The Datasets Dilemma: How Much Do We Really Know About Recommendation Datasets?

Jin Yao Chin*
s160005@e.ntu.edu.sg
Nanyang Technological University
Singapore

Yile Chen*
yile001@e.ntu.edu.sg
Nanyang Technological University
Singapore

Gao Cong
gaocong@ntu.edu.sg
Nanyang Technological University
Singapore

ABSTRACT

There has been sustained interest from both academia and industry throughout the years due to the importance and practicability of recommendation systems. However, several recent papers have pointed out critical issues with the evaluation process in recommender systems. Likewise, this paper takes an in-depth look at a fundamental but often neglected aspect of the evaluation procedure, i.e. the datasets themselves. To do so, we adopt a systematic and comprehensive approach to understand the datasets used for implicit feedback based top-K recommendation. We start by examining recent papers from top-tier conferences to find out how different datasets have been utilised thus far. Next, we look at the characteristics of these datasets to understand their similarities and differences. Finally, we conduct an empirical study to determine whether the choice of datasets used for evaluation can influence the observations and/or conclusions obtained. Our findings suggest that greater attention needs to be paid to the selection process of datasets used for evaluating recommender systems in order to improve the robustness of the obtained results.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Item Recommendation; Evaluation; Datasets; Data Characteristics

ACM Reference Format:
Jin Yao Chin, Yile Chen, and Gao Cong. 2022. The Datasets Dilemma: How Much Do We Really Know About Recommendation Datasets?. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM '22), February 21–25, 2022, Tempe, AZ, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3488560.3498519

1 INTRODUCTION

As a direct consequence of an increasingly digital lifestyle, recommendation systems have gradually become an indispensable part of our daily lives. This has brought about rapid growth in terms of the recommendation performance across a variety of application scenarios, ranging from e-commerce (e.g. Amazon, eBay) to social media (e.g. Facebook, Instagram). Furthermore, several decades of research has led the community from simple and widely known methods such as WMF [20] and ItemKNN [42] to more advanced algorithms such as Mult-VAE [29] and LightGCN [15].

However, recent studies have pinpointed several worrisome problems that appear to undermine years of hard work within the recommendation systems community. First, the authors in [9] have discovered that many recently proposed methods are in fact not reproducible, and for the remaining methods which could be reproduced, they are often outperformed by simpler but well-tuned baselines. Next, [25] has shown that the results obtained from evaluating item recommendation using a randomly sampled subset of candidate items, which is a *common practice* in recent research, can be inconsistent with results acquired using the complete set of candidate items. The former, also known as *sampled metrics*, is no more than a high-bias, low-variance estimator of its exact version, and results obtained using sampled metrics can be highly misleading as the bias is recommender-dependent. Very recently, [48] highlighted an important issue which is often the cause of unreproducible evaluation and unfair comparison, i.e. the lack of any effective benchmark for evaluation. The authors noted that *researchers often choose different datasets heuristically*, and there are many seemingly trivial factors, e.g. data pre-processing and splitting strategies, evaluation metrics, etc., which can influence the recommendation performance. Although the aforementioned studies have all brought attention to critical issues hampering measurable progress for recommendation in general, there might be other problems that we are oblivious to.

In fact, we are aware of one such problem that has long been overlooked by the community: ***How much do we really know about recommendation datasets?*** To answer this question, we perform various analyses and experiments in this work to improve our understanding of the datasets used for evaluating *top-K item recommendation* in a systematic and comprehensive manner. Specifically, we adopt the following three steps: **(1)** First, we examined the proceedings of 5 top-tier conferences for the past 5 years, and narrowed it down to 48 papers which address the problem of top-K item recommendation based on implicit feedback. There are a total of 45 publicly available datasets which have been used in one or more of these papers, and we performed various analyses to illustrate ***how*** different datasets have been utilised in recent papers. **(2)** After which, we introduce a set of data characteristics which can be used as a representation for each of these datasets, and at the same time, are easily obtained by analysing the (binary) user-item interaction matrix. We leveraged these characteristics to clearly understand ***what*** are the similarities as well as differences between various datasets. **(3)** Finally, we performed an empirical study using a variety of recommendation algorithms to find out ***if*** the choice of

*Corresponding authors.

WSDM '22, February 21–25, 2022, Tempe, AZ, USA

ACM ISBN 978-1-4503-9132-0/22/02…$15.00
https://doi.org/10.1145/3488560.3498519

datasets used for evaluating item recommendation could influence the observations and/or conclusions obtained.

In summary, this paper conducts a retrospective survey on the datasets used for top-K recommendation and seeks to shed light on whether the current practice of selecting datasets arbitrarily would lead to unintended (and perhaps, negative) consequences. To the best of our knowledge, we are the first to provide an extensive overview [1] of recommendation datasets, which is a fundamental but often neglected aspect of the evaluation procedure.

2 COLLECTION AND ANALYSIS

2.1 Paper and Dataset Collection

We start by examining the proceedings of 5 top-tier conferences (KDD, SIGIR, TheWebConf, WSDM, and RecSys) for the past 5 years (2016 to 2020), and gathering a list of long papers focusing on implicit feedback based top-K recommendation. Concretely, we performed a keyword search to narrow down to around 400 papers with titles containing either *'recommend'* or *'collaborative'*. Next, we manually filtered these papers to include only those which **(1)** focuses on implicit feedback top-K recommendation, **(2)** evaluates using classification and/or ranking metrics, such as Precision, Recall, and Normalized Discounted Cumulative Gain (**nDCG**), and most importantly, **(3)** utilises at least one publicly available dataset.In other words, we do not include papers which either **(1)** formulate it as a rating prediction problem, or **(2)** tackle other recommendation tasks, e.g. content-aware or session-based recommendation. As summarised in Table 1, we managed to obtain a total of 48 long papers which satisfy the aforementioned requirements. Collectively, there are 45 publicly available datasets [2] which have been used in one or more of these papers. The strict requirements imposed for collecting & filtering the datasets lead to a useful property for analysing the usage patterns: *A dataset used in any single one of these papers can be used in every other paper as well.* Put simply, there is no reason for a dataset to be inapplicable for any of these papers.

2.2 Dataset Usage Analysis

Figure 1 illustrates the usage patterns of the 45 datasets across all 48 papers using a scatter plot. There are 11 datasets (24.45%) used in 5 or more papers, 10 datasets (22.22%) utilised in 2 to 4 papers, and 24 datasets (53.33%) which have been used in just 1 paper. Unsurprisingly, a handful of these datasets happen to be really popular, e.g. *Netflix*, *MovieLens-1M* (**ML-1M**), *Yelp*, and *MovieLens-20M* (**ML-20M**). However, we can also observe that the majority of these datasets are underutilised.

