Exploring Data Splitting Strategies for the Evaluation of Recommendation Models

Zaiqiao Meng
University of Glasgow
zaiqiao.meng@glasgow.ac.uk

Richard McCreadie
University of Glasgow
richard.mccreadie@glasgow.ac.uk

Craig Macdonald
University of Glasgow
craig.macdonald@glasgow.ac.uk

Iadh Ounis
University of Glasgow
iadh.ounis@glasgow.ac.uk

ABSTRACT

Effective methodologies for evaluating recommender systems are critical, so that different systems can be compared in a sound manner. A commonly overlooked aspect of evaluating recommender systems is the selection of the data splitting strategy. In this paper, we both show that there is no standard splitting strategy and that the selection of splitting strategy can have a strong impact on the ranking of recommender systems during evaluation. In particular, we perform experiments comparing three common data splitting strategies, examining their impact over seven state-of-the-art recommendation models on two datasets. Our results demonstrate that the splitting strategy employed is an important confounding variable that can markedly alter the ranking of recommender systems, making much of the currently published literature non-comparable, even when the same datasets and metrics are used.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; *Test collections.*

KEYWORDS

Recommender Systems, Spliting Strategy, Model Evaluation, Leave-one-out, Temporal Split

ACM Reference Format:
Zaiqiao Meng, Richard McCreadie, Craig Macdonald, and Iadh Ounis. 2020. Exploring Data Splitting Strategies for the Evaluation of Recommendation Models. In *Fourteenth ACM Conference on Recommender Systems (RecSys '20), September 22–26, 2020, Virtual Event, Brazil.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3383313.3418479

1 INTRODUCTION

Recommender systems (RecSys) have been subject to extensive research examining how to most effectively find items of interest that a user would like to buy or consume within large datasets. Recommendation spans a range of domain-specific sub-tasks (such as grocery recommendation [24] and venue recommendation [11]) and different scenarios (such as session-based recommendation [29] and sequential recommendation [14]). Many approaches have been proposed to solve these tasks over the last two decades, among which neural network-based recommendation models are currently very popular, due to their high effectiveness and adaptability to different sub-tasks and scenarios [29]. As the recommender systems field matures, advances in performance naturally become more incremental, leading to smaller increases in model effectiveness. This places more strain on the evaluation methodology's ability to distinguish between systems with similar performance, as researchers and practitioners chase ever smaller performance gains.

With the current influx of very similar neural network-based recommendation models being published, there needs to be increased emphasis placed on eliminating confounding factors that can lead to uncertainty during evaluation, otherwise it will be impossible to confidently determine whether gains are truly being made. In the Information Retrieval (IR) domain, standardization efforts such as TREC, and other evaluation initiatives like NTCIR, CLEF and FIRE laid down guidelines on what constitutes a sound evaluation methodology in that domain. However, standardization efforts in the recommender systems domain appear to have been less successful, with most current research papers reporting a wide-range of distinct combinations of datasets, metrics, baselines and data splitting strategies, which makes it difficult to measure progress in the field [4, 17, 29].

Standardization of datasets and baselines within the RecSys community is an on-going process. In particular, while recent works [4, 17, 18] tend to share similar baseline models (e.g. some variant of BPR [15]) and in some cases may share datasets, there are no commonly agreed-upon standards for important aspects that can impact performance such as data preparation. Indeed, a recent study [18] found that suitably tuned baselines could in some cases match or out-perform state-of-the-art approaches, highlighting the importance of hyper-parameter tuning and standardized benchmarks [3, 4, 17, 18] to enable fair comparisons and reproducibility. However, beyond these known issues, one factor that is often overlooked (and typically is not detailed sufficiently in prior works to be reproducible) is the *data splitting strategy* employed. This is how a recommendation dataset is split into training, validation and testing sets. In the IR domain, this split is usually explicitly defined by the test collection (i.e. training and test query sets). However, there is often no equivalent guidance in RecSys scenarios, leading to a wide range of strategies for dividing any particular dataset being employed and reported [14, 20, 21, 29]. Hence, it is natural to ask 'does the data splitting strategy matter?', because if it does, much of the recently published work is not comparable, even when performances are

RecSys '20, September 22–26, 2020, Virtual Event, Brazil

ACM ISBN 978-1-4503-7583-2/20/09.
https://doi.org/10.1145/3383313.3418479

reported under the same dataset and metrics. As such, in this paper, we make an analysis of data splitting strategies for next-item/basket recommendation tasks, with the aim of answering this question.

