	0.6449	0.2031
	anneal steps	357709	118728
	anneal cap	0.1076	0.3358
	batch size	1024	256
	encoding size	503	512
	layer size multiplier	6	10
	max n hidden layers	1	4

Table 59. Selected hyperparameter values for the purely collaborative GNN models.

Algorithm	Hyperparameter	Yelp2018	Amazon-Book
GDE	epochs	845	5
	batch size	4096	4096
	learning rate	1.00E-06	1.02E-03
	embedding size	344	168
	sgd mode	adam	adam
	beta	0.1000	3.7201
	feature type	both	both
	drop out	0.9000	0.2956
	reg	1.00E-01	1.63E-04
	smooth ratio	0.0135	0.0040
	rough ratio	0.0021	0.0001
	loss type	bpr	adaptive
GTN	epochs	85	700
	batch size	512	256
	learning rate	6.92E-03	1.21E-04
	embedding size	93	88
	sgd mode	adam	adam
	GNN layers K	5	4
	embedding smoothness weight	5	7
	l2 reg	2.74E-04	6.73E-05
	dropout rate LightGCN	8.00E-01	1.15E-01
	dropout rate GTN	3.26E-01	2.34E-01
RGCF	-	-	-
HCCF	epochs	65	400
	batch size	256	512
	learning rate	8.71E-05	8.93E-06
	embedding size	172	278
	sgd mode	adam	adam
	GNN layers K	2	2
	HYP layers C	3	1
	hyperedge size	51	350
	dropout	0.1547	0.4043
	contrastive loss weight	0.0080	0.0000
	l2 reg	1.47E-06	1.64E-05
	contrastive loss temperature τ	0.4450	0.1365
	leaky relu slope	0.0100	0.0100
INMO	epochs	355	150
	batch size	256	256
	learning rate	2.99E-04	4.28E-04
	embedding size	304	350
	sgd mode	adam	adam
	K	6	4
	l2 reg	1.00E-06	1.89E-05
	template loss weight	0.0709	0.1000
	template node ranking metric	page rank	degree
	dropout	0.3442	0.4744
	template ratio	0.9074	0.9264
	normalization decay	9.90E-01	9.90E-01

Table 60. Selected hyperparameter values for the purely collaborative GNN models.

Algorithm	Hyperparameter	Yelp2018	Amazon-Book
SimGCL	epochs	100	20
	batch size	512	2048
	learning rate	1.56E-04	7.68E-04
	embedding size	322	344
	sgd mode	adam	adam
	GNN layers K	6	1
	noise magnitude ϵ	0.0476	0.0379
	contrastive loss temperature τ	0.0824	0.1873
	contrastive loss weight	0.1000	0.0755
	l2 reg	7.41E-06	1.12E-05
LightGCN	epochs	710	530
	batch size	256	256
	learning rate	1.25E-04	2.74E-04
	embedding size	350	298
	sgd mode	adam	adam
	GNN layers K	5	6
	l2 reg	2.65E-05	9.17E-06
	dropout rate	2.28E-01	1.00E-01

Table 61. Selected hyperparameter values for the GNN models that include a knowledge base.

Algorithm	Hyperparameter	Yelp2018	Amazon-Book
HAKG	-	-	-
KGCL	epochs	25	630
	batch size	1024	512
	learning rate	1.81E-03	1.43E-04
	sgd mode	adam	adam
	GNN layers K	4	5
	contrastive loss temperature τ	0.0342	0.0123
	GNN dropout rate	4.91E-01	3.50E-01
	knowledge graph dropout rate	4.79E-01	3.39E-01
	user interaction dropout rate	8.38E-01	3.84E-01
	embedding size	211	323
	mix ratio	0.8547	0.7095
	uicontrast	weighted-mix	weighted-mix
	entities per head	20	7
	l2 reg	7.91E-05	1.02E-04
	self supervised loss weight	0.0122	0.0002

Table 62. Experimental results for all analyzed methods of SIGIR 2022 for the Amazon Book dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0007	0.0003
TopPop	0.0370	0.0168
GlobalEffects	0.0003	0.0001
UserKNN CF	0.2301	0.1371
ItemKNN CF	0.2436	**0.1496**
$P^3\alpha$	0.2432	**0.1459**
$RP^3\beta$	0.2474	**0.1488**
GF-CF	-	-
$EASE^R$	0.2479	**0.1543**
SLIM BPR	0.2337	0.1399
SLIM	**0.2511**	**0.1563**
NegHOSLIM	-	-
NegHOSLIM (EN)	0.2472	**0.1536**
MF-BPR	0.1633	0.0911
MF-WARP	0.1672	0.0945
SVDpp	0.2163	0.1211
PureSVD	0.1418	0.0862
NMF	0.1262	0.0749
iALS	0.2378	0.1356
MultVAE	0.2485	**0.1473**
GDE	0.0004	0.0002
GTN	0.1852	0.0996
HAKG	-	-
RGCF	-	-
HCCF	0.1328	0.0651
INMO	**0.2511**	0.1456
KGCL	0.2425	0.1403
SimGCL	0.2441	0.1412
LightGCN	0.2442	0.1407