As for how different datasets have been used in recent papers, we note that there is little to no regularity. On one end, we have papers like [49] and [51] performing their evaluation on 10 datasets and 9 datasets, respectively. On the other end, an overwhelming majority of these papers, i.e. 36 (or 75%) of them, utilised at most 3 publicly available datasets. Furthermore, we use the Apriori algorithm [2] to determine the combinations of datasets which have been used together in 2 or more papers. Table 2 presents the dataset quadruplets and triplets which are commonly used together. First, we have [32] and [33] with 4 datasets in common. However, it should be noted that both papers are written by the same author. Next, there are 7 different dataset triplets which appear in 2 or more papers. Notably, the triplet {*ML-20M*, *Million Song Dataset*, and *Netflix*} has been used together by 5 different papers. In addition, [49] and [51] ended up having just 3 datasets, i.e. {*ML-1M*, *ML-20M*, and *Meetup (NYC)*}, in common despite the substantial number of datasets used for evaluation. Finally, while not shown in Table 2, there are also 21 distinct pairs of datasets which have been used in multiple papers. Amongst them, the most frequent pairing would be {*ML-20M*, *Netflix*}, which has been evaluated at the same time in 9 separate papers.

Overall, although all the aforementioned papers address the exact same problem of implicit feedback based top-K recommendation, our analyses have demonstrated that the choice of datasets is often determined arbitrarily. Even when two papers have been evaluated on the same dataset(s), the results might *not* be comparable due to different data pre-processing and/or splitting strategies [48].

3 DATASETS AND THEIR CHARACTERISTICS

Intuitively, we should rely on a diverse collection of datasets, i.e. in terms of quantifiable dataset characteristics such as *size* and *density*, in order to validate the robustness of any proposed method. Therefore, as described next in Section 3.1, we propose using a more complete set of dataset characteristics as the representation for each dataset. These characteristics are then leveraged in Section 3.2 to illustrate the similarities and differences of different datasets.

3.1 Dataset Characteristics

Even after decades of research, we still lack a thorough understanding of the characteristics of different recommendation datasets. In [1], the authors used an exploratory modelling (**EM**) approach to investigate the impact of different dataset characteristics on the performance of various recommendation algorithms for the *rating prediction* task. More recently, [10] adopted an identical approach in order to assess the *robustness* of several recommender algorithms against *shilling attacks*, using the same characteristics as [1]. Inspired by [1] and [10], we consider two different types of data characteristics which can be easily derived from the user-item interaction matrix in the *implicit feedback* setting.

3.1.1 Structural Characteristics. First, we introduce 3 data characteristics which are based on the structure of the interaction matrix, namely $Space_{log}$, $Shape_{log}$, and $Density_{log}$. For any dataset, we have the set of users $\mathcal{U}$, the set of items $\mathcal{I}$, and the set of interactions $\mathcal{K}$. Structural data characteristics can be easily obtained by building upon basic statistics such as the number of users $|\mathcal{U}|$, the number of items $|\mathcal{I}|$, and the number of interactions $|\mathcal{K}|$.

Definition 1 ($Space_{log}$)**.** For any given dataset, $Space_{log}$ is defined as follows:

$$Space_{log} = log_{10}(\frac{|\mathcal{U}| \text{ x } |\mathcal{I}|}{sc}) \quad (1)$$

whereby sc is a scaling factor to constrain the possible values of $|\mathcal{U}| \text{ x } |\mathcal{I}|$. Following [1], the value of sc is set as 1,000.

[1] Source code and detailed information regarding the datasets used in this paper can be found at https://github.com/almightyGOSU/TheDatasetsDilemma.

[2] Different versions of a dataset are counted only once. For instance, Yelp (https://www.yelp.com/dataset) releases a different version of the dataset for each iteration of its dataset challenge.

Table 1: Recent papers for implicit feedback based top-K recommendation

Conference	Year					Total
	2016	2017	2018	2019	2020	
KDD	-	-	[5, 8]	[6]	[21, 23, 35, 36, 46]	8
RecSys	[7, 34, 41]	[39]	[59]	[12, 33, 37]	-	8
SIGIR	[18, 57]	[3]	[11, 16]	[30, 52, 54]	[4, 13, 15, 44, 47, 53]	14
WSDM	[55, 56]	-	-	[38]	[31, 43, 45, 51, 58]	8
TheWebConf	[28]	[17]	[26, 29, 49]	[50]	[22, 24, 27, 32]	10
Total	8	3	8	9	20	48

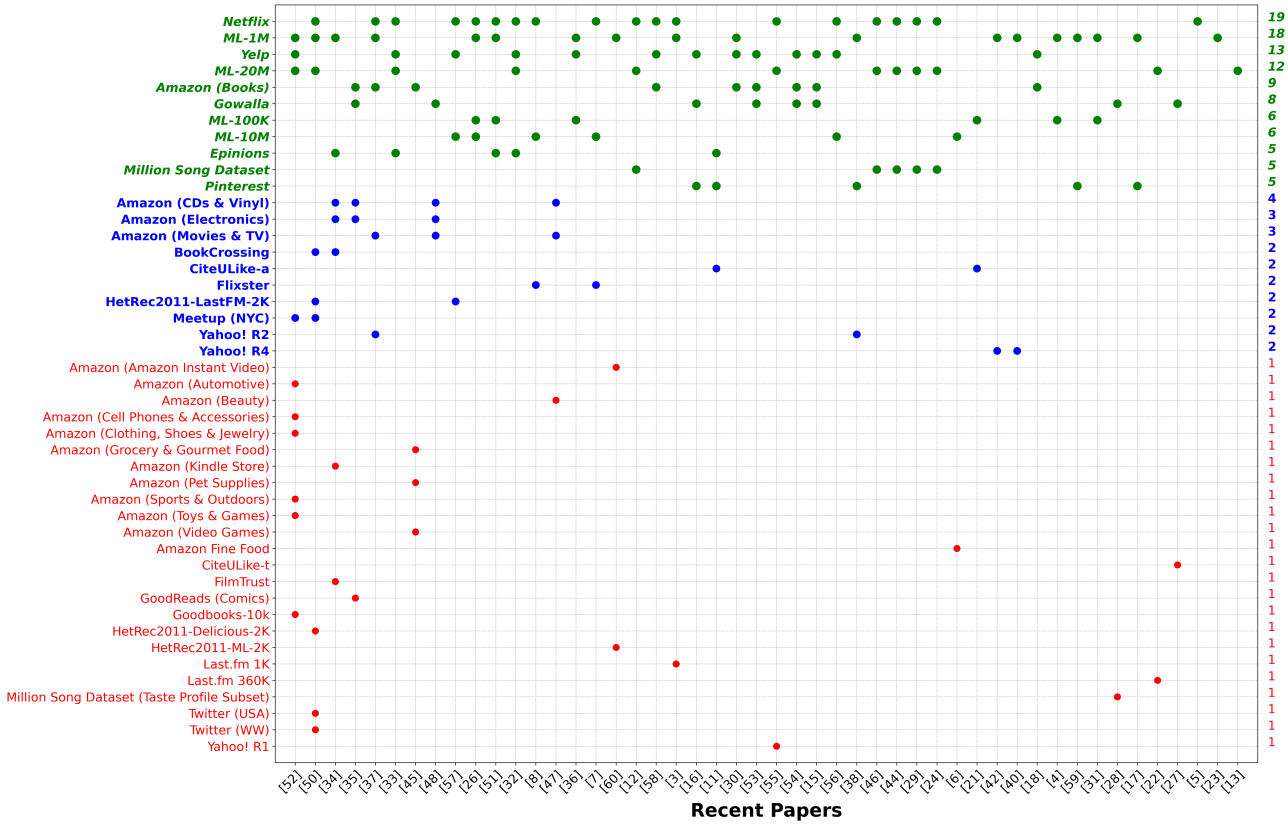

Figure 1: Dataset Usage Patterns (Best viewed in colour)

