Reproducibility and Artifact Consistency of the SIGIR 2022 Recommender Systems Papers Based on Message Passing

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

Graph-based techniques relying on neural networks and embeddings have gained attention as a way to develop Recommender Systems (RS) with several papers on the topic presented at SIGIR 2022 and 2023. Given the importance of ensuring that published research is methodologically sound and reproducible, in this paper we analyze 10 graph-based RS papers, most of which were published at SIGIR 2022, and assess their impact on subsequent work published in SIGIR 2023. Our analysis reveals several critical points that require attention: (i) the prevalence of bad practices, such as erroneous data splits or information leakage between training and testing data, which call into question the validity of the results; (ii) frequent inconsistencies between the provided artifacts (source code and data) and their descriptions in the paper, causing uncertainty about what is actually being evaluated; and (iii) the preference for new or complex baselines that are weaker compared to simpler ones, creating the impression of continuous improvement even when, particularly for the Amazon-Book dataset, the state-of-the-art has significantly worsened. Due to these issues, we are unable to confirm the claims made in most of the papers that we examined and attempted to reproduce.

CCS Concepts: • **Information systems** → **Recommender systems**; *Collaborative filtering*; • **General and reference** → Evaluation.

Additional Key Words and Phrases: Recommender Systems, Graph Neural Networks, Evaluation, Reproducibility

ACM Reference Format:
Maurizio Ferrari Dacrema, Michael Benigni, and Nicola Ferro. 2018. Reproducibility and Artifact Consistency of the SIGIR 2022 Recommender Systems Papers Based on Message Passing. In . ACM, New York, NY, USA, 40 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Reproducibility is a central issue in several areas of science that face the so-called *reproducibility crisis* [50]. As reported by Baker [6], approximately 70% of researchers in physics and engineering are unable to reproduce someone else's experiments, and roughly 50% fail to reproduce even their own experiments. Computational and data-intensive sciences [24, 47] are no exception, and this is also true for *Information Retrieval* (IR) and *Recommender Systems* (RS) [11, 21, 23, 38], especially considering how both fields are heavily reliant on machine learning approaches today [43] and how reproducibility is a concern for machine learning itself [26, 53].

Defining what *reproducibility* means for IR and RS is still an open issue. However, despite the terminological debate, it is generally understood as the ability to obtain results very similar to those reported in the paper, whether in terms of the absolute values of the effectiveness scores or the actual labels (or recommendation lists) produced by the model [11, 12, 46]. One of the first issues that arises is ensuring that the original source code and data artifacts provided by the authors are, in fact, *consistent* with what was described in the paper. There can be several sources of inconsistencies, and

Manuscript submitted to ACM

indeed we report several examples of this, from erroneous data splits (an issue already identified by Ferrari Dacrema et al. [20]) to changes in the model implementation after the publication of the paper. These types of inconsistencies are problematic because they hide *what* is actually being *reproduced*, whether it is what is described in the paper or something else, and may easily propagate to subsequent publications if other researchers use those artifacts themselves. Once the consistency of the artifacts has been established, the experimental evaluation can proceed to obtain the results and determine whether they reproduce what is reported in the original paper according to the desired criteria. This approach focuses on the reproduced method but does not consider the context in which the result was presented. Indeed, in machine learning research the results are generally interpreted in a *relative* manner; that is, the absolute value of an effectiveness metric does not provide much information on its own but becomes meaningful when compared with the same effectiveness score measured for other methods. Typically, this is used to support claims that the proposed method is able to outperform the state-of-the-art in a given scenario. This opens a new challenge that goes beyond reproducing the absolute results, that is, confirming the *conclusions* that the proposed method is indeed competitive with the state-of-the-art. This requires reevaluating several aspects of the experimental protocol, from the training procedure used for the proposed method to the way in which the state-of-the-art baselines were selected and optimized. Several previous studies over the years have shown that a large number of papers present methods that are, in fact, not competitive against simple baselines due to various bad practices, such as poor baseline optimization, information leakage, and anomalous data splits [3, 5, 10, 20–22, 33, 44, 58, 62, 77]. Indeed, back in 2009 Armstrong et al. [5] raised the issue by observing how new methods in IR were not competitive against some older and simpler baselines. More than 10 years later, this was found to still be the case by Kharazmi et al. [33] and Lv et al. [44]. Similar issues have been reported in the RS domain by Ferrari Dacrema et al. [20, 21], who observed that it was not possible to reproduce several deep learning techniques in the RS domain and that most of them were not competitive with a set of simple fine-tuned baselines when applied to the traditional top-n collaborative filtering task.

In the last few years, graph-based techniques for neural networks and embeddings [1, 7, 60] have become a very active and impactful area of research with many ramifications including RS [74], which is the focus of this paper. To the best of our knowledge, He et al. [28] published the first paper on these topics at a SIGIR conference, which subsequently attracted attention and inspired follow-up work in later years, particularly at SIGIR 2022 [18, 19, 28, 42, 52, 69, 75, 76, 78, 80] and SIGIR 2023 [14, 27, 36, 41, 55, 56, 70, 73, 79, 81, 82].

Given the general interest in graph-based techniques for RS, the surge of papers on this topic at SIGIR, and the previous experiences and concerns of the IR and RS communities regarding reproducibility and weak baselines, the question of whether the reproducibility of the results and reliability of the experimental methodology has improved naturally arises and papers addressing these questions are starting to appear, e.g., Anelli et al. [4]. Therefore, in this paper, we investigate the following research questions about a set of graph-based RS papers published at SIGIR 2022:

- **Can we reproduce the results reported in the original papers using the same data splits and source code?** Our results indicate that slightly less than half of the reported results can be reproduced, with large variations across papers ranging from 0% to 66%, consistent with observations from prior studies in related fields.
- **Are the experimental methodologies used in the original papers, as well as the publicly available artifacts, correct and consistent?** While 90% of the papers provided publicly available artifacts, we identified several major inconsistencies related to anomalous data splitting. The consistency of model implementations is generally good, with rare exceptions. However, the early-stopping process is often inconsistent with the descriptions in the papers, and in some cases even relies on test data to select the number of training epochs.

- **Can we confirm that the proposed methods are competitive against simple yet robust baselines, beyond those reported in the original papers?** We observe that most methods fail to outperform simple baselines, particularly on the Amazon-Book dataset, where message-passing models fall substantially behind. In some cases, the analyzed method remains highly competitive, being outperformed by only a small margin by a single baseline. However, we also find several instances where a simple ItemKNN largely outperforms them.

Moreover, we asked ourselves about the impact that (ir)reproducible papers might have on follow-up research. To address this, we surveyed papers published the following year at SIGIR 2023 [14, 27, 36, 41, 55, 56, 70, 73, 79, 81, 82] to qualitatively understand whether and how the analyzed SIGIR 2022 papers influenced their evaluations and findings. This analysis reveals that a comparison is almost impossible, due to how different the results are even for papers that adopt similar evaluation protocols, which is surprising and worrying.

The paper is organized as follows. In Section 2, we describe our research method and how we attempted to reproduce the selected papers. The results of our analysis on methodology, consistency and the re-execution of the experiments including additional baselines are presented in Section 3. We finally summarize our findings along several dimensions and discuss the implications of our research in Section 4.

2 RESEARCH METHOD

2.1 Terminology

While over the years there has been a lot of discussion, which is still ongoing, on how reproducibility and replicability should be defined [17, 54], we follow the definitions by the updated *ACM Policy on Artifact and Review Badging*,[1] which are aligned to those of the International Vocabulary for Metrology (VIM) [32] and the NISO definitions for reproducibility badging [49]. These are also the definitions adopted by the *ACM SIGIR Artifact Badging* committee:[2]

- **Reproducibility (Different team, same experimental setup)**: The main results of the paper have been obtained in a subsequent study by a person or team other than the authors, using, in part, artifacts provided by the author.
- **Replicability (Different team, different experimental setup)**: The main results of the paper have been independently obtained in a subsequent study by a person or team other than the authors, without the use of author-supplied artifacts.

This study has two main components. First, a *consistency* analysis of the original source code and data artifacts provided by the original paper, as well as the methodology adopted for the evaluation. Note that previous studies on reproducibility typically do not conduct an in-depth consistency analysis of the artifacts. Second, an assessment of the *reproducibility* (same artifacts/experimental setup) of the results, using the original source code, the same datasets, the experimental setup, and the same optimal hyperparameters for the proposed methods. However, there are instances where we also performed some *replicability* (different artifacts/experimental setup) of the results. This occurred when it was necessary to fix the original source code, replace an erroneous data split after identifying potential issues such as anomalous distributions or information leakage among training/validation/test sets, apply our own early-stopping methodology when the original one was not properly specified or implemented, or include our own baselines as a common reference for comparison across all the analyzed papers. However, in the remainder of the paper, we will use only the term reproducibility for the sake of readability, as it is the main focus of this study. Note that in defining the

[1] https://www.acm.org/publications/policies/artifact-review-and-badging-current
[2] https://sigir.org/general-information/acm-sigir-artifact-badging/

criteria for determining whether a paper was successfully reproduced or not, as further detailed in Section 2.3, we also considered the guidelines of the *ACM SIGIR Artifact Badging* committee for awarding this kind of badge.

We did not perform only a reproducibility analysis; since we also verified the source code, its availability, documentation, as well as its consistency with the paper, we conducted, in the terms of the *ACM SIGIR Artifact Badging* committee:

- **Artifacts Evaluated – Functional**: The artifacts associated with the research are found to be documented, consistent, complete, exercisable, and include appropriate evidence of verification and validation.
- **Artifacts Evaluated – Reusable and Available**: The artifacts associated with the paper are of a quality that significantly exceeds minimal functionality. That is, they have all the qualities of the Artifacts Evaluated – Functional level, but, in addition, they are very carefully documented and well-structured to the extent that reuse and repurposing are facilitated.

Note that, as explained earlier, different papers used different datasets or the same datasets but with different preprocessing or splitting. As a consequence, most of the methods examined can not be directly compared to determine which is more effective and under which conditions, even when the experiment uses the same dataset. In order to address this issue, we also conduct an experiment similar to what done in [4], where we independently optimize all the hyperparameters of the methods we attempt to reproduce on a limited set of datasets. Although such an analysis on all the originally reported datasets or on a more ample set of datasets and conditions would be appropriate and interesting, it focuses on the *generalizability* of the results and therefore falls outside the scope of this study and is left for future work. Moreover, such an analysis would be extremely computationally expensive. However, since we fine-tuned our baselines for each of the datasets used, they still provide a reference to qualitatively assess the relative merits of the methods examined.

2.2 Selection of Candidate Papers

In this study, we focus on RS based on message passing. Since the number of papers published on the topic is very large, we chose to select those published at one of the most prominent conferences in the field, SIGIR.

Reproducibility Study on the SIGIR 2022 Papers. The selection of papers was carried out through the following process. First, a list of candidate papers was retrieved by scanning the *Collaborative Filtering* and *Recommender Systems* sessions in the proceedings of SIGIR 2022 [2]. A paper was considered a candidate for reproduction if it (i) proposed a graph-based recommender technique with message passing and (ii) focused on the top-n recommendation problem. We did not select papers that develop e.g., session-based or reinforcement learning methods, nor papers that propose more general methods not strongly connected to graph-based approaches, e.g., [25] or that use message passing but as a secondary component, e.g., [83]. Since all the selected papers are strongly based on LightGCN [28], a method that has had a substantial impact on the community, we included it in our analysis, even though it was published earlier at SIGIR 2020. After this screening process, we ended up with a collection of ten candidate papers [18, 19, 28, 42, 52, 69, 75, 76, 78, 80].

We then checked for the availability of artifacts, i.e., source code and data, provided by the original authors. If the source code was not publicly available, we contacted the authors via email. We were able to retrieve the artifacts for all the ten candidate papers.[3]

[3]Note that the GitHub repository of Liu et al. [42] remained empty until January 2023, six months after SIGIR 2022.

Qualitative Impact on SIGIR 2023 Papers. The selection of papers was carried out through the following process. First, a list of candidate papers was retrieved by scanning the *Collaborative Filtering*, *Collaborative Filtering and Graph Neural Approaches*, and *Knowledge Graphs for Recommendation* sessions in the proceedings of SIGIR 2023 [13]. A paper was considered a candidate for the qualitative analysis if it focused on the same task selected for the SIGIR 2022 papers and used one of the SIGIR 2022 papers as a baseline in the experiments, along with LightGCN. After this screening process, we ended up with a collection of eleven papers [14, 27, 36, 41, 55, 56, 70, 73, 79, 81, 82] for the qualitative analysis.

2.3 Consistency and Reproducibility Analysis

The first stage of the analysis is to assess whether the original source code and data artifacts are *consistent* with what is described in the paper. To do so, we performed a manual inspection of the source code as well as some simple analyses on the data splits (e.g., we compared the number of occurrences the items have in the training and test data). When assessing consistency, we must remain mindful that the provided source code is often a simplified version of the original experimental pipeline, which may only include the implementation or correct configuration for one of the experiments reported in the paper.[4] Therefore, we consider the original artifacts to be *consistent* when:

- There are no apparent anomalies in the statistical properties of the data splits.
- The core algorithm and the details of the training process are consistent with the description in the paper.

For the second stage of the analysis, a candidate paper must meet several conditions to be considered successfully reproduced:

- The provided source code can be executed successfully and is correct. Note that we fix minor errors when necessary and explicitly report when we do.[5]
- At least one of the datasets used in the original evaluation is available, either because the original training-test split is provided or because the dataset is publicly available and the paper contains sufficient detail to perform the data preprocessing and splitting.
- It is possible to *closely* reproduce the numerical results reported in the original paper by using the provided artifacts.

Regarding the first requirement, an important aspect to consider is whether reproducing a paper also requires us to reproduce possible methodological mistakes that were made in the original implementation, e.g., information leakage. In those instances, we argue that running experiments which contain errors has little scientific value, therefore we correct errors when present and describe them in our analysis. This issue typically arises when selecting the optimal number of epochs to train a machine learning model. Often published papers report the optimal number of epochs but do not explain how it was obtained or, in some cases, the provided implementation performs early-stopping on the test data.

In order to ensure that our analysis is done with a consistent setup, we integrate all the original implementations into our own evaluation framework [20]. This typically involves leaving the original model unchanged while using our implementation for early-stopping and evaluation. For efficiency and consistency across implementations, we replace

[4]Providing all the source code should, in principle, enhance reproducibility. However, full experimental pipelines can be quite long, particularly when they rely on complex structured libraries. In such cases, if they are not well-organized and supported by adequate documentation, it may be challenging for a researcher to discern the complete experimental protocol. This could make it very difficult to identify inconsistencies between the paper and the artifacts, as well as to recognize important details that were omitted. Consequently, while the results may be reproducible, the transparency of the process is not necessarily improved.

[5]For example, we correct the source code whenever early-stopping is performed on test data.

the original data sampling implementation with one developed in Cython,[6] ensuring that the original sampling strategy is not altered.

Finally, the notion of what *closely reproduced* means is still an open issue. Breuer et al. [11] proposed a set of measures, among which relative distance among the effectiveness scores, which we adopt also here. However, neither Breuer et al. [11] nor subsequent work by Maistro et al. [46] provided guidance on what ranges of effectiveness delta should be considered indicative of successful reproduction. Therefore, we follow a simple rule-of-thumb, where a relative difference of 2% or less in at least one metric on a dataset is used to determine that the results have been successfully reproduced on that dataset. This definition aims to categorize as non-reproducible only those methods that produce results very different from those reported in the original paper, while acknowledging that results can sometimes be affected by factors that are difficult to control (e.g., stochastic behavior of the method, different hardware configurations, etc.). If the results have been successfully reproduced on some but not all of the datasets used in the paper, we consider that paper to be only partially reproduced.

Overall, we were able to conduct this analysis for nine out of the ten candidate papers, as the artifacts provided by Liu et al. [42] did not meet our requirements. The artifacts include various scripts, but they lack instructions on how to execute them, the sequence to follow, and the appropriate environment settings. Additionally, much of the relevant source code appears to be commented out. We also observed inconsistencies in the hard-coded paths, with some referencing the Last-FM dataset while others the Amazon-Book dataset, which is not mentioned in the paper. The state of the source code, the absence of execution instructions, and the missing preprocessed data files prevented us from assessing the consistency of the processing procedure and running the experiments. We contacted the authors for assistance but did not receive a response.

