A guideline-based approach for assisting with the reproducibility of experiments in recommender systems evaluation

Nikolaos Polatidis[*]

School of Computing, Engineering and Mathematics, University of Brighton, Lewes Road Brighton, BN2 4GJ, United Kingdom
N.Polatidis@Brighton.ac.uk

Elias Pimenidis

Department of Computer Science and Creative Technologies, University of the West of England Bristol, BS16 1QY, United Kingdom
Elias.Pimenidis@uwe.ac.uk

Andrew Fish

School of Computing, Engineering and Mathematics, University of Brighton, Lewes Road Brighton, BN2 4GJ, United Kingdom
Andrew.Fish@Brighton.ac.uk

Stelios Kapetanakis

School of Computing, Engineering and Mathematics, University of Brighton, Lewes Road Brighton, BN2 4GJ, United Kingdom
S.Kapetanakis @Brighton.ac.uk

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

Recommender systems' evaluation is usually based on predictive accuracy and information retrieval metrics, with better scores meaning recommendations are of higher quality. However, new algorithms are constantly developed and the comparison of results of algorithms within an evaluation framework is difficult since different settings are used in the design and implementation of experiments. In this paper, we propose a guidelines-based approach that can be followed to reproduce experiments and results within an evaluation framework. We have evaluated our approach using a real dataset, and well-known recommendation algorithms and metrics; to show that it can be difficult to reproduce results if certain settings are missing, thus resulting in more evaluation cycles required to identify the optimal settings.

Keywords: Recommender systems; Evaluation; Reproducibility; Replication.

1. Introduction

Recommender systems are decision support systems found in online web services, mainly in e-Commerce for movies, music, videos or general item recommendation. During the

[*] Corresponding author

last few years, research in recommender systems, both in academia and in industry, has led to numerous publications. The popularity of recommender systems' research has led to the increasingly important problem of reproducibility and replication of experiments during the evaluation of such systems[1,2]. The evaluation of recommendation algorithms is important for measuring the quality of the results and for making objective comparisons of algorithms. A positive aspect found in the literature is the availability of papers that describe in detail their proposed recommendation algorithms, the evaluation methods, the settings and the datasets used[3,4,5]. In the research community, there are different recommendation frameworks that can be used for the evaluation of algorithms. These include Apache Mahout[6], LensKit[7], MyMediaLite[8] and Recommender101[9]. Apache Mahout has been developed by the Apache Foundation, whereas the rest have been developed by researchers in academia. All of these recommendation frameworks provide essentially the same portfolio of algorithms and evaluation metrics. Although, it might be difficult to reproduce a study from research papers the aforemention evaluation libraries provide all the main algorithms and evaluation metrics, while the results for those are very similar and comparable between them[1]. We focus on the reproducibility of new algorithms that will have to be reproduced in the future, despite of the framework that they will be based on. In this paper we define the term reproducibility as a study that is reproducible when the original data and the code are available or can be derived from the explanation steps of the algorithm presented here. Additionally, we define replication as the act of repeating the experiments of a study, leading to the exact or very similar reproducibility of a study.

To assist towards the reproducibility and replication of experimental results in recommender systems we:

- Provide a guideline-based approach that can be used within an evaluation framework when performing experiments, which reduces the number of evaluation cycles necessary to reproduce a result.
- Performed different experiments using a real dataset with the results validating the proposed approach.

The rest of the paper is organized as follows: Section 2 provides the required background, section 3 is the proposed approach, section 4 delivers the experimental evaluation, section 5 contains the discussion and section 6 is the conclusion and future work.

2. Background

In the literature, the quality of recommender systems algorithms tends to be measured using accuracy and classification evaluation methods. The most well-known and often used accuracy methods are the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE), whereas the most well-known classification methods are Precision and Recall. State of the art work about the evaluation of recommender systems can be found in[3,9]. Research papers that propose new recommendation algorithms will typically describe the experimental setup, the dataset and the framework used; the term reproducibility is used to mean the replication and validation of the results by third parties[1,2].