Table 2: Combinations of datasets used in 2 or more papers

Datasets	Papers
Epinions, ML-20M, Netflix, Yelp	[32, 33]
ML-20M, Million Song Dataset, Netflix	[12, 24, 29, 43, 45]
Amazon (Books), Gowalla, Yelp	[15, 52, 53]
Amazon (CDs & Vinyl; Electronics), Gowalla	[35, 47]
Flixster, ML-10M, Netflix	[7, 8]
ML-100K, ML-1M, Netflix	[26, 50]
ML-10M, Netflix, Yelp	[55, 56]
ML-1M, ML-20M, Meetup (NYC)	[49, 51]

Definition 2 ($Shape_{log}$)**.** For any given dataset, $Shape_{log}$ can be derived as follows:

$$Shape_{log} = log_{10}(\frac{|\mathcal{U}|}{|\mathcal{I}|}) \tag{2}$$

Definition 3 ($Density_{log}$)**.** For any given dataset, we can calculate its $Density_{log}$ as follows:

$$Density_{log} = log_{10}(\frac{|\mathcal{K}|}{|\mathcal{U}| \text{ x } |\mathcal{I}|}) \tag{3}$$

The number of users $|\mathcal{U}|$ and the number of items $|\mathcal{I}|$ can fall within a wide spread of values, ranging from hundreds to even millions of users/items. Similarly, the number of interactions $|\mathcal{K}|$

can be as little as ∼35K (FilmTrust), or as much as ∼100M (Netflix). Therefore, all structural characteristics are log transformed in order to normalise its distributions and enable meaningful comparisons between datasets with significantly different statistics.

Intuitively, $Space_{log}$ reflects the size (and even the scale) of the user-item interaction space, $Shape_{log}$ measures the ratio of users to items, and $Density_{log}$ represents the proportion of observed interactions among all the possible user-item interactions. Structural characteristics are easily interpretable and closely related to one another. For 2 datasets with similar $Space_{log}$ and $Density_{log}$, the dataset with a considerably larger $Shape_{log}$ would have more users and fewer items, and consequently, less observed interactions per user and more observed interactions per item.

3.1.2 Distributional Characteristics. Here, we describe 2 data characteristics, i.e. $Gini_{user}$ and $Gini_{item}$, which captures how the interactions are distributed across both the users and the items.

Definition 4 ($Gini_{user}$)**.** For any given dataset, $Gini_{user}$ is defined as follows:

$$Gini_{user} = 1 - 2 \sum_{u=1}^{|\mathcal{U}|} \left(\frac{|\mathcal{U}| + 1 - u}{|\mathcal{U}| + 1}\right) \text{x} \left(\frac{|\mathcal{K}_u|}{|\mathcal{K}|}\right) \tag{4}$$

where $|\mathcal{K}_u|$ is the number of interactions belonging to the user u.

Definition 5 ($Gini_{item}$)**.** For any given dataset, $Gini_{item}$ can be derived as follows:

$$Gini_{item} = 1 - 2 \sum_{i=1}^{|\mathcal{I}|} \left(\frac{|\mathcal{I}| + 1 - i}{|\mathcal{I}| + 1}\right) \text{x} \left(\frac{|\mathcal{K}_i|}{|\mathcal{K}|}\right) \tag{5}$$

with $|\mathcal{K}_i|$ being the number of interactions belonging to the item i.

$Gini_{user}$ and $Gini_{item}$ utilise the Gini coefficient [14] to measure the distribution of interactions over the set of users $\mathcal{U}$ and the set of items $\mathcal{I}$, respectively. Since these 2 characteristics are similar in nature, we will describe them by using $Gini_{item}$ as an example. $Gini_{item}$ takes on values between 0 to 1 (both inclusive), and a value of 0 implies total equality (i.e. all items are equally popular and have the same number of observed interactions), while a value of 1 indicates maximal inequality whereby all observed interactions belong to a single item. In reality, skewed frequency distributions are rather common across many application scenarios. For example, a blockbuster movie can attract much more viewers as compared to a niche movie with a relatively smaller target audience.

Unlike structural characteristics, distributional characteristics are rarely considered and/or discussed about in most recommendation papers. In practice, there are often multiple measures that can be used for characterising the frequency distributions of the user-item interaction data. Having said that, it was found in [1] that most of these metrics are highly correlated with one another. Therefore, we only consider $Gini_{user}$ and $Gini_{item}$ in this paper.

3.1.3 Influence of Dataset Characteristics. Structural and distributional characteristics are capable of capturing the intricate relationships between users, items, and interactions in a cohesive manner. Notably, their importance should not be underestimated as these data characteristics can affect most recommendation algorithms in one way or another. For example, it is commonly assumed that datasets with higher density would lead to better performance across recommendation algorithms in general as there would be more interactions observed for each user/item under consideration. Similarly, for a dataset with a high $Shape_{log}$ (i.e. more users than items) and $Gini_{user}$ (i.e. there exists some users with many observed interactions), there is a smaller set of candidate items, and at the same time, there would be a handful of active users who have interacted with many of these items. For such datasets, recommender systems adopting a user-based approach such as UserKNN might perform better as (1) having more candidate users increase the likelihood of potentially finding similar users, and (2) it would be easier to find a similar user among these active users with high item coverage. In some way, it is reasonable to expect these dataset characteristics to influence the performance of commonly used algorithms as most of them are still based on the age-old concept of Collaborative Filtering (**CF**) [20], i.e. identifying similar pairs of items and/or like-minded users. Therefore, we believe that it would be both important and beneficial to have an understanding of recommendation datasets from the perspective of data characteristics.

3.2 Similarities and Differences

To the best of our knowledge, there is no existing work which strives to set apart individual recommendation datasets based on their (measurable) similarities or differences. We begin by considering the 45 datasets shown in Figure 1 (Section 2). 3 of these datasets [3] are excluded as they are too small and sparse for any meaningful comparison. Furthermore, we also include the remaining 9 Amazon datasets [4] which have not been utilised in any of the recent papers for top-K recommendation. This leaves us with a total of 51 datasets.

In practice, datasets used for evaluating recommender systems are rarely used as they are. Therefore, we preprocess these datasets before deriving their data characteristics. Most, if not all, research papers would remove the cold-start users/items (i.e. users/items with too few interactions) as there is insufficient information for recommendation systems to provide inference for these extremely inactive users or highly unpopular items. We would like to point out that some of these datasets are made publicly available in a (partially) preprocessed form. For instance, the classic MovieLens datasets (i.e. ML-100K, ML-1M, ML-10M, and ML-20M) do not include users with less than 20 interactions. As for datasets with explicit feedback (e.g. ratings), we simply follow the commonly used approach (e.g. in [7, 37, 39, 59]) of converting all the observed entries into positive interactions to make them suitable for the implicit feedback setting.