2.4 Baselines

In order to provide a comprehensive view of the possible strong baselines, we selected a representative set of methods from different families, ranging from the nearly 30-year-old user-based *k-Nearest Neighbors* (KNN) to more recent neural and graph-based methods. These methods were identified as highly effective in previous comparative studies [4, 20] as well as based our own experience. In the main paper we focus on the following ones:

- **TopPop**: non-personalized method recommending to all users the most popular items the user has not yet interacted with.
- **UserKNN**: user-based nearest-neighbor algorithm [59], with cosine similarity and shrinkage [9].
- **ItemKNN**: item-based nearest-neighbor algorithm [61], with cosine similarity and shrinkage [9].
- **GF-CF**: a graph-based method that is based on a low-pass filter and has a closed form solution [64].[7]
- **SLIM**: item-based model that uses linear regression to compute the item similarity [48].[8]
- **MF-BPR**: matrix factorization method based on the *Bayesian Personalized Ranking* (BPR) loss [57].
- **iALS**: matrix factorization method for ranking tasks based on alternating least-squares [31].
- **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 [51].

[6] Cython is an extension of Python that allows to include static types and compile the code for substantially improved performance https://cython.org/
[7] Note that occasionally the results for **GF-CF** may be missing due to its memory requirements exceeding the 64GB available on our server.
[8] In particular we optimize the ElasticNet loss.

- **NegHOSLIM (EN)**: linear full-rank model similar to SLIM, which includes higher-order interactions as input-features [66].[9]
- **MultVAE**: variational autoencoder that assumes a multinomial likelihood for user-item interactions [37].

The full results, available in the additional material, include additional baselines for a total of 21 collaborative models (Random, Global Effects, $P^3\alpha$, $EASE^R$, SLIM-BPR, NegHOSLIM, SVDpp, PureSVD, NMF, LightFM) and 6 content-based or hybrid models (ItemKNN, UserKNN and LightFM both content-based and hybrid).

Hyperparameter Optimization. In order to ensure that the baseline algorithms are properly optimized we select their hyperparameters with a Bayesian search [29], implemented by Scikit-Optimize.[10] The search explores a total of 50 cases, with the first 16 used as initial random points. Once the optimal hyperparameters are determined, including the number of epochs, the final model is fitted on the union of training and validation data using these optimal hyperparameters. The considered hyperparameter ranges and distributions are the same as those in Ferrari Dacrema et al. [20] and are listed in the additional material.

Early-stopping. In order to select the optimal number of epochs for both the candidate algorithm as well as for baselines based on iterative optimization, we rely on the widely used early-stopping. The model is trained on the training data, and its effectiveness is evaluated on the validation data every 5 epochs. If the model's effectiveness does not improve for 5 consecutive evaluations, the training is stopped, and the epoch number associated with the best-performing model is selected. Note that we apply this early-stopping method to our baselines and also in some additional experiments with the candidate algorithms to validate the reliability of how the optimal number of epochs was determined.

3 DETAILED ANALYSIS

For each paper, we first summarize its contributions, then analyze it along four dimensions: (i) the datasets originally used for evaluation; (ii) methodological issues and the consistency between the artifacts and the paper; (iii) the reproducibility of the results based on the provided artifacts; and (iv) the competitiveness of the method against baselines.

The reported results are the product of extensive experiments. We report the results of approximately 800 trained models, which required the fitting of approximately 25.000 models during the hyperparameter optimization phase. Overall the experiments required a total computation time of 4 years. Due to the extensive number of experiments and datasets, we report only one table for each paper selecting the dataset in which we obtained the worse outcome in terms of the reproducibility of the results. Notice that this does not necessarily mean *worse results*, but rather the results that are furthest from those published.[11] To complement the main paper, we also provide an online appendix with extensive additional information: the full results for each dataset with up to 26 baseline algorithms, the description and statistics of the datasets as well as a table listing the hyperparameter values of the analyzed methods that we used in our experiments, specifying for each where they were described (i.e., paper or source code).[12] Each table groups baselines into two categories: those with closed-form solutions and those that require iterative training.

[9] Due to the large memory requirement we trained it by using an ElasticNet loss (EN) instead of the originally proposed one, in a similar way as SLIM.
[10] https://scikit-optimize.github.io/
[11] There are two exceptions to this: (i) MovieLens 100k for GDE, because the dataset is very small, and since there are no other datasets in which we could successfully run the experiment but not reproduce the results, we select Gowalla, which is the dataset with the highest number of interactions and successfully reproduced results; (ii) Last-FM for HAKG because we had to apply a significant alteration to the dataset, and therefore we select the dataset with the second worst results in terms of reproducibility.
[12] https://github.com/remaplab/TOIS25_Reproducibility-SIGIR22-GCN

3.1 LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation

He et al. [28] propose LightGCN, a graph-based collaborative filtering method where user and item embeddings are propagated according to the graph adjacency matrix, via message passing. Six of the other methods we attempted to reproduce in this study (all except SimGCL and HAKG) used LightGCN as a baseline.

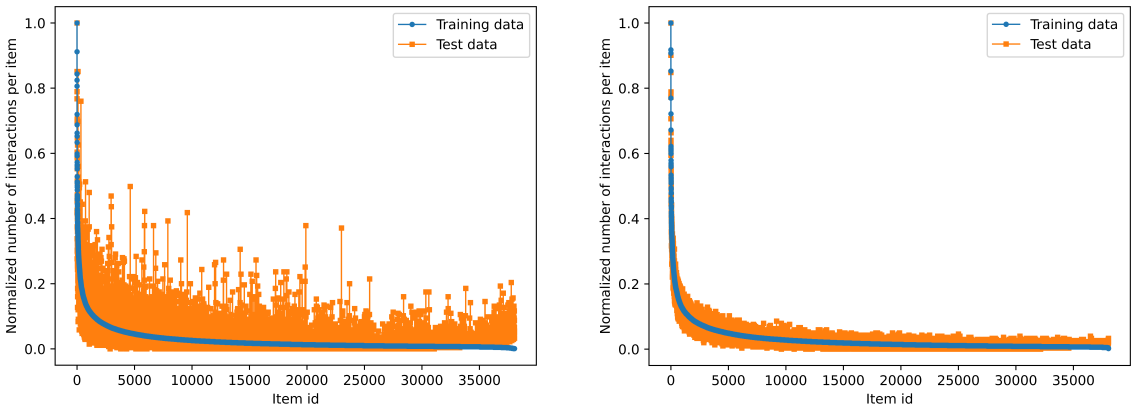
Datasets. LightGCN is evaluated on three datasets: Amazon-Book, Gowalla and Yelp2018. All datasets are preprocessed with a 10-core selection. The evaluation procedure is similar to that adopted by Wang et al. [72], the data splitting is performed with user-wise random holdout as 72% training, 8% validation and 20% test for all the datasets.

Consistency and Methodology. The GitHub repository[13] contains both the implementation and the training-test data split. The provided material is *not fully consistent* with what is described in the paper.

The first issue we observed is that the training and test sets do not exhibit the distribution expected from a user-wise random holdout split. Figure 1 shows the popularity distribution of the items in the training and test sets for both the original data split and a new data split generated by us following the user-wise random holdout procedure described in the paper. The popularity is normalized by dividing it by that of the most popular item in the corresponding set. Items are then sorted by decreasing popularity in the training data. In a user-wise random holdout split, the popularity distributions of the training and test sets should, on average, be similar however, this is not the case for the original split. Note that, as pointed out by Anelli et al. [4] the arXiv version of Wang et al. [72] which published the original data split, includes an updated version of the results indicating that the issue has been fixed.[14] Nonetheless, the original erroneous split has been used by LightGCN and several other papers. Similar issues with anomalous data splits have also been observed in other papers by previous reproducibility studies [20]. To assess the extent of the anomaly, we consider three statistical measures. First, we compute the Gini Index of the popularity distributions to quantify the strength of their popularity bias. We observe that the Gini Index of the original split is quite similar to that of our split, indicating that the two splits do not exhibit different popularity biases. For example, considering Yelp2018 the Gini Index of the original split is 0.53 for the training set and 0.56 for the test set; while the split generated by us has 0.51 for the training set and 0.54 for the test set. As a second step we compute the Kendall's τ and Pearson correlation coefficients between the number of interactions of each item in the training and test sets. The Kendall's τ correlation, given two lists, measures the percentage of couples of elements that are in the same order in both lists. In this context, we aim to confirm that if an item a is more popular than item b in the training set, it also has more interactions than b in the test set. However, a drawback of this metric is that if the full dataset contains many items with a similar number of interactions, the stochastic nature of the holdout split could introduce significant noise, resulting in low Kendall's τ values. The Pearson correlation, on the other hand, measures the linear correlation in the number of interactions, making it robust to the type of noise that Kendall's τ is sensitive to, but it is less intuitive to interpret. In a random holdout split we expect the training and test sets to exhibit a Pearson correlation close to 1 and a high Kendall's τ. The results indicate strong discrepancies. In the original Amazon-Book training-test sets the relative ordering of items differs substantially (Kendall's τ 0.17, Pearson Correlation 0.50) whereas in our split the relative ordering is more consistent, and the item popularities are highly correlated (Kendall's τ 0.52, Pearson Correlation 0.95). On Yelp2018 the findings are similar but less pronounced, with the original split (Kendall's τ 0.37, Pearson Correlation 0.78) showing worse correlations when compared to ours (Kendall's τ 0.59, Pearson Correlation 0.96). On Gowalla the gap is smallest, with the original split (Kendall's τ 0.25, Pearson Correlation 0.85) displaying a distribution very similar to ours (Kendall's τ 0.34, Pearson

[13] https://github.com/gusye1234/LightGCN-PyTorch
[14] The new version with corrected data split and results is dated July 2020, see https://arxiv.org/abs/1905.08108

Correlation 0.96). These unusual data splits are not justified. For the reproducibility part of our study, we retain the original training-test splits. However, when evaluating competitiveness against baselines, we also conduct experiments on our newly generated training-test splits. We chose to retain the original erroneous splits in our experiments for two reasons: (i) to enable a direct comparison with the results in the LightGCN paper, and (ii) because two other papers in our study (SimGCL and GTN) used the same datasets and these inconsistent splits.



(a) Normalized popularity distributions of the original training and test data splits.

(b) Normalized popularity distributions of the training and test data splits randomly generated by us.

Fig. 1. Normalized popularity distributions of the training and test data splits for Yelp2018 used in the LightGCN paper, the value 1 corresponds to the most popular item in that split. Figure 1b shows the expected popularity distribution for a random holdout data split, with the normalized values on both training and validation being on average similar. Figure 1a shows instead the distribution in the original data splits, as can be seen the training and test distributions are different.

A second issue is that the LightGCN paper refers to a previous paper for the experimental methodology [72], which specifies performing early-stopping if the Recall@20 does not improve for 50 successive epochs. However, the provided implementation does not perform early-stopping, rather it only evaluates and prints the results on the test data every 5 epochs. Neither the paper nor the repository material provide details or insights on how many training epochs were used for the evaluation. To address this, we applied both the original methodology and our own early-stopping approach when training the model, to validate the results.

Reproducibility. In our experiments we could *partially reproduce* the results reported in the original paper. In particular, we could closely reproduce the results for both Gowalla and Amazon-Book, while on Yelp2018 (see Table 1) our results were approximately 5% lower than those reported. Regarding early-stopping, both the original and our approaches produced very similar results.

Baselines. LightGCN demonstrates inferior effectiveness compared to our set of baselines across all datasets, both in terms of the results reported in the original paper and those obtained by our experiments. For example, in our experiments, MultVAE, RP$^3\beta$ and GF-CF (the latter two being simple graph-based baselines) outperform all versions of LightGCN on all datasets (see the results for Yelp2018 in Table 1). Particularly striking are the results for Amazon-Book, where LightGCN lags significantly behind the baselines, achieving an NDCG of 0.0315 while the simple ItemKNN reaches almost twice that value, 0.0624. The results are consistent between the original training-test split and the one

generated by us. However, for the Amazon-Book dataset, the split generated by us yields much higher absolute metric values. For example, the ItemKNN baseline achieves an NDCG@20 of 0.0624 in the original split and 0.1680 in the split generated by us. This aligns with our earlier observation that, in the original split, the data distribution of the test set differs from that of the training set. Consequently, it is not surprising that all models perform worse in absolute terms on the original split.

Table 1. Results for LightGCN on the Yelp2018 dataset. The baselines are highlighted in **bold** if they outperform our results for LightGCN. Results for LightGCN are highlighted in **bold** only if they outperform all baselines. Values are underlined if they are better than the results for LightGCN that were reported in the original paper.

	Cutoff 20	
	Recall	NDCG
TopPop	0.0124	0.0101
UserKNN CF	**0.0637**	**0.0533**
ItemKNN CF	**0.0622**	**0.0514**
$RP^3\beta$	**0.0672**	**0.0558**
GF-CF	**0.0693**	**0.0568**
SLIM	**0.0646**	**0.0541**
NegHOSLIM (EN)	0.0590	0.0492
MF-BPR	0.0382	0.0313
iALS	**0.0667**	**0.0546**
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

3.2 Less is More: Reweighting Important Spectral Graph Features for Recommendation

Peng et al. [52] propose *Graph Denoising Encoder* (GDE). The paper analyses graph convolution in the spectral domain and observe that only a limited number of spectral graph features significantly contribute to model effectiveness, specifically the highest and lowest frequencies (i.e., eigenvalues), while intermediate frequencies are less important. This effect is attributed to the different semantics of these frequencies, with higher frequencies representing differences between users and lower frequencies representing commonalities. Based on this observation they introduce GDE, which acts as a band-pass filter by selecting high and low frequencies while removing intermediate ones.

Datasets. GDE is evaluated on five datasets: MovieLens 100k, MovieLens 1M, CiteULike-a, Pinterest, and Gowalla. The data splitting is a random holdout of interactions sampled globally, 20% for training and 80% for testing. Note that, as opposed to what commonly done, the training data is much smaller than the testing data. The validation set is created by splitting 5% of the training set. Moreover, the paper only states that all interactions are made implicit with a value of 1, but does not report the preprocessing applied to the Gowalla and Pinterest datasets, which are much smaller than their original versions.

Methodological Issues. The GitHub repository[15] contains both the implementation and the training-test data split. The provided material is *fully consistent* with what is described in the paper.

During our experiments we observed that GDE is numerically unstable, frequently failing to train properly, particularly on certain datasets. We believe this instability is related to an issue in how the lowest eigenvalues are computed. Consider

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

a matrix containing the user-item interactions defined as $R \in \mathbb{R}^{u \times i}$, where u is the number of users and i is the number of items. GDE requires computing both the highest and the lowest eigenvalues, along with their corresponding eigenvectors, for the item-item similarity matrix $S_{I-I} = R^T R$ and the user-user similarity matrix $S_{U-U} = RR^T$. While computing the highest eigenvalues can be done rather efficiently, computing the *lowest* ones for such large matrices is a much more computationally intensive task and is subject to numerical rounding errors. The original implementation uses the LOBPCG method [34][16] which is very fast but assumes the input matrix to be *positive definite*, i.e., all its eigenvalues are strictly positive. Crucially, this requirement is never verified and, as we will show, it is not met. Any matrix A that can be represented as $A = B^T B$ is at least *positive semi-definite*. However, to be strictly *positive definite* an additional condition must be met, matrix B must have linearly independent columns [68, Ch. 7, p. 396]. The number of linearly independent columns and rows of a matrix is its *rank*, which can never be larger than the smallest of its dimensions, i.e., $rank(R) \leq min(u, i)$. This immediately implies that at least one of the two similarity matrices will *not* be positive definite, simply because R is rectangular. In a typical scenario $u > i$ hence S_{U-U} will not be positive definite. To validate our statements, we compute the rank of the matrix R containing the training set by applying the SVD method, i.e., the rank of the matrix is equal to the number of non-zero singular values.[17] The results are that on MovieLens 100k and CiteULike-a there are fewer than 10 linearly dependent rows or columns, while for MovieLens 1M and Pinterest there are approximately 500 and for Gowalla almost 900.[18]

This issue has two important consequences: (i) since the data violates the assumptions of the method used to compute the eigenvectors its results are prone to be erroneous, potentially invalidating the findings reported in the original paper; (ii) a large number of the smallest eigenvalues that GDE tries to use, possibly *all of them*, will be zero.[19] Therefore, they will not represent the high frequency signals that GDE aims to leverage. Indeed, by examining the hyperparameter values provided in the original GitHub repository, we observe that using the lowest eigenvalues is beneficial only for the MovieLens 100k and 1M datasets, while it is not advantageous for the other datasets. We believe the combination of these two effects explains the unstable behavior observed for GDE. Computing the lowest eigenvalues using the full SVD decomposition is impractical due to its potentially large memory requirements and likely large numerical errors. While it may be possible to address this issue by rethinking how this stage of the computation is carried out, possibly leveraging other algebraic properties, this would require a redesign of the method that goes beyond the scope of this paper.