The four main problems that occur when evaluating recommender systems algorithms are non-adherence to the following suggestions[1,2,11]:

1. The framework used for the generation of the recommendations and the evaluation should be mentioned and it should be publicly available.

2. The details of the algorithms should be clearly mentioned, such as the size of the neighbourhood used.

3. The dataset used for the experiments, along with any possible version of it, should be publicly available.

4. Details of how the dataset has been used should be indicated. These must include training and test splits and it should be clear if these have been randomly selected or if parameters have been used to select specific training/testing parts that will make the reproduction easier. Moreover, if k-folds have been used for cross-fold validation then details about the number of folds and how these have been selected should also be available.

The problem of the replication of experiments and the reproducibility of the results has been an open issue, which is acknowledged in the research community with a workshop on the topic organized in 2013[12]. The outcome of the joint community work identifies the key aspects of the reproducibility problem, although its future direction part is limited[13] since the future directions presented are only theoretically-focused, towards the need of general guidelines to produce better results[11,13,14]. One step further, beyond the purely theoretical guidelines, is RiVal[16]; a toolkit that provides four stages in the recommendation process: data splitting, item recommendation, candidate item generation and performance evaluation. RiVal is not an evaluation framework since it pertains to the three different frameworks of Apache Mahout, LensKit and MyMediaLite. The toolkit provides a user interface where the user can input the data splitting, item generation and recommendation and select which framework will be used for the evaluation. A different approach is that due to Košir et al.[15] -- it is explained how statistics can be used to improve 10-fold cross evaluation methods. Furthermore, there are different approaches to the reproducibility problem, such as that byHernández del Olmo and Gaudioso[17]--the authors propose the use of a general framework for recommender systems and evaluation metrics that operates over a set of sessions, but this still makes it difficult to replicate an algorithm if parameters are missing, assuming that an experiment can be replicated. Finally, another available evaluation metric relevant to recommender systems is the modified Reciprocal Hit Rand Metric (mRHR). This has been proposed by Peker and Kocyigit[18], with the use of an alternative hit rank metric being suggested.

3. Proposed approach

A major challenge in recommender systems' evaluation is that when a new recommendation algorithm is developed the concept of reproducibility is not considered in depth, thus making a study difficult to be reproduced in the future by other researchers. In our previous work we have shown that it is difficult to reproduce results using different evaluation libraries due to differences in algorithm and metric implementations, while highlighting that the results derived from well-known algorithms that can be found across libraries can be easily reproduced[1]. In recommender systems' evaluation to reproduce experiments it is recommended to follow a set of guidelines[1,2]. However, most researchers either do not follow them or do not explain in detail the settings of their research environment. This could happen due to strict page requirements of publication outputs or due to inexperienced new researchers who might not be aware that omitted settings are important since they could lead to large differences in evaluation results. Furthermore, the guidelines proposed or adopted may vary between researchers and the need for their standardization in a form of framework seems necessary. Thus, we propose the use of a specific set of guidelines, as described in section 3.1. In section 3.2 it is explained how these guidelines can be used within the same evaluation library to reproduce results.

3.1. *Guidelines*

The following elements are the ones responsible for the reproducibility of results across libraries and within the same library.

Source code. The most important guideline towards the reproducibility of evaluation results and the replication of an algorithm is the unrestricted availability of the source code in a public domain online repository; guaranteed that it will be available for free retrieval for a long period of time. If the code is not available then numerous settings including the steps of algorithm, evaluation and dataset settings need to be explained in detail.

Library. As it might be difficult to reproduce and replicate results using different libraries due to variations in settings and versions it is sensible to use the same library and version that the researchers who developed the particular algorithm have used.

Main recommendation algorithm. The steps of the algorithms should be described in detail.

Evaluation settings. Some frameworks provide different settings whereas others do not. For example, differences between Recommender101 and Apache Mahout include:

- Recommender101 provides options about the minimum number of ratings per user or per item whereas in Apache Mahout someone will have to manually edit the source code to do that. This parameter could lead to different results.