Indeed, when analysing the literature, we found many inconsistencies in terms of the rankings of different state-of-the-art neural recommendation models [4]. Furthermore, some prior works [14, 29] have indicated that an arbitrary choice of dataset split removes (temporal) recommendation signals that some models aim to leverage. We hypothesize that some of the inconsistencies observed may be caused by particular models being sensitive to the data splitting strategy employed. To validate our hypothesis, we collect and analyze the commonly used data splitting strategies among the state-of-the-art recommendation approaches (particularly the recent neural network approaches) and conduct a comprehensive comparison of algorithms' performance under these strategies.

The contribution of this work is twofold: (1) we report an analysis of recent recommendation literature to illustrate the large variance of data splitting strategies currently being employed; (2) we make a comprehensive analysis of the performance for several state-of-the-art recommendation models over three different data splitting strategies to evaluate the impact of those strategies. Indeed, our analysis highlights the often ignored limitation that the leave one last and the temporal user split strategies have, namely that they 'leak' evidence from future interactions into the model during training. Furthermore, we demonstrate that these different data splitting strategies strongly impact the ranking of systems under the same dataset and metrics - confirming that the data splitting strategy is a confounding variable that needs to be standardized. We also provide best practice recommendations for future researchers based on our analysis.

2 DATA SPLITTING STRATEGIES IN RECOMMENDATION MODELS

Among the different recommendation system evaluation approaches available, "offline" evaluation using historical item ratings or implicit item feedback are by far the most common [3]. As this method relies on a dataset of prior explicit or implicit interactions and current models are based on supervised learning, the dataset needs to be split into training, validation and testing sets. We summarize the four main data splitting strategies from the literature below:

Leave One Last: As its name suggests, leave one last data splitting extracts the final transaction per user for testing, where the second last transaction per user is normally used as validation and the remaining transactions can be used for training. There are two common Leave One Last strategies employed based on the type of transaction involved:

- *Leave One Last **Item**:* Under Leave One Last Item, a transaction corresponds to one $\langle user, item \rangle$ pair per-user. This is one of the most commonly reported strategies in the literature for item-based recommendation tasks. For example, NeuMF [5], CTRec [1] and JSR [28] models were reported using this data splitting strategy.
- *Leave One Last **Basket/Session**:* Under Leave One Basket/Session Out a transaction corresponds to a basket or session (i.e. a $\langle user, [item_1, ..., item_K] \rangle$ tuple) for each user. This strategy is commonly reported in scenarios where an interaction represents a set of items bought together (e.g. in grocery recommendation) where the last basket per user in the dataset is used for testing (e.g. FPMC [16] and Triple2vec [24]).

Among our analysed papers, leave one last data splitting (either item or basket) was the most popular (8 out of 17). The advantage of these data splitting strategies is that they maximize the number of transactions in the dataset that can be used for training. On the other hand, as only the last transaction per user is leveraged for testing, test performance may not reflect the overall recommendation effectiveness for a user over time. This also impacts training, as validation on such a small sample may not be sufficiently robust to enable consistent convergence into an effective and generalizable model. Moreover, the leave-one-last splitting strategies allow interaction data from the future to be used during training. This 'temporal leaking' phenomenon is undesirable from an experimental perspective, as for example, the model can learn about the popularity of an item BEFORE it becomes popular. Figure 1 illustrates this effect.

Temporal User/Global Split: The temporal split strategy is another commonly used evaluation approach that splits the historical interactions/baskets by percentage based on the interaction timestamps (e.g. the last 20% of interactions are used for testing). However, there are two variations of this strategy, which we denote temporal user and temporal global:

- *Temporal **User**:* Temporal user-based splitting is very similar to the leave one last strategy, but with the distinction that a percentage of the last interactions/baskets of each user are reserved for testing, rather than just one. Models such as VAECF [10], SVAE [19] and NGCF [26]) were originally evaluated under this strategy. It is important to note that while temporal user-based splitting does consider the global interaction timestamps, it is still not a realistic scenario since the train/test boundary can vary considerably between users, resulting in the same 'temporal leaking' phenomenon discussed above.
- *Temporal **Global**:* On the other hand, temporal global splitting defines a fixed time-point that is shared across all users, where any interactions after that point are used for testing. VBCAR [12] and DCRL [27] use this strategy and earlier work considered this to be the most strict and realistic setting [2]. However, one limitation of the temporal global splitting is that after calculating the intersection between the training and testing sets (as users/items may no longer exist in both), the total number of users and items retained is much smaller than under the Leave One Last strategies (see Table 2 for an example on the Tafeng dataset), meaning fewer transactions are available for training/validation/testing.