We also identified other methodological issues. In the original data splits, a small number of interactions (11 for CiteULike-a, which is 0.02% of the training set, and 7884 for Pinterest, which is 3.94% of the training set) appear in both the training and test sets. In our experiments we removed these interactions from the training set to avoid any overlap and prevent the possibility of information leakage. Furthermore, although the source code reports the optimal number of training epochs for each dataset, the paper does not explain how these numbers were determined. The original implementation does not apply early-stopping, but rather trains for a large number of epochs and periodically evaluates the model on the test data. To address this, we conducted two experiments: one using the reported number of epochs and the other applying our own early-stopping methodology. We also observe that, due to the data splitting methodology,

[16]The implementation is based on the `lobpcg` function from `PyTorch`, see the reference documentation here https://pytorch.org/docs/stable/generated/torch.lobpcg.html

[17]We applied the function `matrix_rank` from `PyTorch`, see the reference documentation here https://pytorch.org/docs/stable/generated/torch.linalg.matrix_rank.html

[18]The detailed results are the following: MovieLens 100k (users=943, items=1682, rank=940), MovieLens 1M (users=6040, items=3952, rank=3401), CiteULike-a (users=5551, items=16980, rank=5544), Pinterest (users=37501, items=9836, rank=9303) and Gowalla (users=29858, items=40981, rank=28965).

[19]Their value will not be exactly zero due to the limits of machine precision. Manual inspection reveals that, on MovieLens 100k, several of the smallest eigenvalues have an absolute value of approximately 10^{-9}.

only 1% of the data is used for validation, which could result in noisy hyperparameter tuning and early-stopping. Lastly, the paper does not mention that the source code employs a definition of Recall where the denominator is not the number of relevant items for the user but rather the minimum between this value and the length of the recommendation list. While this definition has been used in previous studies [37], and is intended to normalize the metric in cases where the number of relevant items exceeds the length of the recommendation list, the existence of two competing definitions of a metric poses obstacles to reproducibility when the paper does not explicitly state which definition is being used. For our experiments, we use both the normalized definition of Recall used in the GDE paper to enable a direct comparison, as well as the more common non-normalized definition of Recall.

Reproducibility. In our experiments we could *partially reproduce* the results reported in the original paper. In particular, we could closely reproduce the results for both MovieLens 1M and Gowalla (see Table 2). On MovieLens 100k the results were lower and slightly beyond the 2% threshold, with a normalized Recall of 0.5196 compared to the reported 0.5400. On the remaining CiteULike-a and Pinterest datasets GDE exhibited numerical instabilities that caused the training to fail after a few epochs.[20] In our experiments with our early-stopping, we could only reproduce the results for MovieLens 1M.

Baselines. Our early-stopping runs of GDE show competitive results only on the two MovieLens datasets compared to our baselines. On the other hand, our runs of GDE with the number of epochs from the paper are competitive on all datasets except CiteULike-a and Pinterest, where convergence is not achieved and the normalized Recall is around 0.002. Given the unstable nature of GDE, and its fast training time, we decided to perform a new hyperparameter optimization. With our new set of hyperparameters, GDE was able to reach convergence on Pinterest and outperform our baselines as well, with a normalized Recall of 0.1082 compared to the 0.1067 reached by the best performing baseline iALS. Finally, the results reported in the paper for GDE are competitive with our baselines across all datasets. The results obtained with a non-normalized implementation of Recall are consistent with those obtained with the normalized definition used in the original source code.

Table 2. Results for GDE on the Gowalla dataset. The baselines are highlighted in **bold** if they outperform our results for GDE. Results for GDE are highlighted in **bold** only if they outperform all baselines. Values are underlined if they are better than the results for GDE that were reported in the original paper.

	Cutoff 20		
	Recall (normalized)	Recall	NDCG
TopPop	0.0421	0.0298	0.0451
UserKNN CF	0.1128	0.0748	0.1304
ItemKNN CF	0.1119	0.0741	0.1288
$RP^3\beta$	0.1116	0.0737	0.1285
GF-CF	-	-	-
SLIM	0.1057	0.0692	0.1219
NegHOSLIM (EN)	0.1053	0.0690	0.1214
MF-BPR	0.0299	0.0202	0.0319
iALS	0.1361	0.0963	0.1531
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

[20]This issue was also raised by other practitioners on the GDE GitHub repository.

3.3 Are Graph Augmentations Necessary? Simple Graph Contrastive Learning for Recommendation

Yu et al. [80] propose *Simple Graph Contrastive Learning* (SimGCL). The paper argues that an important step in a typical contrastive learning pipeline is to perform graph augmentation by applying perturbations to the adjacency matrix used by message passing, i.e., the user-item interaction matrix. The contrastive loss then pushes the original and augmented graphs toward similar latent representations. The paper claims that the main contribution to the model effectiveness in constrastive learning based models does not come from the graph augmentation (e.g., random edge dropout) but rather comes from the constrastive learning loss function InfoNCE. The InfoNCE loss effect is to increase the separation between positive and negative samples for each user. SimGCL generates contrastive views by applying random perturbations of the embeddings instead of graph augmentations.

Datasets. SimGCL is evaluated on three datasets: Yelp2018 and Amazon-Book, with the same training-test split of LightGCN [28], and Douban Book. Both Yelp2018 and Amazon-Book are preprocessed with a 10-core selection, while for Douban Book only interactions with rating of at least 4 are retained. For all datasets the data splitting is a user-wise random holdout, 72% training, 8% validation and 20% test.

Methodological Issues. The GitHub repository[21] includes both the implementation and the training-test data split. However, the provided materials are only *partially consistent* with what is described in the paper.

The first issue is that the paper relies on the same data splits used in the LightGCN paper for the Yelp2018 and Amazon-Book datasets which, as shown in Section 3.1, are not the result of a correct user-wise random holdout split. We decided to keep these splits in our experiments for the same reasons as for LightGCN, but we also conducted experiments on the new user-wise random holdout data splits we generated following the procedure described in the paper. Secondly, there is a discrepancy between the optimal values of the hyperparameters used by the authors in their experiments and the optimal values obtained from the sensitivity analysis of the same hyperparameters, as reported in the paper. For example, the hyperparameter sensitivity analysis refers to a model with 2 graph convolution layers ($K = 2$), whereas it was previously shown that the best model requires 3 convolution layers. Additionally, the paper states that the optimal noise level ϵ is 0.1 yet the sensitivity analysis plots indicate values 0.05 for Yelp2018, 0.1 for Amazon-Book, and 0.2 for Douban Book. These inconsistencies, albeit small, create uncertainty about the correct configuration that should be used to reproduce the results reported in the paper. Finally, the number of training epochs is determined based on a convergence plot that shows the Recall and BPR loss, but the paper does not specify whether this plot is computed on the training, validation, or test data. Due to this, we conducted two experiments, one training the model using the reported number of epochs and the other applying our own early-stopping methodology.

Reproducibility. In our experiments we could *partially reproduce* the results reported in the original paper. In particular, we could reproduce the results on Amazon-Book and Yelp2018. However, on Douban Books our effectiveness was about 10% lower than what was reported in the paper (see Table 3). SimGCL achieved almost identical results regardless of whether the model was trained using the reported number of epochs or with our early-stopping methodology, across all datasets and data splits.

Baselines. On Yelp2018, SimGCL performs competitively against almost all our simple baselines with an NDCG@20 of 0.0594, with only MultVAE achieving the better result of 0.0602. On Douban Book and Amazon-Book, however, SimGCL performs significantly worse than our baselines, both in our runs and using the higher results reported in the

[21]We use the PyTorch implementation provided by the authors https://github.com/Coder-Yu/SELFRec

original paper. For instance, on Douban Book, the NDCG@20 for SLIM is 0.2226 while SimGCL reaches 0.1583 according to the original paper and 0.1445 in our run. Since SimGCL uses the same training-test splits as LightGCN, we can directly compare the two methods by looking in particular at NDCG@20. On Yelp2018, we observe a 15% improvement of SimGCL (0.0594) over LightGCN (0.0506). For Amazon-Book, the improvement is 20% on our data split (0.1047 for SimGCL and 0.0862 for LightGCN) and a surprising 80% on the original LightGCN data split (0.0402 for SimGCL and 0.0315 for LightGCN. See Section 3.1 for an analysis of its anomalous distribution).

Table 3. Results for SimGCL on the Douban Book dataset. The baselines are highlighted in **bold** if they outperform our results for SimGCL. Results for SimGCL are highlighted in **bold** only if they outperform all baselines. Values are underlined if they are better than the results for SimGCL that were reported in the original paper.

	Cutoff 20	
	Recall	NDCG
TopPop	0.0722	0.0582
UserKNN CF	**0.1686**	**0.1575**
ItemKNN CF	**0.1972**	**0.1908**
$RP^3\beta$	**0.2033**	**0.1841**
GF-CF	**0.1788**	**0.1604**
SLIM	**0.2250**	**0.2226**
NegHOSLIM (EN)	**0.1971**	**0.1833**
MF-BPR	0.0916	0.0774
iALS	**0.1833**	**0.1668**
MultVAE	**0.1885**	**0.1694**
SimGCL paper	**0.1772**	**0.1583**
SimGCL our early-stopping	0.1685	0.1492
SimGCL provided number of epochs	0.1629	0.1445

3.4 Learning to Denoise Unreliable Interactions for Graph Collaborative Filtering

Tian et al. [69] presents *Robust Graph Collaborative Filtering* (RGCF). RGCF comprises two main steps. First, a graph denoising module removes interactions, from the adjacency matrix used in the graph convolution, that are estimated by the model itself as noisy, while assigning a reliability weight to the remaining interactions. Second, a diversity-preserving module builds new interaction graphs (i.e., adjacency matrix) based on the denoised one by adding new edges derived from the trained model's predictions. The model is trained using BPR with an additional contrastive loss, i.e., InfoNCE, that pulls the representations of nodes learned from the augmented graphs closer to each other.

Datasets. RGCF is evaluated on three datasets: Yelp2018, Amazon-Book and MovieLens 1M. Both Yelp2018 and Amazon-Book are preprocessed with a 15-core selection, while for MovieLens 1M only interactions whose rating is at least 4 are retained and associated to an implicit rating of 1. The data splitting is a random holdout of interactions sampled globally, 80% training, 10% validation and 10% test.

Consistency and Methodology. The GitHub repository[22] contains only the implementation but does not contain the training-test data split. The provided implementation is *fully consistent* with what is described in the paper.

Since the provided materials do not include the training-test data split, we applied the described preprocessing on all three datasets but could not reproduce exactly the data statistics reported in the paper. Due to this, we ran RGCF with the optimal hyperparameters and also conducted a new hyperparameter search. Unfortunately, we were able to use the

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

model only on MovieLens 1M. On the other datasets, the gradient update step in PyTorch caused a memory spike that exceeded the 24GB available on our RTX 3090 GPU, preventing us from running the model, while the computational time on a CPU was prohibitive.[23] The implementation correctly uses early-stopping on the validation set to determine the optimal number of training epochs.

Reproducibility. In our experiments we could *partially reproduce* the results reported in the original paper. Considering that our experiments could not use the original training-test split, the results we obtain include an additional level of variance due to the stochastic nature of the data splitting process. On MovieLens 1M, the only dataset we could use, our experiments yielded better results than those reported in the paper (see Table 4) and can be considered fully reproduced, as the HR metric is within 2% of the values reported. Furthermore, we observed that applying our early-stopping approach increased the model's effectiveness, and performing a new hyperparameter optimization resulted in additional improvements.

Baselines. According to the results on MovieLens 1M (see Table 4), the only dataset usable for this reproducibility study, RGCF exhibits worse effectiveness than SLIM and GF-CF, with a gap of up to 10%. Our version, using the newly optimized hyperparameters, shows improved effectiveness but remains below several simple baselines.

Table 4. Results for RGCF on the MovieLens 1M dataset. The baselines are highlighted in **bold** if they outperform our results for RGCF. Results for RGCF are highlighted in **bold** only if they outperform all baselines. Values are underlined if they are better than the results for RGCF that were reported in the original paper.

	Cutoff 10			
	Recall	NDCG	HR	MRR
TopPop	0.0773	0.1213	0.4894	0.2433
UserKNN CF	0.1939	0.2711	0.7733	0.4741
ItemKNN CF	0.1811	0.2578	0.7441	0.4610
$RP^3\beta$	0.1824	0.2557	0.7560	0.4577
GF-CF	**0.2076**	**0.2885**	**0.7897**	**0.4944**
SLIM	**0.2057**	**0.2944**	**0.7870**	**0.5034**
NegHOSLIM (EN)	**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
MultVAE	**0.2029**	**0.2812**	**0.7858**	**0.4845**
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

3.5 INMO: A Model-Agnostic and Scalable Module for Inductive Collaborative Filtering

Wu et al. [75] presents *Inductive Embedding Module for collaborative filtering* (INMO), which aims to improve the effectiveness of matrix factorization models for new users. The paper focuses on matrix factorization models that are *transductive* (i.e., not model based, such as SVD++, MF-BPR etc..) and proposes an *inductive* (model-based) representation of users and items as a function of the embeddings of a selected subset of template users and items. The paper also explores strategies to select these templates.

[23] The issue appears to be related to how PyTorch handles gradient updates rather than the original RGCF implementation. The RGCF paper does not provide details about the hardware configuration. We found that on our i9-9900K CPU with 16 cores and 64GB of RAM, it is only feasible to run the experiment for Yelp2018, which, however, requires 1.5 hours per epoch, resulting in an estimated total runtime of 31 days.

Datasets. INMO is evaluated on three datasets: Yelp2018, Amazon-Book and Gowalla. Both Yelp2018 and Amazon-Book are preprocessed by selecting only the interactions with a rating of at least 4, associating them to an implicit rating of 1, followed by a 10-core selection. No information is provided on the preprocessing of Gowalla. The data splitting is a user-wise random holdout, 70% training, 10% validation and 20% test.

Methodological Issues. The GitHub repository[24] contains the implementation, the training-test data split, and, uniquely among the algorithms analyzed in this study, it also includes the implementation of the baseline models along with their own hyperparameter optimization code. The provided material is *fully consistent* with what is described in the paper.[25] As a minor note, the paper reports a number of interactions for all datasets that is approximately 15% higher than that of the provided data. The number of epochs is correctly selected using early-stopping, where training stops if the NDCG@20 on the validation data does not improve for 50 successive epochs.

Reproducibility. In our experiments we could *partially reproduce* the results reported in the original paper. In particular, we could reproduce the results on Yelp2018 within less than 1% as well as for one of the metrics on Amazon-Book, while for Gowalla (see Table 5) we obtained results that are approximately 2.5-5% lower.

Baselines. On the Yelp2018 dataset, INMO has an NDCG of 0.0647 that consistently outperforms most simple baselines with the sole exception of MultVAE, which reaches 0.0670. However, on the Gowalla dataset (see Table 5), INMO is competitive against some of the baselines but not with RP$^3\beta$, GF-CF, SLIM, NegHOSLIM and MultVAE, which demonstrate comparable effectiveness. On the Amazon-Book dataset, INMO is considerably less effective, with an NDCG of 0.0934, 32% lower than that of the best-performing baselines RP$^3\beta$ (0.1402) or SLIM (0.1451).

Table 5. Results for INMO on the Gowalla dataset. The baselines are highlighted in **bold** if they outperform our results for INMO. Results for INMO are highlighted in **bold** only if they outperform all baselines. Values are underlined if they are better than the results for INMO that were reported in the original paper.

