Reproducibility of experiments in recommender systems evaluation

Nikolaos Polatidis[1], Stelios Kapetanakis[1,2], Elias Pimenidis[3] and Konstantinos Kosmidis[4]

[1]School of Computing, Engineering, and Mathematics, University of Brighton, BN2 4GJ, Brighton, United Kingdom
N.Polatidis@Brighton.ac.uk
S.Kapetanakis@Brighton.ac.uk
[2]Gluru Research, Gluru, WC2B 4HN, London, United Kingdom
stelios@gluru.co
[3]Department of Computer Science and Creative Technologies, University of the West of England, BS16 1QY, Bristol, United Kingdom
Elias.Pimenidis@uwe.ac.uk
[4]School of Computing and Engineering, University of West London, W5 5RF, London, United Kingdom
Konstantinos.Kosmidis@uwl.ac.uk

Abstract. Recommender systems evaluation is usually based on predictive accuracy metrics with better scores meaning recommendations of higher quality. However, the comparison of results is becoming increasingly difficult, since there are different recommendation frameworks and different settings in the design and implementation of the experiments. Furthermore, there might be minor differences on algorithm implementation among the different frameworks. In this paper, we compare well known recommendation algorithms, using the same dataset, metrics and overall settings, the results of which point to result differences across frameworks with the exact same settings. Hence, we propose the use of standards that should be followed as guidelines to ensure the replication of experiments and the reproducibility of the results.

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 last few years, research in recommender systems both in academia and in industry counts numerous publications found in the literature [1]. The popularity in recommender systems research has led to the increasingly important problem of reproducibility and replication of experiments during the evaluation of such systems. The valuation of recommendation algorithms is important for measuring the quality of the results and make objective comparisons among 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 datasets used [2–4].

In the research community, there are different recommendation frameworks that can be used for the evaluation of algorithms. These include, among others, the Apache Mahout [5], LensKit [6], MyMediaLite [7] and Recommender101 [8]. The first one has been developed by the Apache Foundation whereas the rest have been developed by researchers in academia. All these recommendation frameworks provide essentially the same portfolio of algorithms. However, substantial differences exist in the implementation of the algorithms, data management and evaluation methods and while all frameworks provide the same basic evaluation methods, differences in algorithm implementation makes it difficult to compare results across frameworks [1].

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

- Provide a set of standards and best practices that can be used when performing experiments.
- Performed different experiments using a real dataset and different recommendation libraries with the results validating our approach.

The rest of the paper is organized as follows: Section 2 provides the required background, section 3 is a comparison between recommendation libraries, real data and different settings, section 4 delivers the proposed approach and section 5 contains the conclusion and future work parts.

2 Background

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

The four main problems that occur when evaluating recommender systems algorithms are [1, 10]:

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 neighborhood used.

3. The dataset used for the experiments, along with any possible version of it and it should be publicly available.
4. Details of how the dataset has been used. These must include training and test splits and if these have been randomly selected or if parameters have been used to select specific 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 be available.

The problem of the replication of experiments and the reproducibility of the results has been an open issue in the research community with a workshop organized in 2013 [11]. The outcome of the joint community work identifies the key aspects of the reproducibility problem, although its future direction part is limited [12] since future directions are only theoretical towards the need of general guidelines to produce better results [10, 12–14]. One step further from the theoretical guidelines is RiVal [15], 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 three different frameworks, these of Apache Mahout, LensKit and MyMediaLite. The toolkit provides a user interface where the user will input the data splitting, item generation and recommendation and will select which framework will be used for the evaluation. Furthermore, there are different approaches to the problem with one found in [16], where the authors propose the use of a general framework for recommender systems and evaluation metric that operates over a set of sessions. Finally, another related metric available is the modified Reciprocal Hit Rand Metric (mRHR) proposed by [17], where the use of an alternative hit rank metric is proposed. However, in this work the problem of the average reciprocal hit rank (ARHR) is only tackled in the context of the evaluation of recommender systems.

3 Comparing experimental results using different libraries

In this section we present a comparison of two recommendation libraries. Apache Mahout and Recommender 101 based on the Pearson Correlation Similarity (PCC) and Jaccard similarity measurements and the Mean Absolute Error (MAE) error rating prediction metric.

3.1 Settings

The experimental evaluation took place on an Intel i7 with 8GBs of RAM running Windows 10. The dataset used is the MovieLens 1 million [18], which contains 3952 movies, 6040 users and 1,000,209 ratings, with each user having at least 20 ratings. The dataset has been split in 80% for training and 20% for testing and the percentages have been randomly selected once for each library.