Random Split: As the name suggests, random splitting randomly selects the training/test boundary per-user [15, 23, 26]. Early recommender systems were evaluated using a leave one variant of this scheme [15], where only one random item per user is selected for testing. However, this scheme has been gradually abandoned in favour of using the last (in time) interaction (i.e. Leave One Last Item) for each user. One limitation of random splitting strategies is that they are not reproducible unless the data splits used are released by the author(s).

User Split: The user split strategy is another less common evaluation approach that splits the dataset by user rather than by interaction. In this case, particular users (and hence their transactions) are reserved for training, while a different user set (and their transactions) are used for testing. Few works use this strategy, as it requires that the underlying models have the capability to recommend items

Table 1: Overview of data splitting strategies reported in the literature, as well as the dataset(s) those papers use.

Model	Leave One Last		Temporal Split		Random Split	User Split	Used Datasets
	Item	Basket/Session	User-based	Global			
BPR [15] (2009)	×	×	×	×	√	×	N
FPMC [16] (2010)	×	√	×	×	×	×	-
NeuMF [5] (2017)	√	×	×	×	×	×	M1, P
VAECF [10] ((2018))	×	×	√	×	×	√	M2, N
Triple2vec [24] (2018)	×	√	×	×	×	×	I, D
SARRec [7] (2018)	√	×	×	×	×	×	A, M1
CTRec [1] (2019)	√	√	√	×	×	×	T, A
SVAE [19] (2019)	×	×	√	×	×	√	M1, N
BERT4Rec [22](2019)	√	×	×	×	×	×	A, M1, M2
NGCF [26] (2019)	×	×	×	×	√	×	A, G, Y
VBCAR [12] (2019)	×	×	×	√	×	×	I
KGAT [25] (2019)	×	×	√	×	×	×	A, Y
Set2Set [6] (2019)	×	×	√	×	×	×	T, D
DCRL [27] (2019)	×	×	×	√	×	×	M2, G
TiSASRec [9] (2020)	√	×	×	×	×	×	M1, A
JSR [28] (2020)	√	×	×	×	×	×	M2, A
HashGNN [23] (2020)	×	×	×	×	√	×	M2, A

M1: Movielens-1M, M2: Movielens-20M, T: Tafeng, D: Dunnhumby
G: Gowalla, I: Instacart N: Netflix, A: Amazon, Y: Yelp, P: Pinterest

for new (cold-start) users, which many approaches do not support. It is also notable that some papers (e.g. VAECF [10] and SAVE [14]) that use this strategy, still split the interaction history of the training users into *fold-in* and *fold-out* sets, such that users with partial histories are included during training. These works suffer from the same issue of "future data" leaking into the model during training as with the temporal user-based strategy.

To provide an overview of where these strategies are being used, we analyze seventeen prior papers that propose and evaluate recommendation models (focusing on recent neural network approaches) and categorize them by the data splitting strategies employed. Table 1 summarizes what strategies are employed by each prior work. As we can see from Table 1, there is little in the way of consistency in terms of the data splitting strategy used/reported, even in cases where two works use the same datasets[1]. For example, the VAECF [10] and TiSASRec [9] models use the same Movielens-1M dataset, but are tested under leave one last and temporal splitting strategies respectively. Furthermore, we can see from Table 1 that very few (2 out of 17) models are being evaluated using what is considered to be the most realistic splitting strategy [2], namely temporal global splitting.

3 EVALUATION METHODOLOGY

Having shown that prior works report performance under a wide range of data splitting strategies, we next answer our primary research question: 'Does the data splitting strategy matter?' In the remainder of this section we describe our experimental setup for answering this question.

Data Split: We experiment with three of the most popular data splitting strategies discussed above, namely: *leave one last item*; *leave one last basket*; and *global temporal split* strategies. User-based temporal split produces near-identical splits to leave one last item, and hence we exclude it to save space. Meanwhile, we omit the user split scheme since it is both rarely used and mandates a very different evaluation pipeline [14].