	Cutoff 20		
	Recall	Precision	NDCG
TopPop	0.0303	0.0083	0.0208
UserKNN CF	0.1834	0.0493	0.1376
ItemKNN CF	0.1908	0.0508	0.1431
RP$^3\beta$	**0.2029**	**0.0548**	**0.1523**
GF-CF	**0.2014**	**0.0525**	**0.1483**
SLIM	**0.2037**	**0.0574**	**0.1573**
NegHOSLIM (EN)	0.1934	**0.0526**	**0.1478**
MF-BPR	0.1308	0.0350	0.0979
iALS	0.1820	0.0491	0.1362
MultVAE	**0.2079**	**0.0555**	**0.1563**
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

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

[25] We note that the paper uses the BPR loss and defines it by using the log sigmoid function: $-ln(\sigma(x))$, while the implementation instead uses the softplus$(-x) = ln(1+e^{-x})$ function. This occurs in a few of the papers we analyze and we wish to point out that the two formulations are mathematically identical: $-ln(\sigma(x)) = -ln\,(1/(1+e^{-x})) = ln(1+e^{-x}) = \text{softplus}(-x)$.

3.6 Hypergraph Contrastive Collaborative Filtering

Xia et al. [76] proposed *Hypergraph Contrastive Collaborative Filtering* (HCCF), which extends LightGCN's message-passing approach on the user-item adjacency matrix. In addition, HCCF incorporates a layer of message-passing on a learnable hypergraph adjacency matrix, which is decomposed into the product of two lower-dimensional matrices. The model also includes a step called Hierarchical Hypergraph Mapping, which applies message-passing on the learned hypergraph adjacency matrix. The model is trained using contrastive learning.

Datasets. HCCF is evaluated on three datasets: Yelp2018, Amazon-Book and MovieLens 10M. Both Yelp2018 and MovieLens 10M are preprocessed with a 10-core selection, while Amazon-Book with a 20-core one. The data splitting is a random holdout, 70% training, 10% validation and 20% test, but the paper does not specify if the sampling is done globally or user-wise.

Consistency and Methodology. The GitHub repository[26] provides two versions of the implementation (one in TensorFlow 1, which is now obsolete, and one in PyTorch) and the data split for training, validation, and testing. The provided material is *partially consistent* with what is described in the paper.

Both the PyTorch and TensorFlow versions include a setting,[27] which is not described in the paper, that limits the number of training samples per epoch to 10^4. Given that the datasets contain between $1.5 \cdot 10^6$ and 10^7 interactions, this setting has a significant impact on the training process, as the model requires a much higher number of epochs to converge, thereby affecting the early-stopping process. There is also an inconsistency between the optimal hyperparameters used in the source code and the search space reported in the paper. For example, the contrastive loss weight λ_1 has optimal values of 10^{-6} for MovieLens 10M and 10^{-7} for Amazon-Book, despite the smallest value in the reported search space being 10^{-5}. Finally, the criteria used to select the optimal number of epochs is not described.

Lastly, the PyTorch implementation of HCCF differs from the TensorFlow version, as well as from the version described in the paper, in several ways: (i) when computing the contrastive loss, the PyTorch implementation uses the embeddings obtained through the graph convolution but does not update them; (ii) only a single layer is used instead of the multi-layer hypergraph convolution; and (iii) the learned hypergraph adjacency matrix is excluded from the hypergraph convolution. According to the description provided in the GitHub repository, these modifications were introduced to improve the model's effectiveness on sparse data, which partially contradicts the original claims of the paper. In our experiments, we updated the PyTorch implementation to ensure its consistency with the HCCF model described in the paper. Note that we do not report the results for the simplified model because it is effectively a different algorithm.

Reproducibility. In our experiments we could *partially reproduce* the results reported in the original paper. In particular, we could reproduce the results on Yelp2018 and achieved a *substantial improvement* of about 70% on MovieLens 10M (see Table 6). However, HCCF failed to converge on Amazon-Book, exhibiting effectiveness comparable to a random recommender. This issue also occurred with the TensorFlow version of HCCF.[28] The substantial improvement observed on MovieLens 10M may be explained by the fact that the original training limited the number of samples drawn per epoch, whereas in our experiments, we used all available samples. Since MovieLens 10M is the dataset with the highest number of interactions, this difference is likely more pronounced.

[26] https://github.com/akaxlh/HCCF
[27] This setting is referred to as `trnNum` in the original source code.
[28] We observed that adjusting the regularization hyperparameters allowed the model to exhibit limited effectiveness, however performing a new hyperparameter optimization for non reproducible models goes beyond the scope of this paper.

Baselines. In all datasets, HCCF performs largely below the baselines, both when considering the results reported in the paper and those from our experiments. On MovieLens 10M (see Table 6), the best-performing baseline achieves results that are 16% better than the best results we obtained for HCCF. On Yelp2018, the best-performing baseline MultVAE reaches an NDCG@40 of 0.1264, while HCCF of 0.0592, outperforming it by more than twice. On Amazon-Book, where our run of the model failed to converge, the best-performing baseline achieves results that are 546% better than the original result reported in the paper, with an NDCG@40 of 0.1803 compared to 0.0330 for HCCF.

Table 6. Results for HCCF on the MovieLens 10M dataset. The baselines are highlighted in **bold** if they outperform our results for HCCF. Results for HCCF are highlighted in **bold** only if they outperform all baselines. Values are underlined if they are better than the results for HCCF that were reported in the original paper.

	Cutoff 20		Cutoff 40	
	Recall	NDCG	Recall	NDCG
TopPop	0.1363	0.1903	0.2114	0.2022
UserKNN CF	**0.3503**	**0.4448**	**0.4700**	**0.4595**
ItemKNN CF	0.2816	0.3645	0.3884	0.3790
RP$^3\beta$	0.2886	**0.3761**	0.3960	0.3895
GF-CF	**0.3342**	**0.4210**	**0.4484**	**0.4354**
SLIM	**0.3387**	**0.4422**	**0.4578**	**0.4563**
NegHOSLIM (EN)	**0.3430**	**0.4430**	**0.4630**	**0.4582**
MF-BPR	0.2849	0.3569	0.3989	0.3759
iALS	**0.3368**	**0.4232**	**0.4593**	**0.4426**
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

3.7 HAKG: Hierarchy-Aware Knowledge Gated Network for Recommendation

Du et al. [18] proposes *Hierarchy-Aware Knowledge Gated Network* (HAKG), which integrates both collaborative interactions and knowledge-based graphs. The paper aims to leverage the hierarchical structure of knowledge graphs and the "higher order" relations in collaborative data through the use of hyperbolic embeddings.

Datasets. HAKG is evaluated on three datasets: Alibaba-iFashion, Yelp2018 and Last-FM. All datasets are preprocessed with a 10-core selection. For Last-FM the split is the same used by KGAT [71] (user-wise random holdout, 72% training, 8% validation and 20% test). For the other two datasets the split is 80% training, 10% validation and 10% test, but the paper does not specify if the sampling is done globally or user-wise.

Consistency and Methodology. The GitHub repository[29] contains both the implementation and the training-test data split. The provided material is *not consistent* with what is described in the paper.

First, in the Last-FM data splits, a substantial number of interactions (169782 which is 13.17% of the training data) appear in both the training and test data. This erroneous split originates from a previous paper [71]. In our experiments, we removed these interactions from the training set to avoid any overlap and prevent the possibility of information leakage. Furthermore, the training-test split of the Yelp2018 dataset exhibits an item popularity distribution that is not consistent with a user-wise random holdout split, see Figure 2. Using the same procedure described in Section 3.1, we computed three statistics. The Gini Index of the item popularity in the entire dataset is 0.58, for the original training

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

data is 0.59, while for the test data is 0.63, indicating that the distribution is more unbalanced. Comparing the item popularity between the training and test data, we observe that the Kendall's τ is 0.38 and the Pearson Correlation is 0.83. In a user-wise random holdout data split both values should be much higher.[30] Unfortunately, we are not able to run HAKG on Yelp2018 due to its memory requirements exceeding the 24GB available on our RTX 3090 GPU and its prohibitive computational time on CPU.[31] As a minor note, while the paper states the splitting of the data is 80% training, 10% validation and 10% test, the provided data split is actually 72% training, 8% validation and 20% test. Finally, the provided source code performs early-stopping using the test data, which introduces information leakage. In our experiments, we corrected this issue and performed early-stopping exclusively using the validation data.

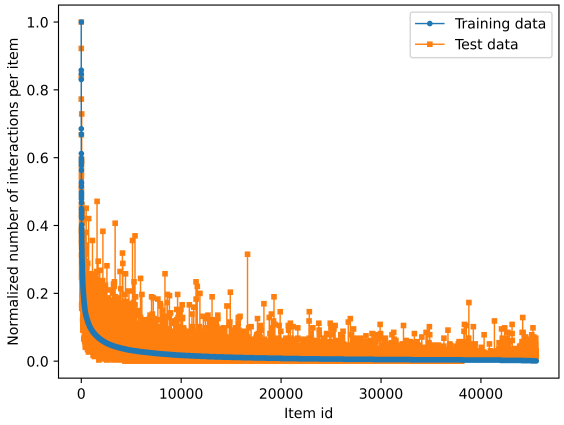


Fig. 2. Normalized popularity distributions of the original training and test data splits for Yelp2018 used by the HAKG paper.

Reproducibility. In our experiments we could *not reproduce* the results reported in the original paper on any of the datasets. In particular, we obtained results that were 7% lower than those reported in the paper on the Alibaba-iFashion dataset (see Table 7), possibly due to the early-stopping on the test data used in the original source code. On the other hand, we achieved nearly twice as high results on the Last-FM dataset, 0.1644 compared to the reported 0.0931. This unusual result on Last-FM may be attributed to the inconsistent data split used in the paper, which we corrected for our experiments. Although it may seem counterintuitive that removing information leakage would lead to an increase in effectiveness for all models, this effect arises because, for this task, items already interacted with by a user are not recommended to them again. This prevented all models from recommending several items present in the user's test data that had already been observed by the user. We were not able to run HAKG on Yelp2018 due to memory limitations and therefore we were unable to evaluate their reproducibility on this dataset.

Baselines. On both the Alibaba-iFashion and Last-FM datasets, HAKG performs worse than many simple baselines. This is despite the significant improvements we observe on Last-FM, where we achieved an effectiveness that is nearly double the results reported in the paper, our run of HAKG has an NDCG@20 of 0.1644 compared to 0.2014 for RP$^3\beta$ and 0.2078 for SLIM. On the Alibaba-iFashion dataset (see Table 7), while our run of HAKG falls below several baselines, the

[30] In a new user-wise random holdout split generated by us, we obtain a Kendall's τ of 0.59 and a Pearson Correlation of 0.96.

[31] The HAKG paper does not provide details on the hardware configuration. We attempted to run the experiment on a i9-9900K CPU with 16 cores and 64GB of RAM. However, on Yelp2018 the training time for each epoch is approximately 7 hours resulting in an estimated runtime of between 30 days (for 100 epochs) and 300 days (for 1000 epochs). For this reason it is not possible to run this experiment.

original results reported in the paper would have been competitive against all baselines except MultVAE. However, the use of test data as part of the early-stopping process in the provided source code raises concerns about the reliability of this result.

Table 7. Results for HAKG on the Alibaba-iFashion dataset. The baselines are highlighted in **bold** if they outperform our results for HAKG. Results for HAKG are highlighted in **bold** only if they outperform all baselines. Values are underlined if they are better than the results for HAKG that were reported in the original paper.

	Cutoff 20	
	Recall	NDCG
TopPop	0.0312	0.0167
UserKNN CF	0.1090	0.0700
ItemKNN CF	**0.1264**	**0.0818**
$RP^3\beta$	0.1247	**0.0807**
GF-CF	0.1182	0.0742
SLIM	**0.1276**	**0.0832**
NegHOSLIM (EN)	0.1259	**0.0822**
MF-BPR	0.0761	0.0460
iALS	**0.1268**	**0.0807**
MultVAE	**0.1388**	**0.0898**
HAKG paper	**0.1319**	**0.0848**
HAKG original early-stopping	0.1261	0.0787
HAKG our early-stopping	0.1263	0.0789

3.8 Graph Trend Filtering Networks for Recommendation

Fan et al. [19] introduce *Graph Trend Filtering Networks for Recommendation* (GTN), which proposes an adaptive method for assessing the reliability of interactions. To achieve this, a smoothness constraint is applied to the embeddings, penalizing interactions between users and items with very different embeddings. The paper further proposes to use the Proximal Alternating Predictor-Corrector method and formulates an iterative solver that operates through three steps.

Datasets. GTN is evaluated on four datasets: Yelp2018, Amazon-Book and Gowalla (using the same training-test split of LightGCN [28]) and Last-FM (using the same training-test split of KGAT [71]). All datasets are preprocessed with a 10-core selection and splitted with a user-wise random holdout, 72% training, 8% validation and 20% test.

Consistency and Methodology. The GitHub repository[32] contains both the implementation and the training-test data split. The provided material is *not consistent* with what is described in the paper.

First, none of the datasets have splits that follow the procedure described in the paper. The splits of the Amazon-Book, Yelp2018 and Gowalla datasets are not correct random holdout splits, as we also reported for LightGCN in Section 3.1. For consistency, we report results for both the original splits and the correct splits we generated for LightGCN using the procedure described in the paper. Furthermore, the Last-FM dataset exhibits the same issue previously described in Section 3.7 for HAKG, with significant overlap between test and training data. In our experiments we have removed these overlapping interactions from the training set. Finally, the paper does not provide details on how the optimal number of epochs is determined. Additionally, the model's implementation does not employ early-stopping, rather it evaluates the model on the test data every 5 epochs and prints the results.

[32] https://github.com/xiangwang1223/knowledge_graph_attention_network

Reproducibility. In our experiments we could *partially reproduce* the results reported in the original paper. In particular, we could reproduce the results on Yelp2018 and Gowalla, while on Amazon-Book we obtained results that are 10% better than those reported in the paper (see Table 8). For the Last-FM dataset we obtained an NDCG@20 of 0.1776 achieving a 90% improvement over the 0.0857 reported in the paper. This large discrepancy is again due to the removal of the overlap between the training and test data, as discussed in Section 3.1.

Baselines. On Yelp2018, GTN reaches an NDCG@20 of 0.0559, outperforming all baselines with the exception of MultVAE (0.0590) and GF-CF (0.0568). On Last-FM, GTN is not competitive with the baselines with an NDCG@20 of 0.1776 compared to 0.1838 for ItemKNN, and on Amazon-Book it falls substantially behind (see Table 8), achieving nearly half the NDCG of a simple ItemKNN. On Gowalla, GTN outperforms the baselines only on NDCG but is outperformed by MultVAE on Recall. These results are consistent for the data splits we generate ourselves.

Table 8. Results for GTN on the Amazon-Book dataset. The baselines are highlighted in **bold** if they outperform our results for GTN. Results for GTN are highlighted in **bold** only if they outperform all baselines. Values are underlined if they are better than the results for GTN that were reported in the original paper.

	Cutoff 20	
	Recall	NDCG
TopPop	0.0051	0.0044
UserKNN CF	**0.0616**	**0.0518**
ItemKNN CF	**0.0750**	**0.0624**
$RP^3\beta$	**0.0701**	**0.0585**
GF-CF	**0.0710**	**0.0585**
SLIM	**0.0757**	**0.0600**
NegHOSLIM (EN)	**0.0754**	**0.0609**
MF-BPR	0.0254	0.0203
iALS	0.0451	0.0347
MultVAE	**0.0553**	**0.0435**
GTN paper	0.0450	0.0346
GTN our early-stopping	0.0496	0.0384

3.9 Knowledge Graph Contrastive Learning for Recommendation

Yang et al. [78] present the *Knowledge Graph Contrastive Learning framework* (KGCL) framework, which aims to mitigate the impact of noisy knowledge bases. This is achieved by incorporating a knowledge graph augmentation schema to guide the contrastive learning process. KGCL employs a parameterized attention matrix on the concatenation of user and item embeddings to estimate the relevance. Additionally, it utilizes a translation-aware loss function to handle relations within the knowledge base.

Datasets. KGCL is evaluated on three datasets: Yelp2018 (using the same training-test split as in LightGCN and HAKG [18] but a different knowledge base), Amazon-Book and MIND. Both Yelp2018 and Amazon-Book are preprocessed with a 10-core selection, while MIND only contains users with at least 5 interactions within a specific six weeks time frame. The data splitting is not described in the paper but based on the cited papers we assume is user-wise a random holdout, 72% training, 8% validation and 20% test, with the exception of MIND which uses global sampling.

Consistency and Methodology. The GitHub repository[33] contains both the implementation and the training-test data split. The provided material is *partially consistent* with what is described in the paper.