- Recommender101 provides settings for both the minimum and maximum rating value that should be taken into consideration during the evaluation, whereas Apache Mahout does not.

Hence, it is important for each framework to provide the same settings during the evaluation process. We believe that a framework such as Recommender101 that provides more options is more suitable for research and standardization development. A standard that defines all possible parameters for recommender system evaluation is necessary for reproducibility.

Dataset settings. This is related to the evaluation settings and is necessary for users to follow them to make reproducibility easier. The standard should provide guidelines related to the percentage of training and test set. It should be clearly mentioned which part of the dataset has been used for training and which for testing and while pure random selection makes results more reliable it affects the reproducibility. Furthermore, if k-folds have been used for cross-fold evaluation then details about what the folds are should be available. The dataset used should also be explained in detail, including any possible version of it, its size and any other available parameters.

3.2. *Replication*

In section 3.1, we discussed how the architecture, the algorithm implementation, the evaluation settings and the dataset settings can be accountable for different results. Thus, the main problem resides in reproducing the results of a study based on the proposed algorithm, the evaluation settings, the data used and the library that the researchers have used to run the experiments. Therefore, it is important to follow a set of guidelines, or good practices, to make the results reproducible. We have identified the following steps that, can assist researchers in reproducing results.

1. Explanation regarding the library and version used. If the library has been self-developed it should be available online.

2. Step by step explanation of the evaluation settings should be provided. These may include: the number of user neighbours used, if ratings below a threshold have been removed, if users that have not rated a certain number of items have been removed, if items with few ratings or too many ratings have been removed, and if a potion of the user pool that has been usedor if all the users in the dataset have been used. Furthermore, any other settings, such as if there is a threshold for forming the user neighbourhood or a threshold of common rated items, should also be mentioned. Finally, it should be noted that some of the settings available in a library might not be available in another and it would be useful if every paper produced highlighted this. All available settings of the library used should be mentioned and it should be noted that different algorithms might use different settings and parameters, thus resulting in a non-exhaustive list.

3. Regarding the dataset, it should be clear which dataset and version has been used, the dataset split using test/train identified, and/or whether a cross-fold validation has been used and how many folds have been used. In addition, it should be made clear which exact part has been used for training, which for testing, and how the selection has been made to enable use of the same training/testing parts in the reproduction of the experiments.

4. Explanation of the settings of the algorithms that have been used for the

comparison. Typically, researchers compare their proposed algorithm to a number of traditional and state of the art algorithms to show that their algorithm is better in terms of accuracy or quality of the top-N recommendations, but the details of these algorithms should be available to avoid confusion.

5. The optimization of parameters should be explained in detail, including the exact parameters used for the proposed algorithm and the algorithms used for comparisons. For example, if optimized settings are used for one algorithm and not for the other then this can make a difference in the output results and if the settings are not specified then only assumptions about them can be made.

3.3. *Proposed approach application*

In Recommender101 library all existing recommendation algorithms extend the *AbstractRecommender* class. Moreover, all future algorithms that are based on Recommender101 are required to extend the same class. Thus, the aforementioned abstract class has been extended to enforce the use of guidelines.

The *AbstractRecommender* class has been extended to include the following 4 methods (functions):

1. libraryDetails()
2. evaluationSettings()
3. datasetInformation()
4. algorithmDetails()

All of the aforementioned methods are implemented, non-abtract methods that their functionality is to remind the user that certain information regarding their algorithm, settings and library used should be included. Each of the methods opens a predefined empy text file with a relevant name and if the file is empty when the evaluation starts the user is reminded that the respective details should be included in the text file. Algorithms 1, 2, 3 and 4 explain each of the above method follows.