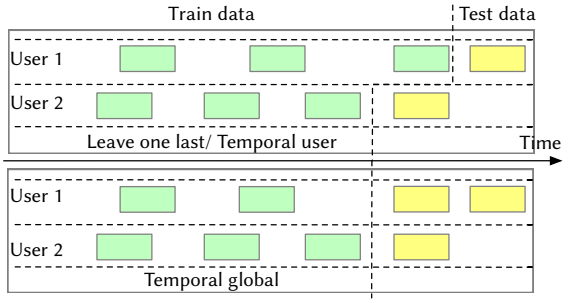

Figure 1: Temporal global split v.s. Temporal user split/Leave one last.

Datasets: We conduct experiments on two real-world grocery transaction datasets, namely the *Tafeng*[2] and *Dunnhumby*[3] datasets, which both contain the needed information (i.e. interactions, baskets and timestamps) for the three data splitting strategies we examine [24]. For the leave one last item/basket data splitting strategies, we first filter items that were purchased less than 10 times, then use the most recent item/basket for testing, the second recent item/basket for validation and the remaining items/baskets for training. For the global temporal split, any user that has purchased less than 30 items and/or has less than 10 baskets is filtered out, and any item that was purchased less than 20 times is removed, following [12]. Then, we split all the baskets for each of the datasets into training (80%) and testing (20%) subsets based on time order, where the last 20% of the training subset is used for validation. Note that under the global temporal split strategy, the number of test users is further reduced, since only users that have an item/basket after the global temporal boundary are used (this particularly impacts

[1]Note that we only list those datasets that are still commonly used (appeared twice) in the recent literature.

[2]http://www.bigdatalab.ac.cn/benchmark/bm/dd?data=Ta-Feng
[3]http://www.dunnhumby.com/careers/engineering/sourcefiles

Table 2: Dataset statistics used in our experiments. Interactions are reported by training/validation/test sets post sampling.

Dataset	Data split	#Users	#Items	#Baskets	#Interactions
Tafeng	raw data	9,238	7,973	77,202	464,118
	leave one item	9,238	7,857	-	444,207 / 9,238/ 9,238
	leave one basket	9,238	7,857	58,654	346,378 / 58,076 / 58,229
	global temporal	1,997	2,017	20,190	83,374 / 26,408 / 18,107
Dunnhumby	raw data	2,500	92,339	2,764,842	2,595,732
	leave one item	2,492	23,404	-	2,379,184 / 2,492 / 2,492
	leave one basket	2,486	23,404	261,976	2,330,466 / 26,610 / 26,951
	global temporal	2,162	25,393	84,128	715,007 / 156,476 / 169,578

the Tafeng dataset). We report the statistics of each dataset under each splitting strategy in Table 2.

Testing Models: To determine the impact of the splitting strategy, we experiment with a set of seven recommenders from the literature. First, we include two classical models (i.e. NMF [8] and BPR [15]). Second, we select three state-of-the-art neural item recommendation models that have been shown to be effective (NeuMF [5], VAECF [10] and NGCF [26]). Finally, we include two state-of-the-art neural grocery recommendation models (Triple2vec [24] and VBCAR [12]). All models support all splitting strategies. For our first experiment using all seven models, for algorithms that have hyper-parameters, we use the recommended values from the original works (either the associated paper or source code). For the second experiment, we tune NeuMF [5], Triple2vec [24] and VBCAR [12] using different hyper parameter settings (embedding size, learning rate, activator, optimiser and alpha values).

Evaluation Metrics: The two commonly used ranking metrics, NDCG@10 and Recall@10, are used to evaluate the performance for each model. To quantify differences in pairs of model rankings for the different splitting strategies we also report ranking correlation via **Kendall's** τ.

4 RESULTS

In Section 2 we demonstrated that prior works in item recommendation use very different data splitting strategies, even in cases where the dataset is the same. This is problematic, since even if the dataset and metrics reported are the same in two different papers, the performance numbers may not be comparable due to the confounding variable that is the splitting strategy. Hence, in this section, we investigate what impact the splitting strategy has on a range of classical and state-of-the-art recommendation models. In particular, we compare the ranking of systems produced under three commonly used splitting strategies: leave one last item, leave one last basket and temporal global split. If the ranking of systems significantly differ between splitting strategies, then this serves to demonstrate that much of the recent work in the recommendation space is not comparable, and hence there is a growing need for evaluation standardization.