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

First, the splits of the Amazon-Book and Yelp2018 datasets are not the result of a correct random holdout splits, as we also reported for LightGCN in Section 3.1. More importantly, the provided implementation performs early-stopping using the test data, which introduces information leakage. In our experiments, we performed early-stopping using the validation data.

Reproducibility. In our experiments we could *partially reproduce* the results reported in the original paper. In particular, we could reproduce the results on Amazon-Book, while for MIND (see Table 9) and Yelp2018 we obtained lower results within 6%.

Baselines. Many of our simple baselines exhibit better effectiveness than KGCL. The original results reported in the paper would have been mostly competitive with the baselines on Yelp2018, with a reported NDCG@20 of 0.0493, outperformed only by the 0.0521 of MultVAE. However, the use of test data as part of the early-stopping process in the provided source code, combined with the anomalous data splits, raise concerns about the reliability of this result.

Table 9. Results for KGCL on the MIND dataset. The baselines are highlighted in **bold** if they outperform our results for KGCL. Results for KGCL are highlighted in **bold** only if they outperform all baselines. Values are underlined if they are better than the results for KGCL that were reported in the original paper.

	Cutoff 20	
	Recall	NDCG
TopPop	0.0894	0.0437
UserKNN CF	0.0972	0.0509
ItemKNN CF	**0.1225**	**0.0647**
$RP^3\beta$	**0.1187**	**0.0621**
GF-CF	**0.1017**	0.0524
SLIM	**0.1287**	**0.0686**
NegHOSLIM (EN)	**0.1281**	**0.0681**
MF-BPR	0.0888	0.0435
iALS	**0.1130**	**0.0600**
MultVAE	**0.1321**	**0.0700**
KGCL paper	**0.1073**	**0.0551**
KGCL our early-stopping	0.1006	0.0531

3.10 Comparison Between the SIGIR 2022 Analyzed Methods

To provide a more complete picture of the effectiveness of the GNN models we analyze and to separate this assessment from potential confounding factors related to hyperparameter optimization, which, as we have pointed out, is generally extremely limited, we select two datasets and conduct an independent hyperparameter optimization. It is important to note that this type of comparison comes with a significant computational cost, as all the examined methods require iterative training. The experiments we report in this section required approximately six months of GPU time, making this type of evaluation infeasible for many academic or independent researchers.

Datasets. We selected the Amazon-Book and Yelp2018 datasets provided in [71] because they are the only datasets among those we analyzed that also include the required knowledge base for HAKG and KGCL.

Optimization. The hyperparameter optimization and early-stopping follow the same approach used for the baseline methods, as described in Section 2.4. The metric optimized is NDCG@20. The additional material contains the list of hyperparameters, their ranges and distributions.

Table 10. Results for all the analyzed methods and baselines on the Amazon-Book dataset (Table 11) and the Yelp2018 dataset (Table 12). The baselines are highlighted in **bold** if they outperform our results for all the SIGIR 2022 methods we analyze. Results for the SIGIR 2022 methods are highlighted in **bold** only if they outperform all baselines.

Table 11. Experimental results for the Amazon-Book dataset.

	Cutoff 20		Cases
	Recall	NDCG	Explored
TopPop	0.0370	0.0168	-
UserKNN CF	0.2301	0.1371	50
ItemKNN CF	0.2436	**0.1496**	50
RP$^3\beta$	0.2474	**0.1488**	50
GF-CF	-	-	-
SLIM	**0.2511**	**0.1563**	50
NegHOSLIM (EN)	0.2472	**0.1536**	50
MF-BPR	0.1633	0.0911	50
iALS	0.2378	0.1356	50
MultVAE	0.2485	**0.1473**	50
GDE	0.0004	0.0002	50
GTN	0.1852	0.0996	15
HAKG	-	-	-
RGCF	-	-	-
HCCF	0.1328	0.0651	50
INMO	**0.2511**	0.1456	45
KGCL	0.2425	0.1403	13
SimGCL	0.2441	0.1412	38
LightGCN	0.2442	0.1407	34

Table 12. Experimental results for the Yelp2018 dataset.

	Cutoff 20		Cases
	Recall	NDCG	Explored
TopPop	0.0213	0.0132	-
UserKNN CF	0.0988	0.0668	50
ItemKNN CF	0.1057	0.0718	50
RP$^3\beta$	0.1043	0.0693	50
GF-CF	-	-	-
SLIM	0.1007	0.0700	50
NegHOSLIM (EN)	0.0994	0.0667	50
MF-BPR	0.0556	0.0358	50
IALS	0.1120	0.0750	50
MultVAE	0.1188	0.0796	50
GDE	0.0834	0.0535	50
GTN	0.1000	0.0643	17
HAKG	-	-	-
RGCF	-	-	-
HCCF	0.0610	0.0399	50
INMO	**0.1219**	**0.0809**	28
KGCL	0.1125	0.0745	24
SimGCL	**0.1222**	**0.0822**	30
LightGCN	0.1172	0.0781	30

Results. The results of the experiments, along with the number of hyperparameter sets explored within the allotted 14 days, are reported in Table 10. Some results are missing when the method exceeded either the 64GB of RAM available on our server or the 24GB available on our RTX 3090 GPUs. While we attempted to modify the hyperparameter search space to reduce the memory requirements we could not successfully conduct the optimization for those methods. The additional material reports the optimized values of the hyperparameters for all baselines and GNN algorithms.

The results for the Amazon-Book dataset, reported in Table 11, once again show that none of the GNN methods outperform our selection of simple baselines. However, we observe that INMO matches SLIM in terms of Recall and that, for some methods (INMO, KGCL, SimGCL, and LightGCN), the effectiveness gap is much smaller compared to the previous results we obtained in our reproducibility analysis, see Section 3.1, 3.3, 3.5, and 3.9. On the other hand, both HCCF and GTN prove ineffective. While HCCF completed the allotted 50 hyperparameter trials, GTN was limited to only 15 due to its long training time which may partly explain its poor effectiveness. Again, as we previously observed in Section 3.2, GDE exhibits numerical instabilities that prevented it from training properly. Note that slight differences in preprocessing and data splitting make it impossible to directly compare the absolute values of effectiveness metrics across many of the experiments we reported in the previous sections. For example, in the analysis of SimGCL (see Section 3.3), its results with the original hyperparameters showed an NDCG@20 of 0.1047, whereas SLIM achieved 0.1816, a substantial difference. However, in this comparison experiment on Amzon-Book, the two methods are within 2%. A similar trend is observed for INMO, which previously had a result of 0.0934 compared to SLIM's 0.1451, a 55% difference, while in this experiment the results differ by 7%; KGCL, which reached 0.0794 compared to SLIM's 0.1031, a 30% difference, while in this experiment results differ by 11%; and LightGCN, which originally reached 0.0862 compared to SLIM's 0.1838, a whopping 113% difference, while here the results differ by 11%. Given that the data split used here does not appear to exhibit anomalies, we argue that a major reason for this large difference is the ineffectiveness

of the original hyperparameter tuning. Since the weak baselines used for comparison did not present a sufficient challenge, there was little incentive to properly fine-tune the methods. This further reinforces the observation that the experimental practices adopted in the community are often ineffective and unreliable, and allow researchers to support many contradicting conclusions depending on minor changes in the experimental protocol. As discussed by Shehzad and Jannach [63], *"Everyone's a Winner"*, any method can appear competitive if the analysis is conducted carelessly, and current peer review practices do not seem capable of detecting such cases.

The results for the Yelp2018 dataset, reported in Table 12, present a different picture, with two GNN-based methods, INMO and SimGCL, outperforming all baselines. This is partially aligned with our previous analysis of Section 3.3, where SimGCL was the best method on the original (erroneous) Yelp2018 data split and the second-best method after MultVAE on the split we generated. Similarly, INMO, which was originally the second-best method after MultVAE, see Section 3.5, shows improved effectiveness after our optimization. Once again, GDE and HCCF fall significantly behind, with HCCF in particular achieving only half the Recall of the best-performing method, consistent with our original experiments in Sections 3.2 and 3.6. Notably, GTN exhibits very weak effectiveness in this setting, despite ranking as the third-best method after MultVAE and GF-CF in our previous reproducibility experiments in Section 3.8. This drop in effectiveness can be attributed to our hyperparameter optimization being truncated after only 17 trials, as the method exceeded the allotted 14 days.

Overall, the only conclusive statement we can make is that there is little consistency between the results of the experiments conducting according to the original papers and those we conducted in this section. For the Amazon-Book dataset, our optimized GNN models often performed much better than originally reported, though not enough to outperform MultVAE or SLIM, highlighting the insufficiency of the original hyperparameter optimization. A similar, though less pronounced, trend was observed on Yelp2018, where some progress over the baselines could be seen. It seems, therefore, that unless transparent hyperparameter optimization is given greater attention, the results reported in research papers may have little meaning. Overall, if the reported baselines are weak and poorly optimized then the evaluation is conducted in a scenario where the recommendation problem itself is simply not challenging enough. When that is the case, there is no need to push our methods and models forward and, despite the large number of papers published each year, the field risks a long phase of stagnation.

4 DISCUSSION

In this section, we summarize our findings across the following dimensions: artifacts, methodological issues, reproducibility, and baselines.

4.1 Artifacts

Availability. Our study reveals a substantial improvement in the accessibility of the original artifacts, source code, and data splits compared to prior studies [15, 20, 67]. Nearly all of the ten papers analyzed in this study provide the essential artifacts online, with only one paper providing incomplete and non-executable ones. For recommender systems research, the availability of original artifacts has risen from less than 50%, as reported by Ferrari Dacrema et al. [20] for papers published between 2015 and 2018, to 90% in our study.

Consistency of the data artifacts. However, when evaluating the *consistency* of the artifacts with the descriptions in the papers, several issues were identified. Table 13 summarizes the datasets used in each paper we analyzed, the availability of the original training-test data splits, and their consistency with the descriptions provided in the respective papers.

Out of the nine papers that provided complete artifacts, five (LightGCN, SimGCL, HAKG, GTN and KGCL) shared training-test splits that did not appear to be consistent with a correct random holdout splitting process as described in the papers. These unusual splits were not motivated in the papers and could have introduced biases in the evaluation. Non-uniform splits have the potential to alter the distribution or popularity bias of the test set, which may affect the relative effectiveness of the algorithms evaluated. Furthermore, we found it concerning that three papers (GDE, HAKG, and GTN) used at least one dataset where the training and test sets partially overlapped. This was particularly evident for the Last-FM dataset, whose split was sourced from a prior study [71]. These issues lead to evaluations conducted under anomalous conditions as well as potential information leakage between training and testing, both of which could call into question the validity of the results presented in the papers. While reusing datasets and splits from previous publications can facilitate the comparison of results across different studies, it is essential for authors to carefully examine the quality of the splits before utilizing them. In some cases, the paper did not explicitly state that the training-test splits were taken from previous studies, but we were able to discover this through an automated data inspection. We emphasize the importance of explicitly stating when a training-test split is reused from prior work, as this transparency aids in comparing results and identifying potential issues. A further step to streamline the use of datasets and ensure reliance on commonly agreed and verified splits could involve adopting the concept of, or even directly utilizing, `ir_datasets`[34] [45], which provides a standardized interface and API for numerous datasets in Information Retrieval. While relying on shared and commonly available data splits has advantages, it also has disadvantages. A potential risk is that models may overfit to a specific dataset, since research contributions are generally evaluated based on their ability to outperform prior work on a given test set. This can lead to implicit selection bias, where models are optimized to exploit patterns specific to a dataset rather than on their ability to generalize. Furthermore, the more research studies iterate on the same benchmark data split, the higher the risk of overfitting not only on the dataset but also on the specific train-test split [16]. This phenomenon is known as *leaderboard chasing* and has been widely discussed in the Information Retrieval community [8], which leverages several benchmarking datasets some of which, i.e., MS MARCO, have been in use for a decade. Indeed, previous research by Lin et al. [39] observed how several of the best results for the MS MARCO leaderboard are not statistically different. Addressing this issue requires a combination of practices, including the use of multiple test sets, careful statistical validation, and, when feasible, blind evaluation settings where test labels remain hidden from researchers until final submission [8].

As a further issue, papers often do not clearly specify which approach was used to perform a random holdout. In the papers we analyzed, we found two commonly used approaches: a *global* one, where interactions are randomly sampled from the entire dataset, and a *user-wise* one, where the sampling process is performed independently for each user. Splitting the dataset using these two approaches, even when applying the same split percentages, can lead to different distributions, especially if the data is very sparse. For example, consider a user with only two interactions. Depending on the strategy adopted, these interactions may end up in different splits, possibly leaving the user with no iterations in either the training or the test data. If many such cases occur, the data distribution between a global and a user-wise split may differ substantially. Therefore, omitting this information in the paper is an obstacle to correctly reproducing the experimental conditions.

Consistency of the source code artifacts. When considering the consistency of the source code artifacts, our findings are much more positive, with all papers providing implementations of the proposed model that are consistent with what is described in the papers. A summary of our assessment on the consistency of the artifacts is reported in Table

[34] https://ir-datasets.com/

Table 13. Summary of the analysis on the data artifacts summarizing the preprocessing applied, as well as whether the original training-test split was available and consistent with the description in the paper.

Paper	Datasets	K-Cores	Data split protocol		Training-test split	
			Sampling	train-validation-test	Is available?	Is consistent?
LightGCN	Amazon-Book, Gowalla, Yelp2018	10	user-wise	72% - 8% - 20%	Yes	No
GDE	CiteULike-a, Gowalla, MovieLens 100k and 1M, Pinterest	none	global	19% - 1% - 80%	Yes	Yes[a]
SimGCL	Amazon-Book, Douban Book, Yelp2018	10[b]	user-wise	70% - 10% - 20%	Yes	Partially[c]
RGCF	Amazon-Book, MovieLens 1M, Yelp2018	15[d]	global	80% - 10% - 10%	No	-
INMO	Amazon-Book, Gowalla, Yelp2018	10[e]	user-wise	70% - 10% - 20%	Yes	Yes
HCCF	Amazon-Book, MovieLens 10M, Yelp2018	10 or 20[f]	not described	70% - 10% - 20%	Yes	Yes
HAKG	Alibaba-iFashion, Last-FM, Yelp2018	10	not described	80% - 10% - 10% [g]	Yes	Partially[h]
GTN	Amazon-Book, Gowalla, Last-FM, Yelp2018	10	user-wise	72% - 8% - 20%	Yes	No[c] [h]
KGCL	Amazon-Book, MIND, Yelp2018	10[i]	user-wise[j]	72% - 8% - 20%	Yes	Partially[c]

[a]Negligible number of interaction overlap between training and test data for CiteULike-a and Pinterest.
[b]With the exception of Douban Book, where all interactions with a rating of at lest 4 are retained.
[c]The data split for Amazon-Book, Gowalla and Yelp2018 is not consistent and is the same used in LightGCN.
[d]With the exception of MovieLens 1M, where no k-core is applied.
[e]This is preceded by selecting only the interactions with a rating of at least 4. No information is provided on the preprocessing of Gowalla.
[f]Both Yelp2018 and MovieLens 10M are preprocessed with a 10-core selection, while Amazon-Book with a 20-core one.
[g]With the exception of Last-FM which uses a user-wise sampling, 72% - 8% - 20%.
[h]The data split for Last-FM comes from a previous paper [71] outside the scope of this study.
[i]With the exception of MIND.
[j]With the exception of MIND that uses global sampling.

15. However, when the training process is examined, it is not uncommon to find discrepancies. For instance, some papers state that the number of epochs is selected using early-stopping, yet their implementation does not include it. Furthermore, only one paper (INMO) provides an implementation of the hyperparameter optimization used for the proposed model. The remaining nine papers partially list the optimal hyperparameters in the paper and partially in the source code, creating fragmentation. Lastly, HCCF employs hyperparameters outside the search space reported in the paper and limits the number of samples drawn per epoch in a manner that is not documented in the paper. While these inconsistencies might be regarded as minor, they create additional obstacles for the research community in conducting reliable reproducibility studies, even when artifacts are made available.