Algorithm 1: libraryDetails

```
START EXECUTION
Step 1: Evaluation starts
Step 2: libraryDetails is called

Open 'librarydetails.txt'
If file missing
        Display message: Please create the 'librarydetails.txt' file and add the relevant
        library details.
Else if:  File there but empty
        Display message: Please add more than one line of text at the 'librarydetails.txt'
        regarding the details of the library used.
Else
        Proceed: Next step of the evaluation cycle
```

```
END OF EXECUTION
```

Algorithm 2: evaluationSettings

```
START EXECUTION
Step 1: Evaluation continues
Step 2: evaluationSettings is called

Open 'evaluationsettings.txt'
If file missing
        Display message: Please create the 'evaluationsettings.txt' file and add the
        relevant evaluation settings information.
Else if: File there but empty
        Display message: Please add more than one line of text at the
        'evaluationsettings.txt' regarding the settings of the evaluation.
Else
        Proceed: Next step of the evaluation cycle
END OF EXECUTION
```

Algorithm 3: datasetInformation

```
START EXECUTION
Step 1: Evaluation continues
Step 2: datasetInformation is called

Open 'datasetinformation.txt'
If file missing
        Display message: Please create the 'datasetinformation.txt' file and add the
        relevant dataset details.
Else if: File there but empty
        Display message: Please add more than one line of text at the
        'datasetinformation.txt' regarding the details of the dataset.
Else
        Proceed: Next step of the evaluation cycle
END OF EXECUTION
```

Algorithm 4: algorithmDetails

```
START EXECUTION
Step 1: Evaluation continues
Step 2: algorithmDetails is called

Open 'algorithmdetails.txt'
If file missing
        Display message: Please create the 'algorithmdetails.txt' file and add the
        relevant algorithmic details.
```

```
Else if: File there but empty
        Display message: Please add more than one line of text at the
        'algorithmdetails.txt' regarding the algorithmic details.
Else
        Proceed: Next step of the evaluation cycle
END OF EXECUTION
```

At the end of the evaluation all four text files are combined in one final text file called finalResult which included the evaluation outcome at the end and a message is displayed to the user that all details and results are now saved at the finalResult text file.

4. Experimental Evaluation

In this section, we present the experimental evaluation. The scope of the evaluation is to show that if certain parameters or certain settings are omitted then a number of evaluation cycles is necessary until an optimum level is reached. Therefore, scope of the evaluation section is to:

1. Show that simple baseline algorithms produce similar results despite of changes in the few settings required for those. This is shown in subsection 4.4.
2. Show that in more advanced algorithms, such as the BPR method, if certain settings are not mentioned then a number of evaluation cycles is necessary until the optimum is result is obtained. This is shown in subsection 4.5.

4.1. *Recommendation methods*

For the experimental evaluation, two recommendation methods have been used, CF and multi-level CF:

- **CF.** User-related CF, based on PCC, is defined in Eq. (1). In PCC the sum of ratings between two users is compared. Sim (*a, b*) is the similarity between users *a* and *b*, whilst $r_{a,p}$ is the rating of user a for product *p*, $r_{b,p}$ is the rating of user *b* for product *p*, and r'_a and r'_b represent the users' average ratings. *P* is the set of all products. Moreover, the similarity value ranges from -1 to 1 and a higher value is better (i.e. a higher value indicates more similar).

- **Multi-level CF**. This method is defined in Eq. (2)[5]. It relies on the similarity value returned from PCC and the number of co-rated items. In this method, T stands for the total number of co-rated items, multiple accuracy levels are being used with t1, t2, t3 and t4 being fixed values for the number of co-rated items, Ia and Ib being the items rated by users a and b respectively, x being a positive value added to the PCC value if the condition holds and y is a positive number that PCC should be larger than or equal for the condition to hold. More specifically, four levels are used and in each level the number of co-rated items should be between two fixed t values and the PCC value above or equal to a fixed value *y* and if the condition holds then a fixed number *x* is added to PCC, thus resulting in different user neighborhoods. This is true for the first four levels, whereas at the last level the similarity is fixed to zero.