3.2.1 Clustering Datasets based on their Characteristics. By using the 5 data characteristics described in Section 3.1, we can visualize the similarities and differences between various datasets based on their pairwise Euclidean distance. However, in order to obtain a better overview of recommendation datasets, we first cluster these datasets before taking a closer look at the characteristics of each cluster. Naturally, datasets in the same cluster are more similar to one another, and at the same time, considerably different from

[3] The 3 datasets that are excluded are {HetRec2011-Delicious-2K; Twitter (USA); Twitter (WW)}. These datasets have between ∼35K to ∼105K interactions and between ∼36K to ∼69K items, and are too small and sparse for meaningful analysis after the data preprocessing step.

[4] The remaining 9 Amazon datasets are {Apps for Android; Baby; Digital Music; Health & Personal Care; Home & Kitchen; Musical Instruments; Office Products; Patio, Lawn & Garden; Tools & Home Improvement}.

the datasets in other clusters. Each dataset is represented by a 5-dimensional vector, i.e. $[Space_{log}, Shape_{log}, Density_{log}, Gini_{user}, Gini_{item}]$, and we use k-means (i.e. based on Euclidean distance) to cluster these 51 datasets. Empirically, we set the number of clusters [5] to 5. Table 3 lists the datasets in each cluster, while Table 4 presents the characteristics of each cluster (i.e. the cluster centroid).

Cluster 1 consists of gigantic but sparse datasets with the most users and items, e.g. Last.fm 360K which has $\sim 359K$ users, $\sim 88K$ items, and $\sim 17M$ interactions. As for **Cluster 2**, it contains datasets with much fewer interactions ($\leq 1M$ interactions) and the number of users and items are often just in the thousands. Unlike other clusters, there are several datasets here with significantly more items than users (hence the negative $shape_{log}$). For instance, CiteULike-t has $\sim 4K$ users, $\sim 7K$ items, and $\sim 82K$ interactions. The datasets in **Cluster 3** have a moderate number of users, items, and interactions, and are relatively sparse as compared to datasets in other clusters. Next, we have **Cluster 4** with just 4 datasets. This is an interesting cluster as the datasets here have the least number of users, items, and interactions, as well as the highest densities. The frequently used ML-100K dataset, which has 943 users, 1,349 items, and 99,287 interactions, turns out to be the dataset with the most interactions within this cluster. Finally, **Cluster 5** contains datasets with a considerable amount of users and items (much lesser than Cluster 1, but slightly more than Cluster 3) and the datasets are quite dense as well. Notably, the datasets in this cluster have a lot more users than items. For example, ML-20M has roughly 7 times more users than items with $\sim 138K$ users and $\sim 18K$ items. In addition, many frequently used datasets such as ML-10M, ML-20M, Netflix, and Pinterest belong to Cluster 5. Last but not least, we can observe that the interactions are usually more evenly distributed across users than items across all clusters. The most popular items can easily amass thousands or even millions of interactions, but it is rather unlikely even for an extremely active user to have that many interactions. By leveraging the various characteristics and focusing solely on the datasets themselves, we are able to put forward a thorough analysis, and consequently, improve our perception of different recommendation datasets substantially.

4 EXPERIMENTS AND RESULTS

Up to this point, we have seen how the choice of datasets is often determined arbitrarily (Section 2) as well as how various datasets can be distinctively different from one another (Section 3). Nonetheless, a key question still remains: Could it actually affect the observations and/or conclusions obtained ***if*** we select the datasets in a different manner? For instance, would we obtain similar results across different recommendation algorithms if we utilise datasets with similar characteristics? In this section, we conduct an empirical study to investigate the performance of several well known recommendation algorithms using a wide variety of datasets with different characteristics.

4.1 Experimental Setup

First, it is impractical to evaluate and analyse the performance of multiple recommendation algorithms on all 51 datasets. Therefore, for each cluster, we select the 3 datasets which are closest to the corresponding cluster centroid based on their Euclidean distances. These selected datasets would have data characteristics which are highly representative of their own cluster, and at the same time, be significantly different from datasets in other clusters. Overall, as shown in Table 5, we conduct the experiments on a total of 15 datasets (5 clusters x 3 datasets).

Next, for each dataset, we adopt the widely used *leave-one-out* approach [16, 17, 23, 49, 50] to construct the training, validation, and testing sets. For each user, the latest interaction is used for testing, the penultimate interaction is used for validation, and all her remaining interactions are used for training. As for datasets without timestamps, the validation and testing interactions for each user are sampled randomly.

For all 15 datasets, we evaluate the performance of 5 different recommendation algorithms as follows:

(1) **UserKNN**: UserKNN [19] is a classic *user neighbourhood-based* CF approach using the Cosine Similarity to find like-minded users. Following [9], both the neighbourhood size and shrinkage term are selected from $[5, 1000]$.
(2) **ItemKNN**: ItemKNN [42] is an *item neighbourhood-based* variant of UserKNN. Similarly, we use the Cosine Similarity, and both the neighbourhood size and shrinkage term are selected from $[5, 1000]$.
(3) **RP3beta**: RP3beta [40] is a simple yet effective *graph-based* method which performs a random walk between users and items based on the observed interaction matrix. The similarity between two items is further divided by each item's popularity raised to the power of a coefficient β. The number of neighbours is set between $[5, 1000]$, and both the damping factor α and the coefficient β are set to real values between $[0, 2]$.
(4) **WMF**: Weighted Matrix Factorization (**WMF**) [20] uses a *latent factor model* approach designed specifically for the implicit feedback setting. The weight for unobserved entries is set to 1, while the weight for observed entries (i.e. *confidence*) is chosen from $[2, 100]$. The number of latent factors is set between $[100, 200]$, and WMF is trained using Alternating Least Squares (**ALS**).
(5) **Mult-VAE**: Mult-VAE [29] is a deep generative model which has been shown to be a strong deep learning-based method [9]. The autoencoder-based architecture for Mult-VAE with 1 hidden layer (of size 600) is $[I \rightarrow 600 \rightarrow 200 \rightarrow 600 \rightarrow I]$, whereby I is the number of items. The hyperparameter β (for KL annealing) is selected from $\{0.1, 0.2, 0.3, 0.5, 1.0\}$, and it is optimised by Adam with a learning rate 0.001 for 200 epochs.

The recommendation algorithms are selected for various reasons: **(1)** They represent different *'families'* of recommendation methods with distinct inductive bias, i.e. *neighbourhood-based*, *graph-based*, *latent factor models*, and *generative models*. **(2)** These straightforward and yet effective algorithms do not require extensive hyperparameter tuning to achieve satisfactory performance. **(3)** Finally, if the hyperparameters are selected adequately [9], conceptually simple methods (e.g. ItemKNN and RP3beta) have been shown to outperform recently proposed methods such as [11, 17, 59] [6].

[5] For the clustering, we considered internal validation measures such as Silhouette Coefficient and Davies-Bouldin Score, and the optimal number of clusters is 4 or 5.