Documentation. Among the potential obstacles to conducting a reproducibility study, a lack of documentation for the provided artifacts can be a significant challenge. When source code is shared without adequate documentation, it can be extremely difficult to determine how to use it and resolve any issues that arise especially if it contains a complex experimental pipeline involving multiple stages and scripts. In this regard, our findings are largely positive, for nine out of ten candidate papers the artifacts were sufficiently well-organized and documented to allow us to conduct the experiments. The only exception was the artifacts provided by Liu et al. [42], which did not meet this threshold. The source code lacked instructions on the required steps and contained several hard-coded paths for preprocessed data, which were also undocumented, making it impossible for us to proceed.

ACM SIGIR Badges. Overall, we can be conclude that only two out of the ten reviewed papers (GDE and RGCF) meet the requirements for the less demanding of the ACM SIGIR Badges, specifically the *Artifact Evaluated - Functional* badge. In all other cases, the provided artifacts were *inconsistent* in various ways with the contents of the respective papers.

4.2 Methodological Issues

When examining the experimental protocols adopted in the papers, several methodological issues emerge.

Evaluation Procedure. The design of the evaluation procedure involves several decisions, such as selecting datasets and preprocessing methods, determining which effectiveness measures to report, and the cutoffs. Previous studies observed that it was very rare to find two papers employing identical evaluation procedures, even when addressing the same task [20]. In our analysis, we note a slight improvement in this regard, as 8 out of 9 papers report results on commonly used datasets like Amazon-Book or Yelp2018, as shown in Table 13. Conducting part of the experimental analysis on one or two commonly used datasets can greatly improve the transparency of results and facilitates comparisons across papers. Unfortunately, nearly all the papers apply different preprocessing strategies, making direct comparisons of their results almost impossible. For example, while eight out of nine papers utilize the Yelp2018 dataset, it is used with six slightly different preprocessing and splitting techniques, most of which are unique, with the exception of a user-wise holdout with quotas for training, validation and test sets of 72% - 8% - 20% and 70% - 10% - 20% that occur twice. Similarly, although seven papers use Amazon-Book, the dataset is preprocessed and split in five slightly different ways with the most common being a user-wise holdout split with quotas 72% - 8% - 20% that occurs for three papers. The use of inconsistent and unmotivated data preprocessing raises questions about whether a specific preprocessing was chosen based on the goals of the paper and assumptions related to the scenario and task of interest, or whether it was treated as a hyperparameter to be explored in search of a good result, potentially influenced by confirmation bias. Indeed, even a seemingly inconspicuous preprocessing step can significantly affect how the results should be interpreted. Consider for example the second most frequently used dataset, Amazon-Book. In its original form it contains $2.3M$ items, $8M$ users and $22.5M$ interactions corresponding to a density of 10^{-6}, however these numbers change drastically if we apply different commonly used preprocessing strategies: 5-core ($367k$ items, $603k$ users, $8.8M$ interactions, density of $4 \cdot 10^{-5}$), 10-core ($128k$ items, $158k$ users, $4.7M$ interactions, density of $2.3 \cdot 10^{-4}$), or 20-core ($38k$ items, $35k$ users, $1.9M$ interactions, density of $1.4 \cdot 10^{-3}$). When applying 10-core preprocessing, the most commonly adopted one (see Table 13), the density of the dataset increases by a factor of 50 and the statistics change so drastically that, for all practical purposes, these become entirely different datasets. In such cases, the name of the dataset creates a misleading perception of "familiarity" in the evaluation, which may not be accurate at all. Indeed, hidden within this seemingly minor preprocessing detail could be significant limitations of the proposed model, such as performing well only on very dense data, requiring a substantial amount of memory, or exhibiting very limited scalability. As a final observation, which will be further expanded in the following Sections, despite Amazon-Book being the second most used dataset, it is the one where the original and reproduced results of the analyzed message-passing methods perform worse compared to the simple baselines often by a very large margin, while the difference is less pronounced in our independent optimization, see Section 3.10. It is surprising that this could happen for such a frequently used dataset without anyone seeming to take notice.

Model Optimization. A second very common methodological issue is a lack of transparency regarding how the hyperparameters of the proposed method, as well as its number of training epochs, are selected. As shown in Table 14, even though the methods have numerous hyperparameters, up to 18 for KGCL, only a small subset is reported as being tuned. The remaining hyperparameters are typically set to fixed values, often without explanation or with only a generic reference to ensuring a *fair* comparison, without clarifying why such settings should be considered fair. This lack of transparency makes it generally impossible to determine whether the hyperparameter tuning process was conducted reliably. This becomes problematic when the proposed model's results are compared with those of baselines and when it obscures potential limitations of the model, such as requiring particularly careful fine-tuning to achieve competitive results. Indeed, it is known that improper optimization can drastically alter the outcomes of an experiment,

Table 14. Summary of the analysis on the optimization of the model hyperparameters, in particular on how many of the method hyperparameters were stated to be tuned, whether the number of training epochs was selected with early-stopping and whether the provided implementation uses test data during training.

Paper	N. of Hyperparameters tuned	Paper mentions early-stopping?	Implementation uses test data during training?
LightGCN	1 of 9	No	No, but computes results during early-stopping
GDE	3 of 11	Yes	No, but computes results during early-stopping
SimGCL	2 of 10	No	No
RGCF	Not stated	Yes	No
INMO	2 of 12	Yes	No
HCCF	4 of 14	No	No, but computes results during early-stopping
HAKG	3 of 8	Yes	**Yes, to select optimal epochs during early-stopping**
GTN	1 of 12	No	No, but computes results during early-stopping
KGCL	3 of 18	Yes	**Yes, to select optimal epochs during early-stopping**

often being the difference between supporting a conclusion or its opposite [63]. Our independent optimization confirms this observation, yielding results that are substantially different from those obtained with the original hyperparameters and showing a much smaller relative difference compared to the best baselines (see Section 3.10). While the inclusion of ablation studies, which are often reported for a limited number of hyperparameters (2–3), helps and is positive, that alone is not sufficient to ensure transparency.

A particular hyperparameter is the number of epochs for which the model should be trained. This is the only hyperparameter for which it is sometimes possible to identify the selection criteria within the source code artifacts and therefore assess the consistency with what is described in the paper. Several issues emerge in this regard as well. Only two of the analyzed papers (RGCF and INMO) provided implementations with correct early-stopping based on validation data. In two other cases (HAKG and KGCL), the implementations performed early-stopping based on test data, which introduces information leakage and results in an overestimation of the model's effectiveness. For five of the papers we analyzed, there is little to no information on how the number of training epochs was determined, either in the paper or in the source code. Some of these papers include a plot showing how the model converges during training, but with limited details, such as whether the convergence is measured on validation or test data. Only one paper (GDE) specifies the number of training epochs for each dataset, but it provides no information on how these numbers were determined. However, all papers compute and plot the loss function on the test set during training, despite there being no reason to do so, and at the risk of researchers using this information to fine-tune the training process.

Sampled Metrics. On a positive note, we observe that not a single paper has applied *sampled metrics* in their evaluation. Typically the evaluation is performed by ranking all available items in the dataset, while sampled metrics only rank a small number of them (often 1000 or 100) aiming to reduce the computational cost of the evaluation. The use of sampled metrics had become commonplace a few years ago but was later found by Krichene and Rendle [35] to produce results that are inconsistent with the traditional evaluation. They concluded: "*It has shown that most metrics are inconsistent under sampling and can lead to false discoveries. ... For this reason, sampling should be avoided as much as possible during evaluation*". Since then, some efforts have been made to address this issue, but the broader community has largely reacted by abandoning the use of sampled metrics. This constitutes a good example of the community self-healing, by reacting to an issue in the way experiments were conducted and moving away from methods that were shown to produce unreliable results.

4.3 Reproducibility

We could partially reproduce the results for only five of the papers we analyze, namely GDE, SimGCL, RGCF, INMO, and KGCL. For two algorithms (RGCF and HAKG), we were not able to run experiments on all datasets due to high computational resource requirements. Table 15 summarizes the percentage of individual effectiveness metrics that we could reproduce. As shown, there is wide variation, ranging from 0% to 66% depending on the method. While a reproducibility rate of 50% is relatively low, it aligns with the outcomes of previous studies in physics and engineering [6].

One particular issue we would like to raise is that the computational requirements of new methods have been steadily growing for several years. Consider how, in 2009, the Netflix Prize provided a dataset containing 100 million interactions. Yet now, despite continuous technological advancements that have substantially increased the computational capabilities of modern hardware, a large portion of research is conducted on datasets that contain less than 5% of that number. For example, in the data splits used in this study, most datasets (Amazon-Book, Pinterest, Gowalla, MIND, etc.) contain between 1 and 3 million interactions, while some (CiteULike and Douban Book) have fewer than a million. Furthermore, almost all of these datasets include far fewer than 100,000 items and users. While this is not a methodological issue per-se, despite the fact it could be argued that as the data becomes smaller the recommendation problem also becomes *simpler* (e.g., many niche users will disappear), it points to the fact that new methods are so computationally expensive that they are impractical to run on the hardware resources that are available to academics. When this is the case, they will also be too computationally demanding for real-world use on datasets of the scale of the Netflix Prize. This issue limits the ability to conduct reproducibility analyses when hardware constraints make running the experiments impossible or impractical, as we found for RGCF and HAKG. One possible approach to mitigate this issue, when authors are aware that their proposed method has significant computational requirements, is to provide smaller datasets that enable researchers with limited hardware to replicate key findings, preferably from publicly available benchmarks. However, it is important to recognize that *even smaller* datasets will be even less representative of real-world scenarios, furthermore they may not be optimal for training the model effectively, as they could contain dynamics that are *too simple*. This recommendation also applies to industrial research, where publicly releasing datasets may not be feasible. In such cases, it is advisable to provide results for at least one publicly available dataset. However, it is important to note that this dataset may not contain all the information the model should leverage and, as such, can only serve as an imperfect proxy for evaluating the model's effectiveness. A complementary strategy is to share pre-trained models, allowing researchers to validate results without requiring full-scale retraining. While this approach can reduce computational burdens, it comes at the cost of limiting the assessment of the reproducibility of the training and optimization process. Additionally, it introduces the risk of data leakage, as pre-trained models may encode information from datasets that should not be publicly accessible. Lastly, providing full experimental pipelines is advisable and should be encouraged, but it is essential to ensure they are accompanied by adequate documentation and instructions so that other researchers can attempt reproducibility analysis even years after the paper has been published. Overall, while ensuring full reproducibility for independent researchers may not always be feasible due to the high computational requirements of particular models structures or domains, ensuring the ability of other researchers to conduct at least a partial reproducibility analysis under reasonable constraints remains a valuable goal.

The issue of large computational cost also relates to the expectation that such a reproduction should be conducted during the review process. Given the number of papers a reviewer must evaluate, it is unreasonable to assume they will attempt to run the provided source code to assess their reproducibility or even check whether it is executable at all. At best, reviewers can assess the clarity and completeness of the provided artifacts and experimental setup, but verifying

Table 15. Summary of the analysis on the consistency, reproducibility of the results, and the comparison against our set of baselines. In particular, we report a judgment on artifact consistency that combines both the data and the source code, how many of the individual results could be reproduced, how many outperform all baselines and how many baselines are on a given dataset always better than the analyzed method.

Paper	Artifact consistency	N. of results reproduced	N. of results better than all baselines	N. of better baselines on each dataset (min - max)
LightGCN	Partial	4/6 (66%)	0/10 (0%)	1 to 10
GDE	Full	4/10 (40%)	8/10 (80%)	0 to 1
SimGCL	Partial	3/6 (50%)	0/10 (0%)	1 to 11
RGCF	Full	1/4 (25%)	0/4 (0%)	6
INMO	Full	4/9 (44%)	0/9 (0%)	1 to 8
HCCF	Full	2/12 (17%)	0/12 (0%)	9 to 13
HAKG	Partial	0/4 (0%)	0/4 (0%)	5 to 13
GTN	Partial	4/8 (50%)	1/12 (8%)	0 to 10
KGCL	Partial	2/6 (33%)	0/6 (0%)	6 to 12

the results independently remains a broader challenge for the research community. In this respect, it is worth citing the work of Lin and Zhang [40] titled *"Reproducibility is a Process, Not an Achievement: The Replicability of IR Reproducibility Experiments"*, as reproducibility should indeed become a frequent and integral part of a larger discussion any healthy scientific community should have, critically reassessing its achievements and continuously reevaluating its practices. To this end, new approaches are needed. For example, the adoption of Registered Reports,[35] where the methodology is reviewed prior to conducting the experiments allowing the researcher to invest their efforts on the experiments only after the methodology has been approved. This approach could ensure greater methodological transparency and improved rigor by reducing confirmation bias pressure on researchers. Furthermore, shifting the primary review focus from the results to the motivation and methodology would encourage research that is more grounded in hypotheses, which are often overlooked today.

4.4 Baselines

The competitiveness of the message passing methods we analyzed compared to simple baselines appears to be relatively weak. Table 15 presents, for each paper, the percentage of individual metrics in which the best result we obtained for the proposed model outperformed *all* our baselines. Additionally, it reports the number of baselines that were *always* better than the model across all metrics on a given dataset. We can observe that GDE is the only model able to outperform our set of baselines in several measurements (80%), corresponding to all results reported in four out of five datasets. It should be noted however that GDE uses a particular data split where the quota of the data used for training (20%) is much smaller than the testing one (80%), which may put it in a somewhat different scenario compared to the other methods. We also report the minimum and maximum number of baselines that consistently outperform the proposed model across all measurements on a dataset. This number provides a more complete perspective on the relative effectiveness of the model. For example, when GDE is not competitive against the baselines, there is only one baseline that outperforms it, indicating that its effectiveness is good. Similarly, for GTN the number of consistently better baselines has a minimum of 0 because, in one dataset, GTN outperforms all baselines on one measurement but not on others, meaning no single baseline is consistently better than GTN across all metrics. In some cases (SimGCL, INMO, LightGCN) we observe datasets where only one baseline outperforms the proposed model. This baseline is typically MultVAE, which emerges as the single strongest baseline in our study. In these cases, while the message passing methods are not the best overall,

[35] https://www.cos.io/initiatives/registered-reports

they still exhibit relatively strong results. However, for the same methods, there are other datasets where between 8 and 11 baselines outperform them, indicating that their effectiveness is inconsistent and can fall severely behind.

Particularly striking are the results for the Amazon-Book dataset, where most of the analyzed algorithms perform considerably worse than simple baselines. The most notable example is LightGCN, which achieves only *half* the effectiveness of the baselines. This is especially surprising considering that Amazon-Book is the second most commonly used dataset, after Yelp2018. It is to be expected, and indeed quite normal, that certain methods or even multiple methods based on the same underlying architecture may not perform well on specific datasets. However, the current common practice of overlooking stronger baselines creates a misleading impression that the effectiveness of state-of-the-art methods is always improving. In many cases, as shown in our analysis, this is not true. For Amazon-Book, the effectiveness of what are considered state-of-the-art methods has declined substantially. The results of our independent hyperparameter optimization in Section 3.10 confirm this, showing how better optimization, or in some cases any optimization at all, can significantly enhance the effectiveness of the methods. However, the state-of-the-art baselines used in the analyzed papers for Amazon-Book were so weak that there was no incentive to perform even this basic form of optimization. We also suggest that including at least one example of a negative result, along with an explanation of how that dataset or task differs from others, should be encouraged as a valuable scientific practice. This approach would offer valuable insights to the research community, highlighting where the proposed methods are most effective, identifying areas where further development is necessary, and scenarios where the method may not be suitable at all.

Overall, the competitiveness of message passing algorithms against simple baselines appears limited, with only GDE in its unique data split outperforming the baselines on most measurements, and only INMO and SimGCL outperforming the baselines on Yelp2018 in our independent optimization. While some methods, such as GTN, achieve results close to the best-performing baseline (MultVAE), their effectiveness varies greatly across datasets. This suggests that message passing algorithms may be effective only in specific scenarios or that further work is needed to improve their generalizability. It is worth noting, however, that all the analyzed methods apply message passing on relatively similar and simple graph structures, typically the traditional bipartite graph derived from user-item interaction data, with only two methods (HAKG and KGCL) extending this approach to more complex knowledge graphs. In this context it may be relevant to draw an analogy from the experience reported by Steck et al. [65] at Netflix. In the paper, the authors explained that, during Netflix's early experiments with deep learning, it proved challenging to achieve results that outperformed traditional baselines, a situation similar to that reported by Ferrari Dacrema et al. [20]. The breakthrough occurred when the model was enriched with several new features, enabling it to leverage the strengths of deep learning. It is possible that message passing methods are in a similar position when applied for traditional collaborative filtering problems, where the strengths of graph-based approaches may not be utilized effectively, resulting in these methods being less competitive than much simpler baselines. Identifying scenarios where these methods can consistently and robustly demonstrate their advantages, and determining whether further adaptations are necessary to align them with the specific graph topologies and semantics of recommendation problems, remains an open question that only further research and industrial experience can answer. However, as we have discussed, many experimental practices that are currently commonplace and accepted at high-level venues hinder the community's ability to achieve this goal.