$$\underset{a,b}{PCC} = \frac{\sum p \in P(r_{a,p} - \acute{r}_a)(r_{b,p} - \acute{r}_b)}{\sqrt{\sum p \in P(r_{a,p} - \acute{r}_a)^2}\sqrt{\sum p \in P(r_{b,p} - \acute{r}_b)^2}} \quad (1)$$

$$\underset{a,b}{Sim}^{Proposed} = \begin{cases} \underset{a,b}{Sim}^{PCC} + x, & if \ \frac{|Ia \cap Ib|}{T} \geq t1 \ and \ \underset{a,b}{Sim}^{PCC} \geq y \\ \underset{a,b}{Sim}^{PCC} + x, & if \ \frac{|Ia \cap Ib|}{T} < t1 \ and \ \frac{|Ia \cap Ib|}{T} \geq t2 \ and \ \underset{a,b}{Sim}^{PCC} \geq y \\ \underset{a,b}{Sim}^{PCC} + x, & if \ \frac{|Ia \cap Ib|}{T} < t2 \ and \ \frac{|Ia \cap Ib|}{T} \geq t3 \ and \ \underset{a,b}{Sim}^{PCC} \geq y \\ \underset{a,b}{Sim}^{PCC} + x, & if \ \frac{|Ia \cap Ib|}{T} < t3 \ and \ \frac{|Ia \cap Ib|}{T} \geq t4 \ and \ \underset{a,b}{Sim}^{PCC} \geq y \\ 0, & otherwise \end{cases}$$

(2)

4.2. *Settings*

The experimental evaluation took place using an Intel i7 with 8GBs of RAM running Windows 10 and it was based on the Recommender101 library version 0.61.

- **Dataset**. The dataset used is the MovieLens 1 million[19], which contains 3952 movies, 6040 users and 1,000,209 ratings, with each user having at least 20 ratings and a 5 cross-fold approach has been used for evaluating.

- **CF**. The traditional CF recommendation method, based on PCC, has been used, as defined in Eq. (1).

- **Multi-level CF**. For this method, x is a pre-defined value that is added to the similarity value returned from PCC and has been set 0.50 for the first(top) level, 0.375 for the second, 0.25 for the third and 0.125 for the last level, while y is the similarity value returned from PCC and has been set to be greater than or equal to 0.33. The values for t1, t2, t3 and t4 variables have been set to be t1=50, t2=20, t3=10, and t4=5. The number of co-rated items for level 1 must be equal or greater than 50, level 2 is between 49 and 20, level 3 is between 19 and 10 and level 4 is between 9 and 5. Finally, from Eq. (2) it is clear that a fifth level exists, and its value is always zero.

4.3. *Evaluation metrics*

For measuring the prediction accuracy, we have used MAE and RMSE, which are defined in Eq. (3) and Eq. (4) respectively. In both measures p_i is the predicted rating and r_i is the actual rating in the summation. MAE is based on absolute actual and predicted values and RMSE is based on squared actual and predicted values, thus RMSE punishes larger gaps between actual and predicted ratings, resulting in larger differences as compared to MAE. Both methods are used for the computation of the deviation between the predicted ratings and the actual ratings. It should be noted that lower values are considered to be better. Both MAE and RMSE have been widely used in previous research for predicting the accuracy of recommender systems[4,20,21].

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|p_i - r_i| \tag{3}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - r_i)} \tag{4}$$

For measuring the quality of the top-N recommendations we have used Precision and Recall, which are defined in Eq. (5) and Eq. (6)[4,10]. Precision is the amount of relevant recommendations found in the retrieved set of recommendations and Recall is the amount of relevant recommendations that have been retrieved successfully.

$$precision = \frac{Correctly\ recommended\ items}{Total\ recommender\ items} \tag{5}$$

$$recall = \frac{Correctly\ recommended\ items}{Relevant\ items} \tag{6}$$

4.4. *Baseline experimental results*

The following tables describe the experimental results. Table 1 presents MAE, RMSE, Precision and Recall results for the CF method based on the MovieLens dataset. Table 2 presents MAE, RMSE, Precision and Recall results for the multi-level CF method based on the MovieLens dataset. K is the number of nearest neighbors and the number of top-N recommendations requested is 10 for both tables.