Table 3: Dataset clusters obtained using k-means

Cluster	Datasets	Size
1	Amazon (Books; CDs & Vinyl; Electronics; Kindle Store; Movies & TV); GoodReads (Comics); Last.fm 360K; Million Song Dataset; Yahoo! R1; Yahoo! R2; Yelp	11
2	Amazon (Amazon Instant Video; Automotive; Digital Music; Office Products); Amazon Fine Food; CiteULike-a; CiteULike-t; HetRec2011-LastFM-2K; HetRec2011-ML-2K; Last.fm 1K; ML-1M; Yahoo! R4	12
3	Amazon (Apps for Android; Baby; Beauty; Cell Phones & Accessories; Clothing, Shoes & Jewelry; Grocery & Gourmet Food; Health & Personal Care; Home & Kitchen; Pet Supplies; Sports & Outdoors; Tools & Home Improvement; Toys & Games; Video Games); BookCrossing; Epinions; Gowalla; Meetup (NYC)	17
4	Amazon (Musical Instruments; Patio, Lawn & Garden); FilmTrust; ML-100K	4
5	Flixster; Goodbooks-10K; ML-10M; ML-20M; Million Song Dataset (Taste Profile Subset); Netflix; Pinterest	7

Table 4: Characteristics (i.e. cluster centroid) of all dataset clusters. For individual data characteristic, we rank the clusters based on their corresponding values in descending order.

Cluster	$Space_{log}$	$Shape_{log}$	$Density_{log}$	$Gini_{user}$	$Gini_{item}$
1	7.274 (1)	0.497 (2)	-3.412 (5)	0.477 (2)	0.657 (2)
2	4.340 (4)	-0.134 (5)	-2.162 (3)	0.441 (3)	0.517 (4)
3	5.619 (3)	0.272 (3)	-3.106 (4)	0.337 (4)	0.504 (5)
4	3.167 (5)	0.116 (4)	-1.670 (1)	0.289 (5)	0.557 (3)
5	6.307 (2)	0.878 (1)	-2.120 (2)	0.502 (1)	0.767 (1)

Table 5: Representative datasets for each dataset cluster

Cluster	Representative Datasets
1	Amazon (Electronics; Movies & TV); Last.fm 360K
2	Amazon (Digital Music); CiteULike-t; Yahoo! R4
3	Amazon (Beauty; Toys & Games; Video Games)
4	Amazon (Musical Instruments; Patio, Lawn & Garden); ML-100K
5	Flixster; ML-10M; ML-20M

Nevertheless, we intend to include other advanced algorithms, e.g. HyperML [51] and LightGCN [15], as part of our future work.

The evaluation is performed using both classification and ranking based metrics, i.e. Recall@K and nDCG@K, respectively. Recall@K measures whether the ground truth item is present in the top-K recommendation list. Since we are using the *leave-one-out* approach, Recall@K is equivalent to the Hit Ratio (**HR**) [17]. Other than that, we also consider the Normalized Discounted Cumulative Gain @ K (**nDCG@K**) which accounts for the position of the ground truth item within the top-K recommendation list. As suggested by [25], we use the *exact version* of these metrics (i.e. we rank the ground truth item among all candidate items), and the number of recommended items, i.e. K, is set to 10. Additionally, to determine the ideal hyperparameters for all recommendation algorithms on each dataset, we adopt a Bayesian search strategy [7] based on the nDCG@10 of the corresponding validation set. For each algorithm, we record its Recall@10 and nDCG@10 obtained on the testing set when the corresponding validation nDCG@10 is the highest.

4.2 Experimental Results

Figures 2 and 3 present the Recall@10 and the nDCG@10 of different recommendation algorithms for the 15 datasets, respectively. As the results (i.e. relative performance of different algorithms for each dataset) are nearly identical, we will focus on nDCG@10. Unless stated otherwise, the observations do hold in terms of Recall@10.

For **Cluster 1**, in most cases, RP3beta tends to perform well while WMF performs poorly [8]. While the results for Amazon (Electronics) and Amazon (Movies & TV) are rather similar, we can see that UserKNN and Mult-VAE does surprisingly well for the Last.fm 360K dataset. The Last.fm 360K dataset is structurally similar to those in Cluster 5, i.e. with a large number of interactions as well as much more users than items, and this could have contributed to the stronger performance of UserKNN and Mult-VAE. As for **Cluster 2**, the graph-based RP3beta is always the best performing method. On the *Yahoo! R4* dataset, RP3beta and UserKNN performs similarly (i.e. the difference is not statistically significant). When it comes to **Cluster 3**, there are a number of interesting observations. First, across all 3 datasets, RP3beta has the best performance while Mult-VAE has the worst performance. Next, both UserKNN and ItemKNN perform significantly worse (with a 3% to 8% relative difference) than RP3beta. At the same time, both UserKNN and ItemKNN perform significantly better (with a 3% to 15% relative difference) than WMF. Finally, WMF performs much better (with a 16% to 27% relative difference) than Mult-VAE. Note that for all 3 datasets, the difference between UserKNN and ItemKNN is never statistically significant. For **Cluster 4**, the relative performance of each algorithm

[6] In addition to the 5 algorithms reported in this paper, we experimented with several representative neural network-based methods, i.e. **CMN** [11], **NCF** [17], and **LRML** [49]. However, in line with the findings from [9], these methods performed poorly across all datasets despite extensive hyperparameter tuning. Furthermore, the time complexity of **CMN** [11] renders it computationally prohibitive for the larger datasets in Clusters 1 and 5. As such, we omit their results from the paper.

[7] Specifically, we use a wrapper function from https://scikit-optimize.github.io, and for each algorithm *(except Mult-VAE whereby Bayesian search is not required)*, we execute a total of 35 runs whereby the first 5 runs are random initial points. The Bayesian search strategy facilitates efficient hyperparameter tuning for all recommendation algorithms and reduces the likelihood of subpar performance due to poorly chosen hyperparameters.

[8] Whenever we state that method A performs better (or worse) than method B, the difference between those two methods is statistically significant with $p\text{-}value < 0.05$ using the paired sample t-test.

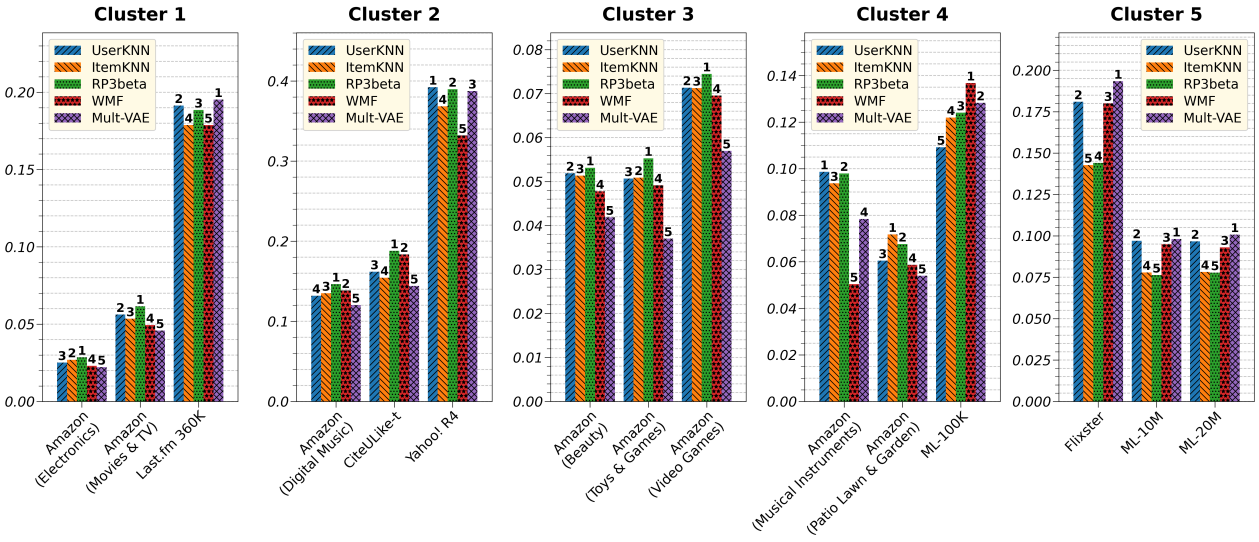

Figure 2: Recall@10 of different recommendation algorithms for the 15 datasets (Best viewed in colour)