4.5 Impact on Follow-up Research

In this section, we conduct a qualitative analysis of how the SIGIR 2022 papers we analyzed influenced subsequent SIGIR 2023 papers that used them as baselines, and whether some of the issues we identified in 2022 persisted in 2023. As

Table 16. Overview of the papers published in SIGIR 2023 that meet our selection criteria described in Section 2, reporting which of the SIGIR 2022 we analyzed they use as baseline, as well as a summary of the preprocessing applied.

Paper	Baselines	Datasets	K-Cores	Data split protocol	
				Sampling	train-validation-test
KRDN [82]	SimGCL, KGCL	Alibaba-iFashion, Last-FM, Yelp2018	not described	not described	not described
MixGCN [70]	KGCL	MovieLens 1M, Last-FM, Book Crossing	1	global	60% - 20% - 20%[a]
AdaMCL [81]	LightGCN, SimGCL, HCCF	Yelp, Amazon-Book, Gowalla, Alibaba-iFashion	15[b]	not described	80% - 10% - 10%
TriSIM4Rec [41]	LightGCN	Amazon-Video, Amazon-Game, MovieLens 1M and 100k	1	global, time-based	80% - 10% - 10%
BSPM [14]	LightGCN, GTN	Gowalla, Yelp2018, Amazon-Book	not described	not described	not described[c]
CoRML [73]	SimGCL	Pinterest, Gowalla, Yelp2018, MovieLens 20M	1[d]	not described	60% - 20% - 20%
VGCL [79]	LightGCN, SimGCL	Douban-Book, Dianping, MovieLens 25M	1[e]	not described	80% - 0% - 20%
DCCF [55]	LightGCN, HCCF	Gowalla, Amazon-Book, Tmall	not described	not described	not described
DGMAE [56]	LightGCN	Youshu, NetEase, Alibaba-iFashion	1	not described	70% - 10% - 20%
CGCL [27]	LightGCN	Gowalla, Yelp2018, Amazon-Book	15[f]	not described	80% - 10% - 10%
GFormer [36]	LightGCN, HCCF	Yelp, Alibaba-iFashion, Last-FM	1	not described	70% - 5% - 25%

[a]The data includes additional *fake* interactions.
[b]The preprocessing is described in another cited paper.
[c]The paper states that the data and train/test split is the same as previous studies, but does not explicitly state which ones.
[d]For MovieLens 20M only users with at least 5 interactions are retained.
[e]Only users with at least 10 interactions are retained.
[f]The preprocessing is described in another cited paper.

opposed to the previous analysis, which focused on evaluating the consistency of artifacts and attempting to reproduce results, this section takes a more qualitative approach, relying on the descriptions provided in each paper.

As described in Section 2, we identified eleven papers from SIGIR 2023, listed in Table 16 alongside the SIGIR 2022 papers they used as baselines. In this section, we will first analyze them along the same dimensions we used for the SIGIR 2022 papers, namely the artifacts availability, evaluation procedure and methodological issues. Finally, we will discuss how the methods from SIGIR 2022 have been used as baseline and attempt to compare their results.

Artifacts Availability. While the proportion of papers that provide publicly available artifacts is higher than observed in other studies and well above 50%, it fluctuates over the years. For SIGIR 2022 the proportion was 100%, whereas for SIGIR 2023 it dropped to 63%, with seven out of eleven papers providing artifacts [14, 27, 36, 55, 73, 81, 82]. We should also note that if the consistency of the artifacts with the descriptions in the papers were evaluated, the number of consistent artifacts would likely decrease, highlighting the need for continued efforts in this area. On a positive note, all seven papers providing artifacts included the data splits or used implementations written with RecBole, an open-source Python library. While relying on open-access libraries introduces potential issues related to the correctness and consistency of the implementations, as is the case with any third-party method implementation, see for example Hidasi and Czapp [30], relying on a limited number of well-maintained libraries can help mitigate this problem. Over time, as issues are identified and corrected, such libraries can contribute significantly to improving the overall quality and reproducibility of research.

Evaluation Procedure. Similarly to what was observed in 2022, the evaluation protocols in the SIGIR 2023 papers vary significantly, see Table 16. Overall, 18 different datasets are used, with Yelp and Gowalla being the most common ones, appearing in 6 and 5 out of eleven papers, respectively. Evaluating methods on common datasets and relying on consistent preprocessing and splitting steps would facilitate easier comparisons of the results reported across different papers. However, we again observe that even when papers use the same dataset, the preprocessing and splitting steps often differ. For example, the most frequently used dataset is Yelp2018, which is however preprocessed in three different ways. A further persistent issue is that detailed information on the data processing may be missing. For instance, three

out of eleven papers [14, 55, 82] do not report the percentage of interactions used for the training-test split, and three out of eleven do not provide details on how the data is preprocessed. There is also a particular instance, in paper [14], where it is stated that to maintain fairness with "previous studies" the same datasets and training test splits are used. Unfortunately, the paper does not reference which previous studies are being referred to, so the reader is left to speculate.

Model Optimization. When analyzing how the model optimization is conducted, we again see strong similarities with what was observed in SIGIR 2022. In most of the SIGIR 2023 papers we analyzed, some form of hyperparameter tuning on a validation set is present. However, many hyperparameters are often fixed a-priori without any particular justification, and most optimizations rely on notoriously inefficient grid searches that explore only a limited number of cases. Typically, key hyperparameters like the learning rate, embedding size, and batch size, critical components of iterative methods, are fixed seemingly arbitrarily or even omitted from the descriptions altogether [27, 36, 55, 56, 70, 73, 79, 81, 82]. These hyperparameters should be carefully tuned, as different configurations can yield to substantially different results for different methods.

Another persistent issue is the lack of transparency in how methods with iterative training determine the number of epochs. Among the papers we analyzed, aside from one article [14] that proposes a method not requiring any training and two articles [27, 81] that properly describe the use of early-stopping, one paper [41] provides incomplete description, while the remaining papers offer no information at all [36, 55, 56, 70, 73, 79, 82].

Regarding the optimization of baselines reported in the papers, it is apparent that insufficient attention is given to clarifying how hyperparameters are selected. Two papers [14, 70] adopt the common yet bad practice of using the baseline hyperparameters values reported in the original papers proposing them, even when, as we very frequently observed, the experimental procedure differs significantly, including the use of different datasets. While tuning hyperparameters based on the methodology in the original paper proposing the method can generally be considered good practice, as done in [82], the frequent occurrences where this tuning was poorly done, ignoring even very important hyperparameters, highlight the need for caution given the importance of this step [63]. Indeed, this tuning should be approached critically to avoid repeating any methodological errors. All other papers simply state that each hyperparameter is tuned properly without providing meaningful details, apart from occasional mentions of the search space for the number of GCN layers in [36, 55]. In one case [14], no information is provided regarding the optimization of baseline hyperparameters.

Papers from SIGIR 2022 used as baselines. Our goal for this section is to assess to what extent irreproducible or weak methods from SIGIR 2022 have been used as baselines in SIGIR 2023. Table 16 lists, for each of the eleven SIGIR 2023 papers we identified, the SIGIR 2022 papers they used as baselines. It is clear that LightGCN is the most influential model, included as baseline in eight out of eleven papers. The second most frequently used baseline is SimGCL, included in four papers, followed by HCCF, which is used in three. In all of these cases, the methods are presented as representing the state-of-the-art, which, as we have observed in this study is not true with very limited exceptions. In our independent optimization, see Section 3.10, HCCF and LightGCN were consistently below the baseline models and SimGCL exhibited competitive results only on Yelp2018. Unfortunately, it is challenging to compare and cross-check the results for the baselines with those reported in the papers that proposed them. For example, one might be interested to assess the consistency in the effectiveness of LightGCN, SimGCL, and HCCF, the three most-used baselines. In their original papers, LightGCN and SimGCL employ splits of 72%-8%-20% on Gowalla, Amazon-Book, and Yelp2018. HCCF, on the other hand, is evaluated on Yelp, Amazon-Book, and MovieLens 10M with a 70%-10%-20% split. When comparing this with the evaluation procedure used in the SIGIR 2023 papers we find that none adopts a data split consistent with

those used in the original baseline papers. Furthermore, one paper [70] uses a 5-fold cross-validation instead of a single hold-out training-test set, while two papers He et al. [27], Yang et al. [79] repeat the experiments 5 times on the same test set. Although these are good practices which allow to measure the variance of the results, they further complicate direct comparisons between papers especially when the variance is not reported. On one hand it is somewhat beneficial that the erroneous preprocessing and splitting steps used by LightGCN and SimGCL do not appear to be adopted in the SIGIR 2023 papers, except possibly [14].[36] However, this comes at the significant cost of losing any hope of comparing results across methods and introduces the continuous risk of new mistakes emerging and propagating before they are identified. Indeed, the outcome of this part of the analysis is that it is not possible to compare the results of the SIGIR 2022 papers we analyzed with those reported as baselines in the SIGIR 2023 papers, which we find worrying.

A further step we can take is to examine the results more qualitatively to assess whether the method produces *similar* results under *similar* data splitting and preprocessing. However, we find this to be highly challenging. Based on the descriptions provided in the papers, we can identify only a handful of cases where such a comparison is possible. While additional comparable protocols may exist, the lack of sufficiently detailed descriptions in the papers means that discovering them would require manually analyzing the provided data. This type of investigative work is an unreasonable expectation for a researcher simply interested in understanding or building upon a paper. The first case is Wei et al. [73], which applies a similar data split to SimGCL [80], holding out 20% of the data for testing but using different partitions for training and validation. The only dataset the two papers have in common is Yelp2018, and the only shared baseline is SimGCL itself. We observe a notable discrepancy in the results for SimGCL, with its NDCG@20 equal to 0.0601 in the original paper but 0.0795 in [73]. Due to the limited details provided on hyperparameter optimization and the use of fixed values for some hyperparameters, we are unable to precisely determine whether this difference stems from variations in the data splits or differences in the optimization process. A more successful comparison can be made with Yang et al. [79], which reports the results of SimGCL and LightGCN on the Douban-Book dataset. Their NDCG@20 values are nearly identical up to the second decimal place. For example, SimGCL achieves 0.1583 in the original paper and 0.1540 in [79], while LightGCN reaches 0.1272 in [80] and 0.1278 in [79]. The results for another baseline, MultVAE, are similarly close, with 0.1103 in SimGCL and 0.1155 in [79]. However, all of these values remain significantly lower than the 0.1694 we obtained for MultVAE in our experiments, which would also slightly outperform the 0.1638 reported for the newly proposed algorithm. The last paper with a similar data split is Li et al. [36], but once again, we observe effectiveness measurements with substantially different absolute values. For example, the original NDCG@20 for HCCF on the Yelp2018 dataset was 0.0510, while [36] reports a lower value of 0.0391. A similar discrepancy is found for LightGCN, where the original result on the (erroneous) Yelp2018 split was 0.0530, whereas in [36], it is 0.0373.

Overall, the outcome of this analysis is that while the SIGIR 2022 papers we examined have been used as baselines to a limited extent, with the exception of the popular LightGCN which however was published at SIGIR 2020, comparing their results with subsequent work from SIGIR 2023 is not feasible. This is due to several factors: (1) the rarity of papers that use the same experimental protocol or data splits in a transparent manner; (2) even when the experimental protocol appears similar, relying on comparable data splits and preprocessing, the reported results may differ significantly, with insufficient details in the papers to pinpoint the cause; and (3) relative comparisons between baselines are also difficult, as each paper tends to report a unique selection of methods with little overlap.

[36] It should be noted that this cannot be confirmed without directly verifying the content of the data splits.

Despite being a research field that heavily relies on experimental findings, the growing body of research on reproducibility shows that these published findings often do not reflect the true effectiveness of a method and therefore provide little guidance for navigating the vast body of available literature. One of the great successes of the scientific method has been moving beyond an *authoritative* conception of science, where arguments were deemed valid based on the prestige of the scholar presenting them. However, in a landscape where reliable points of reference are scarce, results are inconsistent and weak models are frequently published, even at top-tier venues, there is a growing risk that readers will be driven to mainly rely on the reputation of researchers or labs as a proxy to assess the correctness of the paper. This is the opposite of what the scientific process is meant to achieve.

5 CONCLUSION

This study examines the reproducibility of results reported in nine graph-based Recommender Systems papers published at SIGIR 2022. The focus is in particular on the consistency of the provided source code and data artifacts with the descriptions in the papers, the correctness of the experimental methodologies, the reproducibility of the published results, and the competitiveness of the proposed methods against robust baselines. Furthermore, this study explores how these papers may have impacted subsequent work published at SIGIR 2023. Overall, the analysis required considerable experimental efforts, involving the fitting of approximately 25.000 models with a total computation time of 4 years.

The findings of this study highlight significant issues related to artifact consistency and the propagation of poor practices in subsequent publications. Specifically, many of the available training-test data artifacts were clearly the result of an erroneous splitting procedure and were, at the very least, inconsistent with the descriptions provided in the corresponding papers. These data splits, sometimes reused from previous papers, lead to anomalous and questionable evaluation results. When the data split does not appear anomalous, it is typically a unique split created by the authors by applying a specific combination of preprocessing and training-validation-test split percentages, often not clearly described, making comparisons between papers impossible. Indeed, the phenomenon is so prevalent that it is not possible to compare results even between the papers published at SIGIR 2022 and 2023, which we find worrying considering how much of the research literature is based on empirical results. Our analysis also uncovered several bad practices, ranging from arbitrarily setting the values of critical hyperparameters to selecting the number of epochs based on the test data causing information leakage.

When assessing the reproducibility of results, we found it to be poor. Only three out of nine papers were reproducible for at least 50% of their results, with one paper being entirely irreproducible. Lastly, in terms of competitiveness against baselines, most of the analyzed methods demonstrated instances where they were reasonably competitive, alongside cases where they were not. Surprisingly, on the frequently used Amazon-Book dataset most message-passing methods are substantially below simple baselines despite claiming state-of-the-art results in the original papers. This phenomenon has been observed multiple times in previous studies focusing on different methods and remains a persistent issue. It is likely the result of several contributing factors that require further investigation.

Overall, to address these issues, it is imperative for future research to adopt more rigorous standards for artifact documentation and experimental methodology. Ensuring that the provided source code, datasets, and experimental procedures are thoroughly documented and consistent with their description in the paper is of utmost importance. However, it is important to acknowledge that conducting such detailed analyses and reproductions is time-consuming and requires meticulous effort. As such, expecting this level of scrutiny during the peer-review phase is unrealistic, as it would place an unsustainable burden on reviewers. To mitigate this, new approaches are needed. For example, the adoption of Registered Reports and open review tools could improve transparency. Additionally, encouraging the

use of robust and simple baselines over more complex but often weaker ones in many scenarios would lead to more accurate evaluations. Lastly, encouraging the publication and discussion of negative results would help provide a more comprehensive understanding of the strengths and limitations of proposed approaches.

In conclusion, this study underscores the need for improved practices in the publication and evaluation of research in Recommender Systems. It is by addressing these issues that the community can improve the reliability of the published research findings and ensure more robust advancements in the field.

REFERENCES

[1] S. Abadal, A. Jain, R. Guirado, J. López-Alonso, and E. Alarcón. 2022. Computing Graph Neural Networks: A Survey from Algorithms to Accelerators. *ACM Computing Surveys (CSUR)* 54, 9 (December 2022), 191:1–191:38.

[2] E. Amigó, P. Castells, J. Gonzalo, B. A. Carterette, J. S. Culpepper, and G. Kazai (Eds.). 2022. *Proc. 45th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2022)*. ACM Press, New York, USA.