Table 63. Experimental results for all analyzed methods of SIGIR 2022 for the Yelp 2018 dataset.

	Cutoff 20	
	Recall	NDCG
Random	0.0004	0.0003
TopPop	0.0213	0.0132
GlobalEffects	0.0004	0.0002
UserKNN CF	0.0988	0.0668
ItemKNN CF	0.1057	0.0718
$P^3\alpha$	0.1033	0.0682
$RP^3\beta$	0.1043	0.0693
GF-CF	-	-
$EASE^R$	-	-
SLIM BPR	0.0989	0.0659
SLIM	0.1007	0.0700
NegHOSLIM	-	-
NegHOSLIM (EN)	0.0994	0.0667
MF-BPR	0.0556	0.0358
MF-WARP	0.0600	0.0380
SVDpp	0.0879	0.0572
PureSVD	0.0757	0.0512
NMF	0.0744	0.0479
iALS	0.1120	0.0750
MultVAE	0.1188	0.0796
GDE	0.0834	0.0535
GTN	0.1000	0.0643
HAKG	-	-
RGCF	-	-
HCCF	0.0610	0.0399
INMO	**0.1219**	**0.0809**
KGCL	0.1125	0.0745
SimGCL	**0.1222**	**0.0822**
LightGCN	0.1172	0.0781

A BASELINE HYPERPARAMETER RANGES

In this section we report the hyperparameter ranges and distribution for all the baselines in our experiments, see Table 64 (Nearest-Neighbor Collaborative and Content-Based), 65 (Graph-based), 66 (Item-based Machine Learning), 67 (Matrix Factorization), 68 (Factorization Machines Collaborative and Hybrid), and 69 (Autoencoder).

Algorithm	Hyperparameter	Range	Type	Distribution
UserKNN, ItemKNN UserKNN CBF ItemKNN CBF	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	
UserKNN CFCBF ItemKNN CFCBF	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	
	ICM or UCM weight	10^{-2} - 10^{+2}	Real	log-uniform

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

Table 64. Hyperparameter ranges and distributions for our Nearest-Neighbor Collaborative and Content-Based baselines.

Algorithm	Hyperparameter	Range	Type	Distribution
$P^3\alpha$	topK	5 - 1000	Integer	uniform
	alpha	0 - 2	Real	uniform
	normalize similarity[a]	True, False	Categorical	
$RP^3\beta$	topK	5 - 1000	Integer	uniform
	alpha	0 - 2	Real	uniform
	beta	0 - 2	Real	uniform
	normalize similarity[a]	True, False	Categorical	
GF-CF	topK	5 - 5000	Integer	uniform
	alpha	10^{-3} - 10^{+3}	Real	log-uniform
	num factors	1 - 350	Integer	uniform

[a]The *normalize similarity* hyperparameter refers to applying L1 regularization on the rows of the similarity matrix.

Table 65. Hyperparameter ranges and distributions for our Graph-based baselines.

Algorithm	Hyperparameter	Range	Type	Distribution
$EASE^R$	l2 norm	10^0 - 10^{+7}	Real	log-uniform
SLIM	topK	5 - 1000	Integer	uniform
	l1 ratio	10^{-5} - 10^0	Real	log-uniform
	alpha	10^{-3} - 10^0	Real	uniform
SLIM BPR	topK	5 - 1000	Integer	uniform
	epochs	1 - 1500	Integer	early-stopping
	symmetric	True, False	Categorical	
	sgd mode	sgd, adam, adagrad	Categorical	
	lambda i	10^{-5} - 10^{-2}	Real	log-uniform
	lambda j	10^{-5} - 10^{-2}	Real	log-uniform
	learning rate	10^{-4} - 10^{-1}	Real	log-uniform
NegHOSLIM	epochs	1 - 300[a]	Integer	early-stopping
	feature pairs n	1 - 1000	Integer	uniform
	lambdaBB	1 - 10^7	Real	log-uniform
	lambdaCC	1 - 10^7	Real	log-uniform
	rho	1 - 10^7	Real	log-uniform
NegHOSLIM (EN)	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

[a]The number of epochs is lower due to the algorithm being slower, but converging in a lower number of epochs.