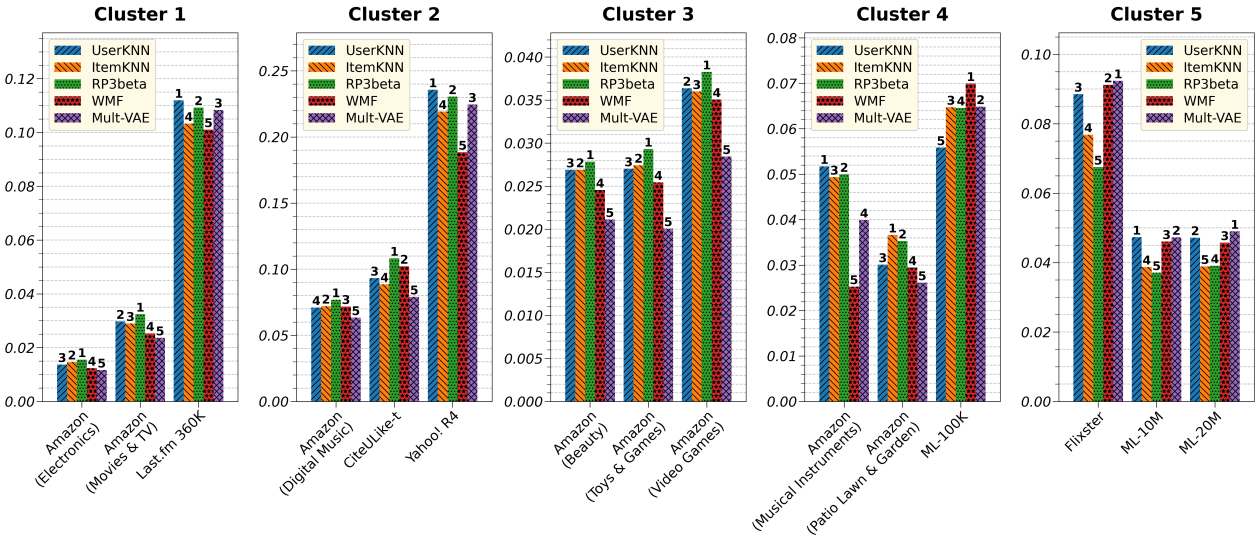

Figure 3: nDCG@10 of different recommendation algorithms for the 15 datasets (Best viewed in colour)

changes drastically across different datasets. In other words, for this particular cluster, we do not have any algorithm which consistently performs better or worse than all the other algorithms. Finally, for **Cluster 5**, we can immediately see the results are drastically different from that of Cluster 3. Across all 3 datasets in Cluster 5, Mult-VAE, UserKNN, and WMF perform incredibly well and hold a 20% relative improvement over ItemKNN and RP3beta. This is in stark contrast with the results of Cluster 3, whereby RP3beta triumphs over remaining algorithms while Mult-VAE consistently ranks last.

Our results highlight a clear and consistent trend: For datasets with relatively similar characteristics, there could be a particular recommendation algorithm which tends to perform significantly better (or worse) than all remaining algorithms. As a matter of fact, it seems entirely plausible that a very different *'ordering'* of recommendation algorithms [9] can be obtained just by selecting datasets from different clusters (e.g. Cluster 3 vs. Cluster 5). Consequently, it could lead to completely different observations and/or conclusions.

4.2.1 Relating the Recommendation Performance for each Cluster to its Data Characteristics. In order to have a better understanding of the results, we provide a few examples relating the performance of various recommender algorithms to the data characteristics of a particular dataset cluster.

[9] The *'ordering'* for **Cluster 3** would be *RP3beta > UserKNN, ItemKNN > WMF > Mult-VAE*, while the *'ordering'* for **Cluster 5** would be *Mult-VAE, UserKNN, WMF > ItemKNN, RP3beta.*

As mentioned in Section 3.2.1, the datasets in **Cluster 3** have very moderate characteristics in terms of the number of users, items, and interactions. At the same time, the relatively low $Gini_{user}$ and $Gini_{item}$ scores imply that the observed interactions are much more equally distributed across both users and items. For these datasets, their corresponding user-item bipartite graphs would have much higher connectivity, and it would be less likely to encounter multiple disconnected vertices (either users or items) when performing the random walks. We hypothesise that it could be the one of the reasons behind the exceptionally strong performance of the *graph-based* algorithm RP3beta for this particular cluster. Additionally, since we have a similar number of users and items in these datasets, both *neighbourhood-based* approaches, i.e. the user-based UserKNN and the item-oriented ItemKNN, have nearly identical performance.

Next, for **Cluster 4**, we have noted that the performance of different algorithms appears to be relatively unstable across the individual datasets. We believe that this could be attributed to the extremely small dataset sizes as the largest dataset in this cluster, i.e. ML-100K, has just $\sim 100K$ interactions as its name implies. As such, it is unlikely for any particular method to learn well enough from the training data, which then leads to unreliable and seemingly random predictions.

Finally, for **Cluster 5**, the *user neighbourhood-based* UserKNN performs incredibly well as it could be easier to find like-minded users among the large number of candidate users in these datasets with an exceptionally high $shape_{log}$ (i.e. much more users than items). If we keep the number of interactions (and hence, density) fixed and simply adjust the ratio of users to items, the feasibility of deriving either the user-user similarity matrix or the item-item similarity matrix would naturally be affected [1].

To summarise, as different recommendation algorithms have distinct inductive bias, i.e. *neighbourhood-based*, *graph-based*, *latent factor models*, or *generative models*, the data characteristics would inherently affect them differently. As such, it would be fair to assume that other methods, e.g. HyperML [51] and LightGCN [15], would be just as susceptible to the changes in the characteristics of a dataset.

4.3 Suggestion

Our findings have shown that the approach used for selecting datasets could in fact influence the observations and/or conclusions obtained from the experimental evaluation. As researchers may face different constraints (e.g. limited to datasets from a specific domain) when performing their experiments, we believe that restricting everyone to a common set of benchmark datasets may not be the ideal solution. Instead, given the large variety of publicly available datasets to choose from, we strongly suggest using datasets with considerably different characteristics as part of the evaluation process. For example, as shown in Section 2.2, frequent dataset combinations such as {*ML-10M*, *Netflix*} and {*ML-20M*, *Netflix*} are all made up of large and dense datasets from Cluster 5. Therefore, the results could be biased, and the same model might not work well for other scenarios.