[3] Vito Walter Anelli, Alejandro Bellogín, Tommaso Di Noia, and Claudio Pomo. 2021. Reenvisioning the comparison between Neural Collaborative Filtering and Matrix Factorization. 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, 521–529. https://doi.org/10.1145/3460231.3475944

[4] Vito Walter Anelli, Daniele Malitesta, Claudio Pomo, Alejandro Bellogín, Eugenio Di Sciascio, and Tommaso Di Noia. 2023. Challenging the Myth of Graph Collaborative Filtering: a Reasoned and Reproducibility-driven Analysis. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023, Singapore, Singapore, September 18-22, 2023*, Jie Zhang, Li Chen, Shlomo Berkovsky, Min Zhang, Tommaso Di Noia, Justin Basilico, Luiz Pizzato, and Yang Song (Eds.). ACM, 350–361. https://doi.org/10.1145/3604915.3609489

[5] T. G. Armstrong, A. Moffat, W. Webber, and J. Zobel. 2009. Improvements That Don't Add Up: Ad-Hoc Retrieval Results Since 1998. In *Proc. 18th International Conference on Information and Knowledge Management (CIKM 2009)*, D. W.-L. Cheung, I.-Y. Song, W. W. Chu, X. Hu, and J. J. Lin (Eds.). ACM Press, New York, USA, 601–610.

[6] M. Baker. 2016. 1,500 scientists lift the lid on reproducibility. *Nature* 533 (May 2016), 452–454.

[7] C. D. T. Barros, M. R. F. Mendonça, A. B. Vieira, and A. Ziviani. 2023. A Survey on Embedding Dynamic Graphs. *ACM Computing Surveys (CSUR)* 55, 1 (January 2023), 10:1–10:37.

[8] Christine Bauer, Ben Carterette, Nicola Ferro, Norbert Fuhr, and Guglielmo Faggioli. 2023. Frontiers of Information Access Experimentation for Research and Education (Dagstuhl Seminar 23031). *Dagstuhl Reports* 13, 1 (2023), 68–154. https://doi.org/10.4230/DAGREP.13.1.68

[9] 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.

[10] Michael Benigni, Maurizio Ferrari Dacrema, and Dietmar Jannach. 2025. Diffusion Recommender Models and the Illusion of Progress: A Concerning Study of Reproducibility and a Conceptual Mismatch. *CoRR* abs/2505.09364 (2025). https://doi.org/10.48550/ARXIV.2505.09364 arXiv:2505.09364

[11] T. Breuer, N. Ferro, N. Fuhr, M. Maistro, T. Sakai, P. Schaer, and I. Soboroff. 2020. How to Measure the Reproducibility of System-oriented IR Experiments. In *Proc. 43rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2020)*, Y. Chang, X. Cheng, J. Huang, Y. Lu, J. Kamps, V. Murdock, J.-R. Wen, A. Diriye, J. Guo, and O. Kurland (Eds.). ACM Press, New York, USA, 349–358.

[12] Timo Breuer, Nicola Ferro, Norbert Fuhr, Maria Maistro, Tetsuya Sakai, Philipp Schaer, and Ian Soboroff. 2020. How to Measure the Reproducibility of System-oriented IR Experiments. 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, 349–358. https://doi.org/10.1145/3397271.3401036

[13] H.-H. Chen, W.-J. Duh, H.-H. Huang, M. P. Kato, J. Mothe, and B. Poblete (Eds.). 2023. *Proc. 46th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2023)*. ACM Press, New York, USA.

[14] Jeongwhan Choi, Seoyoung Hong, Noseong Park, and Sung-Bae Cho. 2023. Blurring-Sharpening Process Models for Collaborative Filtering. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 1096–1106. https://doi.org/10.1145/3539618.3591645

[15] Andy Cockburn, Pierre Dragicevic, Lonni Besançon, and Carl Gutwin. 2020. Threats of a Replication Crisis in Empirical Computer Science. *Commun. ACM* 63, 8 (jul 2020), 70–79. https://doi.org/10.1145/3360311

[16] Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, Ellen M. Voorhees, and Ian Soboroff. 2021. TREC Deep Learning Track: Reusable Test Collections in the Large Data Regime. 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, 2369–2375. https://doi.org/10.1145/3404835.3463249

[17] D. De Roure. 2014. The future of scholarly communications. *Insights* 27, 3 (November 2014), 233–238.

[18] 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

[19] 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

[20] 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

[21] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019*, Toine Bogers, Alan Said, Peter Brusilovsky, and Domonkos Tikk (Eds.). ACM, 101–109. https://doi.org/10.1145/3298689.3347058

[22] Maurizio Ferrari Dacrema, Federico Parroni, Paolo Cremonesi, and Dietmar Jannach. 2020. Critically Examining the Claimed Value of Convolutions over User-Item Embedding Maps for Recommender Systems. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (Eds.). ACM, 355–363. https://doi.org/10.1145/3340531.3411901

[23] N. Ferro. 2017. Reproducibility Challenges in Information Retrieval Evaluation. *ACM Journal of Data and Information Quality (JDIQ)* 8, 2 (February 2017), 8:1–8:4.

[24] J. Freire, N. Fuhr, and A. Rauber (Eds.). 2016. *Report from Dagstuhl Seminar 16041: Reproducibility of Data-Oriented Experiments in e-Science*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Germany.

[25] Yunjun Gao, Yuntao Du, Yujia Hu, Lu Chen, Xinjun Zhu, Ziquan Fang, and Baihua Zheng. 2022. Self-Guided Learning to Denoise for Robust 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, 1412–1422. https://doi.org/10.1145/3477495.3532059

[26] E. Gibney. 2020. This AI researcher is trying to ward off a reproducibility crisis. *Nature* 577 (January 2020), 14.

[27] Wei He, Guohao Sun, Jinhu Lu, and Xiu Susie Fang. 2023. Candidate-aware Graph Contrastive Learning for Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 1670–1679. https://doi.org/10.1145/3539618.3591647

[28] 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

[29] José Miguel Hernández-Lobato, Matthew W. Hoffman, and Zoubin Ghahramani. 2014. Predictive Entropy Search for Efficient Global Optimization of Black-box Functions. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger (Eds.). 918–926. https://proceedings.neurips.cc/paper/2014/hash/069d3bb002acd8d7dd095917f9efe4cb-Abstract.html

[30] Balázs Hidasi and Ádám Tibor Czapp. 2023. The Effect of Third Party Implementations on Reproducibility. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023, Singapore, Singapore, September 18-22, 2023*, Jie Zhang, Li Chen, Shlomo Berkovsky, Min Zhang, Tommaso Di Noia, Justin Basilico, Luiz Pizzato, and Yang Song (Eds.). ACM, 272–282. https://doi.org/10.1145/3604915.3609487

[31] 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

[32] Joint Committee for Guides in Metrology (JCGM). 2008. *International vocabulary of metrology – Basic and general concepts and associated terms (VIM)* (3rd ed.). JCGM 200:2012.

[33] S. Kharazmi, F. Scholer, D. Vallet, and M. Sanderson. 2016. Examining Additivity and Weak Baselines. *ACM Transactions on Information Systems (TOIS)* 34, 4 (June 2016), 23:1–23:18.

[34] Andrew V. Knyazev. 2001. Toward the Optimal Preconditioned Eigensolver: Locally Optimal Block Preconditioned Conjugate Gradient Method. *SIAM J. Sci. Comput.* 23, 2 (2001), 517–541. https://doi.org/10.1137/S1064827500366124

[35] Walid Krichene and Steffen Rendle. 2020. On Sampled Metrics for Item Recommendation. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 1748–1757. https://doi.org/10.1145/3394486.3403226

[36] Chaoliu Li, Lianghao Xia, Xubin Ren, Yaowen Ye, Yong Xu, and Chao Huang. 2023. Graph Transformer for Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 1680–1689. https://doi.org/10.1145/3539618.3591723

[37] 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

[38] J. Lin. 2022. Building a Culture of Reproducibility in Academic Research. *arXiv.org, Information Retrieval (cs.IR)* arXiv:2212.13534 (December 2022).

[39] Jimmy Lin, Daniel Campos, Nick Craswell, Bhaskar Mitra, and Emine Yilmaz. 2021. Significant Improvements over the State of the Art? A Case Study of the MS MARCO Document Ranking Leaderboard. 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, 2283–2287. https://doi.org/10.1145/3404835.3463034

[40] Jimmy Lin and Qian Zhang. 2020. Reproducibility is a Process, Not an Achievement: The Replicability of IR Reproducibility Experiments. In *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2020, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 12036)*, Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins (Eds.). Springer, 43–49. https://doi.org/10.1007/978-3-030-45442-5_6

[41] Jiahao Liu, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, Li Shang, and Ning Gu. 2023. Triple Structural Information Modelling for Accurate, Explainable and Interactive Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 1086–1095. https://doi.org/10.1145/3539618.3591779

[42] Xiaoming Liu, Shaocong Wu, Zhaohan Zhang, and Chao Shen. 2022. Unify Local and Global Information for Top-N 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, 1262–1272. https://doi.org/10.1145/3477495.3532070

[43] A. Lucic, M. Bleeker, M. de Rijke, K. Sinha, S. Jullien, and R. Stojnic. 2022. Towards Reproducible Machine Learning Research in Information Retrieval, See [2], 3459–3461.

[44] Q. Lv, M. Ding, Q. Liu, Y. Chen, W. Feng, S. He, C. Zhou, J. Jiang, Y. Dong, and J. Tang. 2021. Are we really making much progress?: Revisiting, benchmarking and refining heterogeneous graph neural networks. In *Proc. 27th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2021)*, F. Zhu, B. Chin Ooi, C. Miao, H. Wang, I. Skrypnyk, and W. Hsu (Eds.). ACM Press, New York, USA, 133–142.

[45] S. MacAvaney, A. Yates, S. Feldman, D. Downey, A. Cohan, and N. Goharian. 2021. Simplified Data Wrangling with ir_datasets. In *Proc. 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*, F. Diaz, C. Shah, T. Suel, P. Castells, R. Jones, T. Sakai, A. Bellogín, and M. Yoshioka (Eds.). ACM Press, New York, USA, 2429–2436.

[46] M. Maistro, T. Breuer, P. Schaer, and N. Ferro. 2023. An in-depth Investigation on the Behavior of Measures to Quantify Reproducibility. *Information Processing & Management* (2023).

[47] National Academies of Sciences, Engineering, and Medicine. 2019. *Reproducibility and Replicability in Science.* The National Academies Press, Washington, USA.

[48] 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

[49] NISO. 2021. *NISO RP-31-2021 – Reproducibility Badging and Definitions.* National Information Standards Organization (NISO).

[50] Open Science Collaboration. 2015. Estimating the reproducibility of psychological science. *Science* 349, 6251 (August 2015), 943–952.

[51] 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

[52] 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

[53] J. Pineau, P. Vincent-Lamarre, k. Sinha, V. Lariviere, A. Beygelzimer, F. d'Alche Buc, E. Fox, and H. Larochelle. 2021. Improving Reproducibility in Machine Learning Research. A Report from the NeurIPS 2019 Reproducibility Program. *Journal of Machine Learning Research (JMLR)* 22, 164 (May 2021), 1–20.

[54] H. E. Plesser. 2018. Reproducibility vs. Replicability: A Brief History of a Confused Terminology. *Frontiers in Neuroinformatics* 11 (January 2018), 76:1–76:4.

[55] Xubin Ren, Lianghao Xia, Jiashu Zhao, Dawei Yin, and Chao Huang. 2023. Disentangled Contrastive Collaborative Filtering. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 1137–1146. https://doi.org/10.1145/3539618.3591665

[56] Yuyang Ren, Haonan Zhang, Luoyi Fu, Xinbing Wang, and Chenghu Zhou. 2023. Distillation-Enhanced Graph Masked Autoencoders for Bundle Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 1660–1669. https://doi.org/10.1145/3539618.3591666

[57] 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

[58] Steffen Rendle, Walid Krichene, Li Zhang, and John R. Anderson. 2020. Neural Collaborative Filtering vs. Matrix Factorization Revisited. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020*, Rodrygo L. T. Santos, Leandro Balby Marinho, Elizabeth M. Daly, Li Chen, Kim Falk, Noam Koenigstein, and Edleno Silva de Moura (Eds.). ACM, 240–248. https://doi.org/10.1145/3383313.3412488

[59] 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

[60] A. Rossi, D. Barbosa, D. Firmani, A. Matinata, and P. Merialdo. 2021. Knowledge Graph Embedding for Link Prediction: A Comparative Analysis. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 15, 2 (April 2021), 14:1–14:49.

[61] 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

[62] Faisal Shehzad, Maurizio Ferrari Dacrema, and Dietmar Jannach. 2025. A Worrying Reproducibility Study of Intent-Aware Recommendation Models. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2025, Padua, Italy, July 13-18, 2025*, Nicola Ferro, Maria Maistro, Gabriella Pasi, Omar Alonso, Andrew Trotman, and Suzan Verberne (Eds.). ACM, 3155–3164. https://doi.org/10.1145/3726302.3730307

[63] Faisal Shehzad and Dietmar Jannach. 2023. Everyone's a Winner! On Hyperparameter Tuning of Recommendation Models. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023, Singapore, Singapore, September 18-22, 2023*, Jie Zhang, Li Chen, Shlomo Berkovsky, Min Zhang, Tommaso Di Noia, Justin Basilico, Luiz Pizzato, and Yang Song (Eds.). ACM, 652–657. https://doi.org/10.1145/3604915.3609488

[64] 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

[65] Harald Steck, Linas Baltrunas, Ehtsham Elahi, Dawen Liang, Yves Raimond, and Justin Basilico. 2021. Deep Learning for Recommender Systems: A Netflix Case Study. *AI Mag.* 42, 3 (2021), 7–18. https://doi.org/10.1609/aimag.v42i3.18140

[66] 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

[67] Victoria Stodden, Jennifer Seiler, and Zhaokun Ma. 2018. An empirical analysis of journal policy effectiveness for computational reproducibility. *Proceedings of the National Academy of Sciences* 115, 11 (2018), 2584–2589.

[68] Gilbert Strang. 2014. *Differential equations and linear algebra.* Wellesley-Cambridge Press Wellesley.

[69] 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

[70] Jihu Wang, Yuliang Shi, Han Yu, Xinjun Wang, Zhongmin Yan, and Fanyu Kong. 2023. Mixed-Curvature Manifolds Interaction Learning for Knowledge Graph-aware Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 372–382. https://doi.org/10.1145/3539618.3591730

[71] 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

[72] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, Benjamin Piwowarski, Max Chevalier, Éric Gaussier, Yoelle Maarek, Jian-Yun Nie, and Falk Scholer (Eds.). ACM, 165–174. https://doi.org/10.1145/3331184.3331267

[73] Tianjun Wei, Jianghong Ma, and Tommy W. S. Chow. 2023. Collaborative Residual Metric Learning. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 1107–1116. https://doi.org/10.1145/3539618.3591649

[74] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui. 2023. Graph Neural Networks in Recommender Systems: A Survey. *ACM Computing Surveys (CSUR)* 55, 5 (May 2023), 97:1–97:37.

[75] 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

[76] 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

[77] W. Yang, K. Lu, P. Yang, and J. Lin. 2019. Critically Examining the "Neural Hype": Weak Baselines and the Additivity of Effectiveness Gains from Neural Ranking Models. In *Proc. 42nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2019)*, B. Piwowarski, M. Chevalier, E. Gaussier, Y. Maarek, J.-Y. Nie, and F. Scholer (Eds.). ACM Press, New York, USA, 1129–1132.

[78] 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

[79] Yonghui Yang, Zhengwei Wu, Le Wu, Kun Zhang, Richang Hong, Zhiqiang Zhang, Jun Zhou, and Meng Wang. 2023. Generative-Contrastive Graph Learning for Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 1117–1126. https://doi.org/10.1145/3539618.3591691

[80] 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

[81] Guanghui Zhu, Wang Lu, Chunfeng Yuan, and Yihua Huang. 2023. AdaMCL: Adaptive Fusion Multi-View Contrastive Learning for Collaborative Filtering. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 1076–1085. https://doi.org/10.1145/3539618.3591632

[82] Xinjun Zhu, Yuntao Du, Yuren Mao, Lu Chen, Yujia Hu, and Yunjun Gao. 2023. Knowledge-refined Denoising Network for Robust Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 362–371. https://doi.org/10.1145/3539618.3591707

[83] Ding Zou, Wei Wei, Xian-Ling Mao, Ziyang Wang, Minghui Qiu, Feida Zhu, and Xin Cao. 2022. Multi-level Cross-view Contrastive Learning for Knowledge-aware Recommender System. 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, 1358–1368. https://doi.org/10.1145/3477495.3532025

Received July 2024