3.2 **Recommendation methods**

For the experiments we have used PCC and Jaccard. PCC is defined in equation 1. In PCC the sum of ratings between two users is compared. Sim (a, b) is the similarity between users a and b, also $r_{a,p}$ 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 user's average ratings. P is the set of all products. Moreover, in PCC the similarity value between users ranges from -1 to 1 and higher values represent a closer similarity between users. Additionally, Jaccard similarity which is defined in equation 2 and in this approach only the number of co-rated items is taken into consideration. In Jaccard a represents a user and b a second user. Then, Jaccard provides a similarity value between -1 and 1 by measuring the co-rated items and dividing the size of the intersection by the size of the union of the sets. Once again higher values represent better similarity.

$$PCC\ a,b\ = \frac{\sum p \in P (ra,p - r'a)(rb,p - r'b)}{\sqrt{\sum p \in P (ra,p - r'a)^2} \sqrt{\sum p \in P (rb,p - r'b)^2}} (1)$$

$$Jaccard\ a,b\ = \frac{|IaIb|}{|IaIb|} (2)$$

3.3 **Accuracy measure**

For measuring the prediction accuracy, we have used MAE which is defined in equation 3 where *pi* is the predicted rating and *ri* is the actual rating in the summation. This method is used for the computation of the deviation between the predicted ratings and the actual ratings. It should also be noted that lower values are better. MAE has been widely used in previous research for predicting the accuracy of recommender systems [3, 19, 20].

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |pi - ri| \ (3)$$

3.4 **Results**

The following tables describe the experimental results. Table 1 presents the results using the Apache Mahout library and table 2 the results using the Recommender101 library. In both tables the values represent the rating prediction errors based on the MAE metric. It is shown in the tables that different values have been derived after using the same settings and evaluation metric for two different recommendation libraries.

Table 1**.** MAE results for Apache Mahout

	Number of k nearest neighbours used for evaluating MAE					
	60	80	100	200	300	400
PCC	0.843	0.835	0.827	0.802	0.789	0.785
Jaccard	0.798	0.802	0.775	0.790	0.799	0.786

Table 2. MAE results for Recommender101

	Number of k nearest neighbours used for evaluating MAE					
	60	80	100	200	300	400
PCC	0.870	0.862	0.841	0.811	0.785	0.761
Jaccard	0.724	0.718	0.717	0.715	0.710	0.715

4 **Proposed approach**

In recommender systems evaluation to reproduce experiments it is recommended to follow a set of guidelines [1]. However, most researchers either do not follow them or do not explain in detail the settings of their research environment. Furthermore, the guidelines may vary across researchers and the need for their standardization in a form of framework is necessary. A major challenge in recommender system evaluation is that there are many different libraries for evaluating algorithms and the possibility of having one single library or making all the current libraries following a universal or standardized approach is rather impossible. Furthermore, scientists might decide they want to develop their own library for performing the evaluations. Thus, we list a set of guidelines in 4.1 that explain the differences in evaluation libraries and why the results are different and in 4.2 we propose how to replicate studies.

4.1 Guidelines

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

Architecture. By this we mean the architecture of the recommendation library. For example, Apache Mahout has a different architecture when compared to Recommender 101. In the first library the *PearsonCorrelationSimilarity* class extends an *AbstractSimilarity* class but it implements Pearson, whereas in the second library there is a *NearestNeighbors* class implementing *AbstractRecommender,* which includes different neighborhood-based implementations including both Pearson and Cosine.

Main recommendation algorithm. The most widely used method for providing recommendations in e-Commerce and other online environments is Collaborative Filtering (CF) [4]. In CF, a user neighborhood is created based on previous common history between users based on a similarity function such as Pearson Correlation Coefficient (PCC) or Cosine similarity. However, small alterations on how CF works exist between frameworks.

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 affects the reproducibility. Furthermore,

if k-folds have been used for cross-fold evaluation then details about which the folds are should be available. Furthermore, the dataset used should be explained in detail, including any possible version of it, its size and any other available parameters.

4.2 **Replication**

In 4.1 we discussed how the architecture, the algorithm implementation, the evaluation settings and the dataset settings are accountable for different results among different libraries or within the same library. However, the architecture of a library would be very difficult to change as well as the implementation of already established algorithms contained in libraries. Moreover, it would be dreadful to have different libraries behave the same. Thus, the main problem resides in reproducing the results of a study based on a proposed algorithm, the evaluation settings, the data used and the library 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 if followed, will assist people reproducing the results using the same framework.

1. Explanation regarding the library and version you have used. If the library has been self-developed it should be available online.
2. Step by step explanation of the evaluation settings such as the number of user neighbors 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 from the user base satisfying the criteria a sample of the users have been used or if all the users in the dataset have been used. Furthermore, other settings such as if there is a threshold for forming the user neighborhood or a threshold of common rated items should be mentioned. Finally, it should be noted that some of the settings available in a library might not be in another. All available settings of the library used should be mentioned.
3. In the dataset it should be made clear which dataset and version has been used, if the dataset has been split using test/train 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 use the same training/testing parts in the reproduction of the experiments.
4. If the proposed algorithm extends a class of the library used or if it is a standalone file using the library for evaluation purposes.