5 RELATED WORK

Experimental Issues in Recommender Systems. Lately, several papers have pointed out critical experimental issues related to recommender systems. For example, [9] has shown that many recently proposed methods (which are often based on deep learning) are in fact not reproducible and often fail to outperform simple, but fine-tuned, baseline methods such as ItemKNN [42] and RP3beta [40]. [25] has shown that *sampled metrics*, i.e. ranking the relevant items together with a smaller set of random items in order to speed up the computation of metrics, leads to an inaccurate evaluation and should be avoided. [48] takes a broader view of the entire evaluation process and analyses the influence of different factors, e.g. utilised datasets, evaluation metrics, hyperparameter tuning strategies, etc., on the recommendation performance. However, the experiments in [48] are performed on 6 *'popular'* datasets and evaluated using *sampled metrics*, and hence the observations might not be completely accurate.

Data Characteristics of Recommendation Datasets Although we make use of similar data characteristics as [1] and [10], there are several notable differences: **(1)** We focus on the task of *implicit feedback* based top-K recommendation, while [1] and [10] consider rating prediction and robustness against shilling attacks, respectively, using *explicit feedback*. **(2)** We adopt a dataset-centric approach and conduct our empirical study on 15 distinctly different datasets, while both papers utilise 4 arbitrarily chosen datasets. **(3)** Most importantly, we evaluate various recommendation algorithms on the *'complete'* datasets, i.e. just as it would be used in any other empirical study. On the other hand, these 2 papers use a random sampling approach to extract multiple submatrices from the original user-item interaction matrix. The data characteristics and recommendation performance are then derived from these submatrices in order to fit a linear regression model as part of their Exploratory Modelling (**EM**) approach. However, the characteristics as well as performance obtained via the submatrices could in reality end up being very different from that of the actual interaction matrix.

6 CONCLUSION

In this paper, we conducted a retrospective survey on datasets used for implicit feedback based top-K recommendation. We demonstrated how various datasets have been utilised rather arbitrarily in recent papers, and illustrated the similarities and differences between many widely known datasets using a set of well-defined data characteristics. Additionally, we investigated whether the datasets used, with respect to their data characteristics, could in fact lead to different observations and/or conclusions drawn from an empirical study. Our findings have shown that data characteristics do have a significant impact on the obtained results, and we strongly suggest using datasets with considerably different characteristics to improve the robustness of the evaluation process. As part of our future work, we would like to consider more recommendation algorithms as part of the empirical study, as well as other recommendation tasks, e.g. content-aware or session-based recommendation.

ACKNOWLEDGMENTS

The authors would like to thank the reviewers for their meticulous reviews and invaluable suggestions. This research was supported in part by the MOE Tier-2 grant MOE2019-T2-2-181 and the MOE Tier-1 grant RG114/19 (S).

REFERENCES

[1] Gediminas Adomavicius and Jingjing Zhang. 2012. Impact of Data Characteristics on Recommender Systems Performance. *ACM TMIS* 3, 1, Article 3 (April 2012), 17 pages.

[2] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo. 1996. *Fast Discovery of Association Rules.* American Association for Artificial Intelligence, USA, 307–328.

[3] Rocío Cañamares and Pablo Castells. 2017. A Probabilistic Reformulation of Memory-Based Collaborative Filtering: Implications on Popularity Biases. In *SIGIR '17.* 215–224.

[4] Dong-Kyu Chae, Jihoo Kim, Duen Horng Chau, and Sang-Wook Kim. 2020. AR-CF: Augmenting Virtual Users and Items in Collaborative Filtering for Addressing Cold-Start Problems. In *SIGIR '20.* 1251–1260.

[5] Chaochao Chen, Ziqi Liu, Peilin Zhao, Longfei Li, Jun Zhou, and Xiaolong Li. 2018. Distributed Collaborative Hashing and Its Applications in Ant Financial. In *KDD '18.* 100–109.

[6] Yihong Chen, Bei Chen, Xiangnan He, Chen Gao, Yong Li, Jian-Guang Lou, and Yue Wang. 2019. λOpt: Learn to Regularize Recommender Models in Finer Levels. In *KDD '19.* 978–986.

[7] Evangelia Christakopoulou and George Karypis. 2016. Local Item-Item Models For Top-N Recommendation. In *RecSys '16.* 67–74.

[8] Evangelia Christakopoulou and George Karypis. 2018. Local Latent Space Models for Top-N Recommendation. In *KDD '18.* 1235–1243.

[9] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. In *RecSys '19.* 101–109.

[10] Yashar Deldjoo, Tommaso Di Noia, Eugenio Di Sciascio, and Felice Antonio Merra. 2020. How Dataset Characteristics Affect the Robustness of Collaborative Recommendation Models. In *SIGIR '20.* 951–960.

[11] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative Memory Network for Recommendation Systems. In *SIGIR'18.* 515–524.

[12] Ehtsham Elahi, Wei Wang, Dave Ray, Aish Fenton, and Tony Jebara. 2019. Variational Low Rank Multinomials for Collaborative Filtering with Side-Information. In *RecSys '19.* 340–347.

[13] Chen Gao, Chao Huang, Dongsheng Lin, Depeng Jin, and Yong Li. 2020. DPLCF: Differentially Private Local Collaborative Filtering. In *SIGIR '20.* 961–970.

[14] Corrado Gini. 1921. Measurement of Inequality of Incomes. *The Economic Journal* 31, 121 (1921), 124–126.

[15] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR '20.* 639–648.

[16] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial Personalized Ranking for Recommendation. In *SIGIR'18.* 355–364.

[17] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW '17.* 173–182.

[18] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *SIGIR'16.* 549–558.

[19] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. 1999. An Algorithmic Framework for Performing Collaborative Filtering. In *SIGIR '99.* 230–237.

[20] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *ICDM '08.* 263–272.

[21] Shuyi Ji, Yifan Feng, Rongrong Ji, Xibin Zhao, Wanwan Tang, and Yue Gao. 2020. Dual Channel Hypergraph Collaborative Filtering. In *KDD '20.* 2020–2029.

[22] Jyun-Yu Jiang, Patrick H. Chen, Cho-Jui Hsieh, and Wei Wang. 2020. Clustering and Constructing User Coresets to Accelerate Large-Scale Top-K Recommender Systems. In *WWW '20.* 2177–2187.

[23] Manas R. Joglekar, Cong Li, Mei Chen, Taibai Xu, Xiaoming Wang, Jay K. Adams, Pranav Khaitan, Jiahui Liu, and Quoc V. Le. 2020. Neural Input Search for Large Scale Recommendation Models. In *KDD '20.* 2387–2397.

[24] Farhan Khawar, Leonard Poon, and Nevin L. Zhang. 2020. Learning the Structure of Auto-Encoding Recommenders. In *WWW '20.* 519–529.

[25] Walid Krichene and Steffen Rendle. 2020. On Sampled Metrics for Item Recommendation. In *KDD '20.* 1748–1757.

[26] Dongsheng Li, Chao Chen, Qin Lv, Hansu Gu, Tun Lu, Li Shang, Ning Gu, and Stephen M. Chu. 2018. AdaError: An Adaptive Learning Rate Method for Matrix Approximation-Based Collaborative Filtering. In *WWW '18.* 741–751.