Table 66. Hyperparameter ranges and distributions for our Item-based Machine Learning baselines.

Algorithm	Hyperparameter	Range	Type	Distribution
MF-BPR	num factors	1 - 200[d]	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
MF-WARP	num factors	1 - 200[d]	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
	neg item attempts	5, 10, 15, 20	Categorical	
SVDpp	num factors	1 - 200[a]	Integer	uniform
	epochs	1 - 500[b]	Integer	early-stopping
	use bias	True, False	Categorical	
	sgd mode	sgd, adam, adagrad	Categorical	
	batch size	2^0 - 2^{10}	Integer	log-uniform
	item reg	10^{-5} - 10^{-2}	Real	log-uniform
	user reg	10^{-5} - 10^{-2}	Real	log-uniform
	learning rate	10^{-4} - 10^{-1}	Real	log-uniform
	negative quota[c]	0.00 - 0.50	Real	uniform
PureSVD	num factors	1 - 350	Integer	uniform
NMF	num factors	1 - 350	Integer	uniform
	init type	nndsvda, random	Categorical	
	solver beta loss	mult. update:frobenius, coord. descent:frobenius, coord. descent:kullback-leibler	Categorical	
iALS	num factors	1 - 200[d]	Integer	uniform
	epochs	1 - 500[e]	Integer	early-stopping
	confidence scaling	linear, log	Categorical	
	alpha	10^{-3} - $5 \cdot 10^{+1}$ [f]	Real	log-uniform
	epsilon	10^{-3} - 10^{+1} [f]	Real	log-uniform
	reg	10^{-5} - 10^{-2}	Real	log-uniform

[a]The number of factors is lower than PureSVD or NFM due to the algorithm being slower.
[b]The number of epochs is lower than SLIM BPR or MF BPR due to the algorithm being slower.
[c]The *negative quota* is the percentage of samples chosen among items unobserved by the user, having a target rating of 0.
[d]The number of factors is lower due to the algorithm being slower.
[e]The number of epochs is lower due to the algorithm being slower, but converging in a lower number of epochs.
[f]The maximum value of this hyperparameter had been suggested in the article proposing the algorithm.

Table 67. Hyperparameter ranges and distributions for our Matrix Factorization baselines.

Algorithm	Hyperparameter	Range	Type	Distribution
LightFM CF	epochs	1 - 300	Integer	early-stopping
	n components	1 - 200	Integer	uniform
	loss	Categorical	bpr, warp, warp-kos	uniform
	sgd mode	Categorical	adagrad, adadelta	uniform
	learning rate	Real	$10^{-6} - 10^{-1}$	log-uniform
	item alpha	Real	$10^{-5} - 10^{-2}$	log-uniform
	user alpha	Real	$10^{-5} - 10^{-2}$	log-uniform
LightFM ItemHybrid	epochs	1 - 300	Integer	early-stopping
	n components	1 - 200	Integer	uniform
	loss	Categorical	bpr, warp, warp-kos	uniform
	sgd mode	Categorical	adagrad, adadelta	uniform
	learning rate	Real	$10^{-6} - 10^{-1}$	log-uniform
	item alpha	Real	$10^{-5} - 10^{-2}$	log-uniform
	user alpha	Real	$10^{-5} - 10^{-2}$	log-uniform

Table 68. Hyperparameter ranges and distributions for our Factorization Machines Collaborative and Hybrid baselines.

Algorithm	Hyperparameter	Range	Type	Distribution
MultVAE	epochs	1 - 500[a]	Integer	early-stopping
	learning rate	10^{-6} - 10^{-2}	Real	log-uniform
	l2 reg	10^{-6} - 10^{-2}	Real	log-uniform
	dropout	0 - 0.8	Real	uniform
	annealing steps	10^{5} - $6 \cdot 10^{5}$	Integer	uniform
	anneal cap	0 - 0.6	Real	uniform
	batch size	128, 256, 512, 1024	Categorical	
	encoding size	1 - 512	Integer	uniform
	layer size multiplier[b]	2 - 10	Integer	uniform
	max n hidden layers	2 - 4	Integer	uniform

[a]The number of epochs is lower due to the algorithm being slower, but converging in a lower number of epochs.
[b]This hyperparameter is used to generate the decoder architecture. Starting from the encoding size the size of the next hidden layer is computer as the product of the previous one and the layer multiplier. The process terminates when either the desired number of hidden layers is reached or any further hidden layer added would exceed the size of the input data.