Table 3 presents the results of replicating the tests that are presented in table 2. The first two rows represent the first test where the exact same evaluation settings and data have been used as in table 2. The two rows further down represent the second test run with a different random selection of 80% for training and 20% for testing. Finally,

table 4 contains results based on a 5-fold cross-fold validation and not on an 80-20 training/testing scale, which makes a difference because it produces the average of different MAE evaluations and this should be specified in the settings. In table 4 Recommender101 and PCC have been used with 60 neighbors and with a different minimum number of ratings per user. Moreover, in table 3 the 1st and 2nd tests of each method show that when the exact same settings are used but the training and testing parts of the dataset are randomly selected then there are differences in the output values and in table 4 we show that if the number of minimum considered ratings that a user has submitted to be taken into consideration for the evaluation process is different then different results are derived.

Table 3. Reproducing the MAE results based on Recommender101

	Number of k nearest neighbours used for evaluating MAE					
	60	80	100	200	300	400
PCC (1st)	0.870	0.862	0.841	0.811	0.785	0.761
Jaccard (1st)	0.724	0.718	0.717	0.715	0.710	0.715
PCC (2nd)	0.868	0.860	0.841	0.810	0.784	0.760
Jaccard (2nd)	0.723	0.716	0.716	0.713	0.709	0.710

Table 4. 5-fold cross-validation with different settings MAE results based on Recommender101

Method used	Min number of ratings per user (30)	Min number of ratings per user (Not known and not specified – Default value used by the library)
PCC	0.872	0.890

5 Conclusions and future work

In this paper we have performed a comparative analysis and shown that it is difficult to reproduce evaluation results both across different libraries but also when the same library is used. The use of 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 to the overall conclusion. Furthermore, we conclude that there are many different parameters that need to be considered by researchers when performing evaluation that it is very difficult to achieve a complete reproduction. Thus, we proposed a unified approach which can facilitate a common reference baseline for recommendation experiments across different frameworks and a set of guidelines to tackle a cross-industry challenge. 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 comparison across recommender frameworks. The results show that when comparing results using the same settings with different libraries the output is not the same and within the same library small changes affect the output. This paper proposed a set of guidelines that can be followed to solve the problem. Furthermore, the results have been validated using two different libraries and real data.

In the future we aim to work towards the following research directions:
Tests for information retrieval metrics such as Precision and Recall. Further validation will be necessary based on information retrieval metrics to examine the behavior of the libraries.
A framework for the reproduction of experiments. A complete framework that will support researchers in the direction of reproducing experiments should be developed, used and possibly standardized.
A universal metric. The development of a universal metric that will make results comparable across libraries is essential.
Reproducibility in user centric studies. It is essential in the domain of recommender systems to have a set of guidelines in the form of a framework that will assist towards the direction of reproducing results in user centric studies. However, it will be difficult to reproduce studies when humans are involved, since it will be difficult to have the exact number of people with the same background and maintain a similar behavior.

References

1. Said, A., Bellogín, A.: Comparative recommender system evaluation. Proc. 8th ACM Conf. Recomm. Syst. - RecSys ’14. 129–136 (2014).
2. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender systems: An introduction. (2010).
3. 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).
4. Polatidis, N., Georgiadis, C.K.: A multi-level collaborative filtering method that improves recommendations. Expert Syst. Appl. 48, 100–110 (2016).
5. Owen, S., Anil, R., Dunning, T., Friedman, E.: Mahout in Action. (2011).

6. 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).
7. Gantner, Z., Rendle, S.: MyMediaLite: A free recommender system library. Proc. fifth ACM Conf. Recomm. Syst. 305–308 (2011).
8. 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).
9. Shani, G., Gunawardana, A.: Evaluating recommendation systems. Recomm. Syst. Handb. 257–298 (2011).
10. 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).
11. 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).
12. 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).
13. 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).
14. 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).
15. 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).
16. Hernández del Olmo, F., Gaudioso, E.: Evaluation of recommender systems: A new approach. Expert Syst. Appl. 35, 790–804 (2008).
17. 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).
18. Harper, F.M., Konstan, J.A.: The MovieLens Datasets. ACM Trans. Interact. Intell. Syst. 5, 1–19 (2015).
19. Polatidis, N., Georgiadis, C.K., Pimenidis, E., Mouratidis, H.: Privacy-preserving collaborative recommendations based on random

perturbations. Expert Syst. Appl. 71, (2017).
20. 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)