Table 3 reports the ranking of 7 recommendation models from the literature under four scenarios (the combination of two datasets and two evaluation metrics) for each of the three data splitting strategies. The rows are sorted by performance under leave one last (item) splitting, where the up/down arrows indicate relative rank position swaps and the number in brackets indicates the number of ranks moved. As we can see from Table 3, under all four scenarios, rank swaps are observed between systems. For example, for the Dunnhumby dataset under Recall@10, the worst model under leave one last item (NMF) is ranked three places higher under leave one last basket, passing BPR, VAECF and NGCF. Indeed, we observe swaps occurring for all pairs of splitting strategies, and more worryingly, these swaps seem to cluster around the most effective models for each scenario - where being able to accurately distinguish systems is critical. Moreover, we observe that there is a pattern to the occurring swaps - Triple2vec appears particularly favored under leave one last item, while VBCAR ranks much higher under temporal evaluation. This behaviour is likely being caused by both how the instances are being selected (e.g. whether evidence from the future is available when training) and the differing train/validation/test distributions (see Table 2). Hence, this provides evidence both that the splitting strategy is an important factor that impacts reported recommendation performance, and that a splitting strategy may favour particular systems.

However, so far we have only considered 7 recommendation systems. With such a small sample size, these observed swaps could have simply occurred by chance. Hence, to test this, we perform a correlation experiment between a much larger sample of recommendation systems. In particular, we take three of the more effective learning algorithms (NeuMF, VBCAR and Triple2Vec), and generated 230 models by varying their hyperparameters, providing a larger sample set to compare. Figure 2 plots the NDCG@10 performance of these models for pairs of splitting strategies across each of the two datasets, as well as reporting Kendall's τ correlation between the score distributions for each. If a pair of splitting strategies produces a similar ranking of systems, we would expect a Kendall's τ value close to 1.0 and the data points to align close to the linear trend line. As we can see from Figure 2, the Kendall's τ correlations between the pairs of splitting strategies are only moderate, ranging between 0.5284 and 0.7630, demonstrating that there are many rank swaps taking place. Additionally, we can see that at the higher end of the effectiveness scale (top-right of each chart), there is greater horizontal point dispersion than vertical point dispersion. This means that leave one last item data splitting is producing a wider distribution of NDCG@10 and Recall@10 scores, while the temporal and leave one last basket splitting seems to group systems in a more stratified manner. This does not indicate that one strategy is better than another, but is evidence that these splitting strategies are in effect evaluating very different aspects of recommendation.

To conclude, we have shown that the ranking of state-of-the-art systems is strongly affected by the data splitting strategy employed, and hence, is a confounding variable that needs to be accounted for when comparing recommendation systems. We have also observed some evidence that certain splitting strategies may favour particular systems[4].

[4]This is an important direction for future work.

Table 3: Performance comparison of recommendation models under different data splitting strategies. Models are sorted by performance under leave one last (item) splitting, arrows indicate rank position swaps relative to that performance.

Model	Tafeng Dataset, NDCG@10		
	Leave One Item	Leave One Basket	Temporal Global
NMF	0.0879	0.0796	0.1811
BPR	0.1347	0.1987	0.2575
VAECF	0.1580	0.2309	0.2858
NeuMF	0.1738	0.2504	0.3313
VBCAR	0.1739	0.2549	0.3744▼(1)
NGCF	0.1852	0.2726▼(1)	0.3794▼(1)
Triple2Vec	0.1978	0.2555 ▲(1)	0.3569▲(2)

Model	Tafeng Dataset, Recall@10		
	Leave One Item	Leave One Basket	Temporal Global
NMF	0.1739	0.0969	0.1671
BPR	0.2470	0.2306	0.2338
VBCAR	0.2835	0.2633▼(1)	0.3129▼(3)
VAECF	0.2861	0.2651▼(1)	0.2655▲(1)
Triple2Vec	0.2957	0.2622▲(2)	0.3055▲(1)
NeuMF	0.3125	0.2881	0.3110▲(1)
NGCF	0.3364	0.3112	0.3655