Table 69. Hyperparameter ranges and distributions for our Autoencoder baselines.

REFERENCES

[1] Robert M Bell and Yehuda Koren. 2007. Improved neighborhood-based collaborative filtering. In *KDD Cup and Workshop at the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07)*. 7–14.

[2] Andrzej Cichocki and Anh Huy Phan. 2009. Fast Local Algorithms for Large Scale Nonnegative Matrix and Tensor Factorizations. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* 92-A, 3 (2009), 708–721. https://doi.org/10.1587/TRANSFUN.E92.A.708

[3] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010*, Xavier Amatriain, Marc Torrens, Paul Resnick, and Markus Zanker (Eds.). ACM, 39–46. https://doi.org/10.1145/1864708.1864721

[4] 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

[5] 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

[6] Maurizio Ferrari Dacrema, Simone Boglio, Paolo Cremonesi, and Dietmar Jannach. 2021. A Troubling Analysis of Reproducibility and Progress in Recommender Systems Research. *ACM Trans. Inf. Syst.* 39, 2 (2021), 20:1–20:49. https://doi.org/10.1145/3434185

[7] 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

[8] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*. IEEE Computer Society, 263–272. https://doi.org/10.1109/ICDM.2008.22

[9] Lukas Lerche and Dietmar Jannach. 2014. Using graded implicit feedback for bayesian personalized ranking. In *Eighth ACM Conference on Recommender Systems, RecSys '14, Foster City, Silicon Valley, CA, USA - October 06 - 10, 2014*, Alfred Kobsa, Michelle X. Zhou, Martin Ester, and Yehuda Koren (Eds.). ACM, 353–356. https://doi.org/10.1145/2645710.2645759

[10] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis (Eds.). ACM, 689–698. https://doi.org/10.1145/3178876.3186150

[11] Xia Ning and George Karypis. 2011. SLIM: Sparse Linear Methods for Top-N Recommender Systems. In *11th IEEE International Conference on Data Mining, ICDM 2011, Vancouver, BC, Canada, December 11-14, 2011*, Diane J. Cook, Jian Pei, Wei Wang, Osmar R. Zaïane, and Xindong Wu (Eds.). IEEE Computer Society, 497–506. https://doi.org/10.1109/ICDM.2011.134

[12] Bibek Paudel, Fabian Christoffel, Chris Newell, and Abraham Bernstein. 2017. Updatable, Accurate, Diverse, and Scalable Recommendations for Interactive Applications. *ACM Trans. Interact. Intell. Syst.* 7, 1 (2017), 1:1–1:34. https://doi.org/10.1145/2955101

[13] 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

[14] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, Jeff A. Bilmes and Andrew Y. Ng (Eds.). AUAI Press, 452–461. https://www.auai.org/uai2009/papers/UAI2009_0139_48141db02b9f0b02bc7158819ebfa2c7.pdf

[15] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *CSCW '94, Proceedings of the Conference on Computer Supported Cooperative Work, Chapel Hill, NC, USA, October 22-26, 1994*, John B. Smith, F. Donelson Smith, and Thomas W. Malone (Eds.). ACM, 175–186. https://doi.org/10.1145/192844.192905

[16] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001*, Vincent Y. Shen, Nobuo Saito, Michael R. Lyu, and Mary Ellen Zurko (Eds.). ACM, 285–295. https://doi.org/10.1145/371920.372071

[17] Yifei Shen, Yongji Wu, Yao Zhang, Caihua Shan, Jun Zhang, Khaled B. Letaief, and Dongsheng Li. 2021. How Powerful is Graph Convolution for Recommendation?. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong (Eds.). ACM, 1619–1629. https://doi.org/10.1145/3459637.3482264

[18] Harald Steck. 2019. Embarrassingly Shallow Autoencoders for Sparse Data. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia (Eds.). ACM, 3251–3257. https://doi.org/10.1145/3308558.3313710

[19] Harald Steck and Dawen Liang. 2021. Negative Interactions for Improved Collaborative Filtering: Don't go Deeper, go Higher. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021 - 1 October 2021*, Humberto Jesús Corona Pampín, Martha A. Larson, Martijn C. Willemsen, Joseph A. Konstan, Julian J. McAuley, Jean Garcia-Gathright, Bouke Huurnink, and Even Oldridge (Eds.). ACM, 34–43. https://doi.org/10.1145/3460231.3474273

[20] 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

[21] 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

[22] 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

[23] 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

[24] 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

[25] 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

[26] 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