Table 1. CF results based on MovieLens 1m

Metric	**K=20**	**K=50**
MAE	0.845	0.815
RMSE	1.097	1.052
Precision	0.652	0.655
Recall	0.612	0.626

Table 2. Multi-level CF results based on MovieLens 1m

Metric	**K=20**	**K=50**
MAE	0.737	0.715
RMSE	0.965	0.927
Precision	0.675	0.681
Recall	0.349	0.459

4.5. *Advanced recommendation method exaperimental results*

CF and multi-level CF are both based on their nearest neighbor ratings to make recommendations. Thus, in such recommendation methods the larger the user neighborhood used for the recommendations, the better the recommendations are, both in terms of prediction accuracy (MAE, RMSE) and in the quality of the list of the top-N recommendations (Precision and Recall). An important problem is choosing the parameters required for a recommendation method to work with high utility. For CF, being a simple nearest neighbor-based algorithm, the only parameter that can be changed within it is the number of nearest neighbors. However, multi-level extends CF by introducing (a) a fixed number of levels, (b) a number of maximum and minimum co-rated items between levels and (c) and fixed positive value added to the PCC value of each level, while the last level is set to zero. The problem of reproducibility arises when the source code is not available or when parameters are not explained in detail within research papers, resulting in a number of evaluation cycles necessary to identify the optimal values for parameters with a possible loss in time that depends on the number of evaluation cycles.

Case study: The BPR algorithm

BPR is a Bayesian based algorithm for personalized ranking of items and is explained in detail by Rendle et al.[22]. BPR has numerous settings and the implementation used is the one found in the Recommender101 library. The available settings of the algorithm are highlighted below.

```
initialSteps=100|numFeatures=100|uniformSampling=
true|learnRate=0.05|biasReg=0|regI=0.0025|regJ=0.
        00025|regU=0.0025|updateJ=true,\
```

Evaluation cycles:
Considering that one variable is missing from the algorithm explanation of a research paper or that the variable is there, but that its settings are unavailable. If the settings are unavailable, then the number of evaluation cycles that need to take place until the optimum settings can be identified. For this example, the value of the learnRate variable was withheld and 7 evaluation cycles were necessary. Table 3 presents the results of the BPR algorithm with the default values as shown above and since this is an item ranking algorithm only information retrieval metrics such as Precision, Recall or similar ranking metrics can be measured.

Table 3. Evaluation results with default values

Metric	**Results**
Precision	0.628
Recall	0.611

Table 4 presents the results of the first evaluation cycle with the learnRate value set to 0.50.

Table 4. Evaluation cycle 1

Metric	**Results**
Precision	0
Recall	0

Table 5 presents the results of the second evaluation cycle with the learnRate value set to 0.40.

Table 5. Evaluation cycle 2

Metric	**Results**
Precision	0
Recall	0

Table 6 presents the results of the third evaluation cycle with the learnRate value set to 0.30.

Table 6. Evaluation cycle 3

Metric	**Results**
Precision	0
Recall	0

Table 7 presents the results of the fourth evaluation cycle with the learnRate value set to 0.25.

Table 7. Evaluation cycle 4

Metric	**Results**
Precision	0.584
Recall	0.594

Table 8 presents the results of the fifth evaluation cycle with the learnRate value set to 0.15.

Table 8. Evaluation cycle 5

Metric	**Results**
Precision	0.606
Recall	0.599

Table 9 presents the results of the sixth evaluation cycle with the learnRate value set to 0.10.

Table 9. Evaluation cycle 6

Metric	**Results**
Precision	0.622
Recall	0.609

Table 10 presents the results of the first evaluation cycle with the learnRate value set to 0.05.