Model	Dunnhumby Dataset, NDCG@10		
	Leave One Item	Leave One Basket	Temporal Global
BPR	0.1354	0.2413	0.5264▼(1)
NMF	0.1448	0.2496	0.4327▲(1)
VAECF	0.1455	0.2620	0.5790
NGCF	0.1480	0.2647	0.6031
NeuMF	0.2080	0.3407	0.6376
VBCAR	0.2518	0.3873▼(1)	0.6804▼(1)
Triple2Vec	0.3043	0.3607▲(1)	0.6761▲(1)

Model	Dunnhumby Dataset, Recall@10		
	Leave One Item	Leave One Basket	Temporal Global
NMF	0.2498	0.2514▼(3)	0.0908
BPR	0.2514	0.2163▲(1)	0.1018
VAECF	0.2759	0.2371▲(1)	0.1173
NGCF	0.2836	0.2434▲(1)	0.1275
VBCAR	0.3797	0.3342▼(2)	0.1431▼(1)
NeuMF	0.3906	0.3220	0.1410▲(1)
Triple2Vec	0.4391	0.3193▲(2)	0.1454

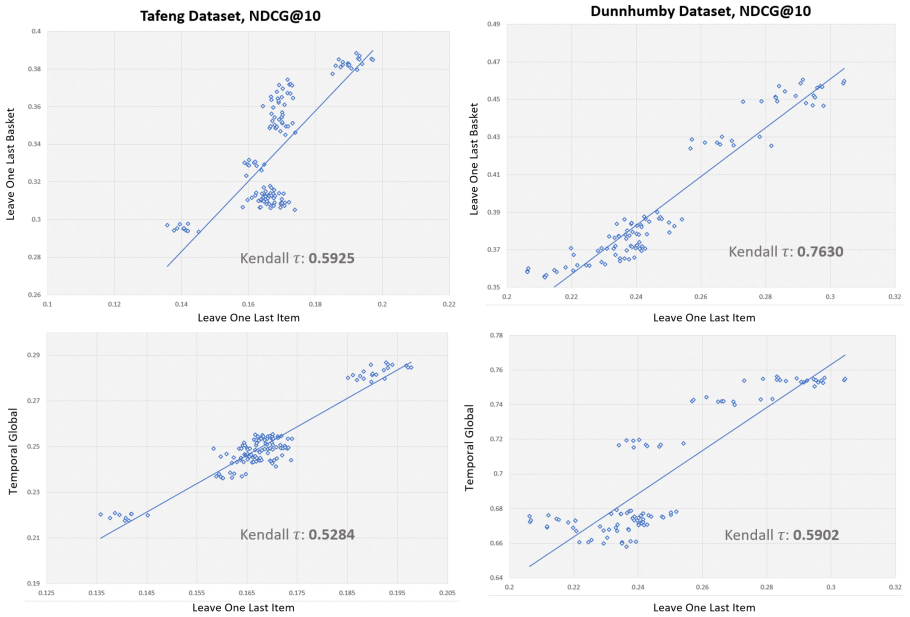

Figure 2: Splitting strategy pair-wise comparison under recommendation NDCG@10 for 230 models.

5 CONCLUSIONS

In this work, we analyzed the impact that different data splitting strategies have on the reported performance of different recommendation models, as the splitting strategies used in the literature vary greatly. Through experimentation using three splitting strategies, seven recommendation models and two datasets, we have shown that the splitting strategy employed is an important confounding variable that can markedly alter the ranking of state-of-the-art systems. This is important, as it highlights that much of the current research being published is not directly comparable, even when the same dataset and metrics are being used. Furthermore, we also have observed that certain splitting strategies favour particular recommendation models - potentially due the different balance of train/validation/test data under each scenario and factors such as whether future evidence is available during training. In terms of best practices for future researchers, we recommend the following:

1) Report the splitting strategy employed: This includes the statistics of the train/validation/test components and any user/item filtering performed, as these can strongly impact performance.
2) Report performance under temporal global splitting: This is generally seen as the most realistic setting, and so should be the default splitting strategy used.
3) Release your data splits: so that they can be re-used by other researchers[5].
4) Use the standardized evaluation tools: to avoid the toolset as a confounding factor, e.g. due to differing implementations or configuration of metrics, such as micro vs. macro averaging. The splitting strategies discussed in this work are all integrated in the BETA-Rec open source project [13][6].

[5]The data splits used in this work can be downloaded from **https://github.com/mengzaiqiao/data_splits**.
[6]https://github.com/beta-team/beta-recsys.