[27] Defu Lian, Haoyu Wang, Zheng Liu, Jianxun Lian, Enhong Chen, and Xing Xie. 2020. LightRec: A Memory and Search-Efficient Recommender System. In *WWW '20.* 695–705.

[28] Dawen Liang, Laurent Charlin, James McInerney, and David M. Blei. 2016. Modeling User Exposure in Recommendation. In *WWW '16.* 951–961.

[29] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *WWW '18.* 689–698.

[30] Chenghao Liu, Tao Lu, Xin Wang, Zhiyong Cheng, Jianling Sun, and Steven C.H. Hoi. 2019. Compositional Coding for Collaborative Filtering. In *SIGIR'19.* 145–154.

[31] Feng Liu, Huifeng Guo, Xutao Li, Ruiming Tang, Yunming Ye, and Xiuqiang He. 2020. End-to-End Deep Reinforcement Learning Based Recommendation with Supervised Embedding. In *WSDM '20.* 384–392.

[32] Huafeng Liu, Liping Jing, Jingxuan Wen, Zhicheng Wu, Xiaoyi Sun, Jiaqi Wang, Lin Xiao, and Jian Yu. 2020. Deep Global and Local Generative Model for Recommendation. In *WWW '20.* 551–561.

[33] Huafeng Liu, Jingxuan Wen, Liping Jing, and Jian Yu. 2019. Deep Generative Ranking for Personalized Recommendation. In *RecSys '19.* 34–42.

[34] Ramon Lopes, Renato Assunção, and Rodrygo L.T. Santos. 2016. Efficient Bayesian Methods for Graph-Based Recommendation. In *RecSys '16.* 333–340.

[35] Chen Ma, Liheng Ma, Yingxue Zhang, Ruiming Tang, Xue Liu, and Mark Coates. 2020. Probabilistic Metric Learning with Adaptive Margin for Top-K Recommendation. In *KDD '20.* 1036–1044.

[36] Khalil Muhammad, Qinqin Wang, Diarmuid O'Reilly-Morgan, Elias Tragos, Barry Smyth, Neil Hurley, James Geraci, and Aonghus Lawlor. 2020. FedFast: Going Beyond Average for Faster Training of Federated Recommender Systems. In *KDD '20.* 1234–1242.

[37] Athanasios N. Nikolakopoulos, Dimitris Berberidis, George Karypis, and Georgios B. Giannakis. 2019. Personalized Diffusions for Top-n Recommendation. In *RecSys '19.* 260–268.

[38] Athanasios N. Nikolakopoulos and George Karypis. 2019. RecWalk: Nearly Uncoupled Random Walks for Top-N Recommendation. In *WSDM '19.* 150–158.

[39] Rasaq Otunba, Raimi A. Rufai, and Jessica Lin. 2017. MPR: Multi-Objective Pairwise Ranking. In *RecSys '17.* 170–178.

[40] Bibek Paudel, Fabian Christoffel, Chris Newell, and Abraham Bernstein. 2016. Updatable, Accurate, Diverse, and Scalable Recommendations for Interactive Applications. *ACM TiiS* 7, 1, Article 1 (Dec. 2016), 34 pages.

[41] Shameem A. Puthiya Parambath, Nicolas Usunier, and Yves Grandvalet. 2016. A Coverage-Based Approach to Recommendation Diversity On Similarity Graph. In *RecSys '16.* 15–22.

[42] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-Based Collaborative Filtering Recommendation Algorithms. In *WWW '01.* 285–295.

[43] Ilya Shenbin, Anton Alekseev, Elena Tutubalina, Valentin Malykh, and Sergey I. Nikolenko. 2020. RecVAE: A New Variational Autoencoder for Top-N Recommendations with Implicit Feedback. In *WSDM '20.* 528–536.

[44] Shaoyun Shi, Weizhi Ma, Min Zhang, Yongfeng Zhang, Xinxing Yu, Houzhi Shan, Yiqun Liu, and Shaoping Ma. 2020. Beyond User Embedding Matrix: Learning to Hash for Modeling Large-Scale Users in Recommendation. In *SIGIR '20.* 319–328.

[45] Harald Steck, Maria Dimakopoulou, Nickolai Riabov, and Tony Jebara. 2020. ADMM SLIM: Sparse Recommendations for Many Users. In *WSDM '20.* 555–563.

[46] Jianing Sun, Wei Guo, Dengcheng Zhang, Yingxue Zhang, Florence Regol, Yaochen Hu, Huifeng Guo, Ruiming Tang, Han Yuan, Xiuqiang He, and Mark Coates. 2020. A Framework for Recommending Accurate and Diverse Items Using Bayesian Graph Convolutional Neural Networks. In *KDD '20.* 2030–2039.

[47] Jianing Sun, Yingxue Zhang, Wei Guo, Huifeng Guo, Ruiming Tang, Xiuqiang He, Chen Ma, and Mark Coates. 2020. Neighbor Interaction Aware Graph Convolution Networks for Recommendation. In *SIGIR '20.* 1289–1298.

[48] Zhu Sun, Di Yu, Hui Fang, Jie Yang, Xinghua Qu, Jie Zhang, and Cong Geng. 2020. Are We Evaluating Rigorously? Benchmarking Recommendation for Reproducible Evaluation and Fair Comparison. In *RecSys '20.* 23–32.

[49] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent Relational Metric Learning via Memory-Based Attention for Collaborative Ranking. In *WWW '18.* 729–739.

[50] Thanh Tran, Xinyue Liu, Kyumin Lee, and Xiangnan Kong. 2019. Signed Distance-Based Deep Memory Recommender. In *WWW '19.* 1841–1852.

[51] Lucas Vinh Tran, Yi Tay, Shuai Zhang, Gao Cong, and Xiaoli Li. 2020. HyperML: A Boosting Metric Learning Approach in Hyperbolic Space for Recommender Systems. In *WSDM '20.* 609–617.

[52] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR'19.* 165–174.

[53] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. 2020. Disentangled Graph Collaborative Filtering. In *SIGIR '20.* 1001–1010.

[54] Ga Wu, Maksims Volkovs, Chee Loong Soon, Scott Sanner, and Himanshu Rai. 2019. Noise Contrastive Estimation for One-Class Collaborative Filtering. In *SIGIR'19.* 135–144.

[55] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. 2016. Collaborative Denoising Auto-Encoders for Top-N Recommender Systems. In *WSDM '16.* 153–162.

[56] Yao Wu, Xudong Liu, Min Xie, Martin Ester, and Qing Yang. 2016. CCCF: Improving Collaborative Filtering via Scalable User-Item Co-Clustering. In *WSDM '16.* 73–82.

[57] Hanwang Zhang, Fumin Shen, Wei Liu, Xiangnan He, Huanbo Luan, and Tat-Seng Chua. 2016. Discrete Collaborative Filtering. In *SIGIR'16.* 325–334.

[58] Yuan Zhang, Xiaoran Xu, Hanning Zhou, and Yan Zhang. 2020. Distilling Structured Knowledge into Embeddings for Explainable and Accurate Recommendation. In *WSDM '20.* 735–743.

[59] Lei Zheng, Chun-Ta Lu, Fei Jiang, Jiawei Zhang, and Philip S. Yu. 2018. Spectral Collaborative Filtering. In *RecSys '18.* 311–319.