Table 10. Evaluation cycle 7

Metric	**Results**
Precision	0.628
Recall	0.611

After seven evaluation cycles the optimum settings as shown in table 6 have been reached. However, there are large differences in Precision which clearly affects the list of the top-N recommendations of the users of a recommendation system. In tables 4, 5 and 6 the values returned are zero in all cases. As the learnRate value decreases there is a 7.5% difference in Precision between the default settings used for the results in table 3 and the results of table 7. However, the difference is decreasing as the learnRate value decreases in tables 8 and 9 and in table 10 the optimal settings have been found.

5. Discussion

Reproducibility of experimental results in recommender systems is of high importance to promote algorithm applicability, extended evaluation and research method cross pollination. At the moment, there is a reported tendency not to explain experimental parameters in baselines, leading to uncertainty in method comparison, reproducibility of results and cross-research conclusions. We believe that increasing the uniformity of the experimental settings, metrics and datasets can lead to more robust research, which will both benefit and accelerate research in the recommender system community. To achieve reproducibility with current technologies researchers should explain in detail all of the implementation and parameter details of their proposed algorithm and the parameters of alternative methods that have been used for comparison. Thus, by delivering all the aforementioned details the reproducibility of results will become easier and the number of evaluation cycles will be reduced. This will assist other researchers to have a more rapid understanding about a proposed method, since a researcher might already be aware about the other methods used in the evaluation as comparison baselines. In this paper we have shown that we there can be significant differences in terms of precision even with only one setting being unknown, which results in requiring a number of evaluation cycles until an optimum setting can be identified. A difference in precision, in terms of percentages, as shown in the evaluation section can easily reach a high value of 7.5% which will highly affect the list of recommendations of most users of a system that utilizes a recommender for product or service recommendation, therefore resulting in different experience for the user and possible loss for a vendor. However, assuming that vendors can avoid that by using recommendation algorithms that are accurate enough for their purposes, this leads to the problem of reproducibility of experiments in recommender systems evaluation. Researchers will have to deal with gaps found in settings of proposed recommendation algorithms and baselines used for comparison purposes in papers. In section 4.5 we have shown that with only one uknown variable for one algorithm seven evaluation cycles were necessary until the optimal setting was reached. However, in a typical scenario, a number of algorithms need to be evaluated and one or more parameters is missing from each algorithm. This will lead to many evaluation cycles taking place for the evaluation of each algorithm, which means that valuable time will have to be spent for optimizing alternative recommendation algorithms, whereas this time could have been devoted for algorithm development instead.

6. Conclusions and future work

In this paper we have shown the difficulty of reproducing evaluation results in recommender systems. Various settings and algorithm implementations lead to producing different results. For example, the selection of which data from datasets will be used for training and which for testing leads to different results, which in turn has an impact on the overall conclusion. We showed that by employing a set of guidelines that can facilitate a common reference baseline for recommendation experiments. This work is the first step towards an extensively broad validation framework for recommender systems and it aims to educate the community while collating feedback towards robust experimentation and evaluation. The results show that if parameters are omitted when evaluating algorithms then more evaluation cycles are necessary to find the optimal settings. This could possibly lead to spending significantly more time to identify settings for algorithms and making comparisons between algorithms. Apart from the guidelines, an extensive experimental evaluation section has been delivered using a well-known evaluation library, algorithms and metrics support the idea that in the future a complete framework that will support researchers in reproducing experiments should be developed, used and possibly standardized.