Acknowledgements

The authors acknowledge the BigDataStack project (funded under the European Community's Horizon 2020 research and innovation programme, grant agreement n° 779747).

REFERENCES

[1] Ting Bai, Lixin Zou, Wayne Xin Zhao, Pan Du, Weidong Liu, Jian-Yun Nie, and Ji-Rong Wen. 2019. CTRec: A Long-Short Demands Evolution Model for Continuous-Time Recommendation. In *SIGIR*. 675–684.

[2] Pedro G Campos, Fernando Díez, and Manuel Sánchez-Montañés. 2011. Towards a more realistic evaluation: testing the ability to predict future tastes of matrix factorization-based recommenders. In *RecSys*. 309–312.

[3] Rocío Cañamares, Pablo Castells, and Alistair Moffat. 2020. Offline evaluation options for recommender systems. *Information Retrieval Journal* (2020), 1–24.

[4] 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*. 101–109.

[5] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.

[6] Haoji Hu and Xiangnan He. 2019. Sets2Sets: Learning from Sequential Sets with Neural Networks. In *SIGKDD*. 1491–1499.

[7] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *ICDM*. 197–206.

[8] Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *NeurIPS*. 556–562.

[9] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. In *WSDM*. 322–330.

[10] Dawen Liang, Rahul G Krishnan, Matthew D Hoffman, and Tony Jebara. 2018. Variational autoencoders for collaborative filtering. In *WWW*. 689–698.

[11] Jarana Manotumruksa, Craig Macdonald, and Iadh Ounis. 2018. A Contextual Attention Recurrent Architecture for Context-aware Venue recommendation. In *SIGIR*. 555–564.

[12] Zaiqiao Meng, Richard McCreadie, Craig Macdonald, and Iadh Ounis. 2019. Variational Bayesian Context-aware Representation for Grocery Recommendation. In *CARS2.0@RecSys*.

[13] Zaiqiao Meng, Richard McCreadie, Craig Macdonald, Iadh Ounis, Shangsong Liang, Siwei Liu, Guangtao Zeng, Liang Junha, Yucheng Liang, Qiang Zhang, Xi Wang, and Wu Yaxiong. 2020. BETA-Rec: Build, Evaluate and Tune Automated Recommender Systems. (2020).

[14] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-aware recommender systems. *Comput. Surveys* 51, 4 (2018), 66.

[15] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*. 452–461.

[16] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *WWW*. 811–820.

[17] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. 2020. Neural Collaborative Filtering vs. Matrix Factorization Revisited. *arXiv:2005.09683* (2020).

[18] Steffen Rendle, Li Zhang, and Yehuda Koren. 2019. On the difficulty of evaluating baselines: A study on recommender systems. *arXiv preprint arXiv:1905.01395* (2019).

[19] Noveen Sachdeva, Giuseppe Manco, Ettore Ritacco, and Vikram Pudi. 2019. Sequential Variational Autoencoders for Collaborative Filtering. In *WSDM*. 600–608.

[20] Alan Said and Alejandro Bellogín. 2014. Comparative Recommender System Evaluation: Benchmarking Recommendation Frameworks. In *RecSys*. 129–136.

[21] Guy Shani and Asela Gunawardana. 2011. Evaluating recommendation systems. In *Recommender systems handbook*. Springer, 257–297.

[22] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *CIKM*. 1441–1450.

[23] Qiaoyu Tan, Ninghao Liu, Xing Zhao, Hongxia Yang, Jingren Zhou, and Xia Hu. 2020. Learning to Hash with Graph Neural Networks for Recommender Systems. In *WWW*. 1988–1998.

[24] Mengting Wan, Di Wang, Jie Liu, Paul Bennett, and Julian McAuley. 2018. Representing and Recommending Shopping Baskets with Complementarity, Compatibility and Loyalty. In *CIKM*. 1133–1142.

[25] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *SIGKDD*. 950–958.

[26] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *SIGIR*. 165–174.

[27] Teng Xiao, Shangsong Liang, and Zaiqiao Meng. 2019. Dynamic Collaborative Recurrent Learning. In *CIKM*. 1151–1160.

[28] Hamed Zamani and W Bruce Croft. 2020. Learning a Joint Search and Recommendation Model from User-Item Interactions. In *WSDM*. 717–725.

[29] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *Comput. Surveys* 52, 1 (2019), 1–38.