References

1. Polatidis N, Kapetanakis S, Pimenidis E, Kosmidis K. Reproducibility of experiments in recommender systems evaluation. InIFIP International Conference on Artificial Intelligence Applications and Innovations 2018 May 25 (pp. 401-409). Springer, Cham.
2. Said, A., Bellogín, A.: Comparative recommender system evaluation. Proc. 8th ACM Conf. Recomm. Syst. RecSys '14. 129–136 (2014).
3. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender systems: An introduction. (2010).
4. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. 22, 5–53 (2004).
5. Polatidis, N., Georgiadis, C.K.: A multi-level collaborative filtering method that improves recommendations. Expert Syst. Appl. 48, 100–110 (2016).
6. Owen, S., Anil, R., Dunning, T., Friedman, E.: Mahout in Action. (2011).
7. Ekstrand, M.D., Ludwig, M., Konstan, J.A., Riedl, J.T.: Rethinking the Recommender Research Ecosystem: Reproducibility, Openness, and LensKit. In: Proceedings of the Fifth ACM Conference on Recommender Systems. pp. 133–140. ACM, New York, NY, USA (2011).
8. Gantner, Z., Rendle, S.: MyMediaLite: A free recommender system library. Proc. fifth ACM Conf. Recomm. Syst. 305–308 (2011).
9. Jannach, D., Lerche, L., Gedikli, F., Bonnin, G.: What recommenders recommend-an analysis of accuracy, popularity, and sales diversity effects. In: User Modeling, Adaptation, and Personalization. pp. 1–13 (2013).
10. Shani, G., Gunawardana, A.: Evaluating recommendation systems. Recomm. Syst. Handb. 257–298 (2011).
11. Konstan, J.A., Adomavicius, G.: Toward Identification and Adoption of Best Practices in Algorithmic Recommender Systems Research. In: Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation. pp. 23–28. ACM, New York, NY, USA (2013).
12. Bellogin, A., Castells, P., Said, A., Tikk, D.: Workshop on reproducibility and replication in recommender systems evaluation. In: Proceedings of the 7th ACM conference on Recommender systems - RecSys '13. pp. 485–486 (2013).
13. Bellogin, A., Castells, P., Said, A., Tikk, D.: Report on the Workshop on Reproducibility and Replication in Recommender Systems Evaluation (RepSys). SIGIR Forum. 48, 29–35 (2014).
14. Beel, J., Breitinger, C., Langer, S., Lommatzsch, A., Gipp, B.: Towards reproducibility in recommender systems research. User Model. User-adapt. Interact. 26, 69–101 (2016).
15. Košir, A., Odić, A., Tkalčič, M.: How to improve the statistical power of the 10-fold cross validation scheme in recommender systems. In: RecSys RepSys 2013: Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation. pp. 3–6 (2013).
16. Said, A., Bellogín, A.: RiVal – A Toolkit to Foster Reproducibility in Recommender System Evaluation. RecSys 2014 Proc. 8th ACM Conf. Recomm. Syst. 371–372 (2014).

17. Hernández del Olmo, F., Gaudioso, E.: Evaluation of recommender systems: A new approach. Expert Syst. Appl. 35, 790–804 (2008).
18. Peker, S., Kocyigit, A.: mRHR: A Modified Reciprocal Hit Rank Metric for Ranking Evaluation of Multiple Preferences in Top-N Recommender Systems. In: Dichev, C. and Agre, G. (eds.) Artificial Intelligence: Methodology, Systems, and Applications: 17th International Conference, AIMSA 2016, Varna, Bulgaria, September 7-10, 2016, Proceedings. pp. 320–329. Springer International Publishing, Cham (2016).
19. Harper, F.M., Konstan, J.A.: The MovieLens Datasets. ACM Trans. Interact. Intell. Syst. 5, 19 (2015).
20. Polatidis, N., Georgiadis, C.K., Pimenidis, E., Mouratidis, H.: Privacy- preserving collaborative recommendations based on random perturbations. Expert Syst. Appl. 71, (2017).
21. Alshammari, G., Jorro Aragoneses, J. L., Kapetanakis, S., Petridis, M., Recio- Garcia, J. A., Diaz-Agoudo, B.: A hybrid CBR approach for the long tail problem in recommender systems. In proceedings of The International Conference in Case-based Reasoning (ICCBR 2017), pp. 35-45 (2017)
22. Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009, June). BPR: Bayesian personalized ranking from implicit feedback. In Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence (pp. 452-461). AUAI Press.