Reproduction of experiments in recommender systems evaluation based on explanations

Nikolaos Polatidis[1] and Elias Pimenidis[2]

[1]School of Computing, Engineering, and Mathematics, University of Brighton, BN2 4GJ, Brighton, United Kingdom
N.Polatidis@Brighton.ac.uk
[2]Department of Computer Science and Creative Technologies, University of the West of England, BS16 1QY, Bristol, United Kingdom
Elias.Pimenidis@uwe.ac.uk

Abstract. The offline evaluation of recommender systems is typically based on accuracy metrics such as the Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE), while on the other hand Precision and Recall is used to measure the quality of the top-N recommendations. However, it is difficult to reproduce the results since there are different libraries that can be used for running experiments and also within the same library there are many different settings that if not taken into consideration when replicating the result might vary. In this paper, we show that it is challenging to reproduce results using a different library but with the use of the same library an explanation based approach can be used to assist in the reproducibility of experiments. Our proposed approach has been experimentally evaluated using a real dataset and the results show that it is both practical and effective.

Keywords: Recommender systems, Evaluation, Explanations, Reproducibility

1 Introduction

Recommender systems are widely known for their use in e-Commerce for recommending products to users, thus reducing the overall searching time of the user and increase sales. Furthermore, it is a technology used in various other less known domains such as music recommendation or people to people recommendation in social media [1]. However, the increasingly use and popularity of recommender systems research both in academia and in industry has lead us to the development of new algorithms and their experimental evaluation. While this is important to do, it should be noted that the problem of reproducing the results exists and it is considered important [2]. For the offline evaluation of recommender systems various metrics can be used such as MAE and RMSE for predicting the accuracy error and information retrieval metrics such as Precision and Recall can be used for measuring the quality of the top-N recommendations [3]. While, there are more metrics it is outside of the scope of this paper to discuss them however further details can be found in [3]. In the literature there are different libraries that can be used for developing and testing a recommendation algorithm and include Recommender101, Apache Mahout, LensKit and MyMediaLite among others [1] [4].

In the work by [1] it has been shown that reproducing the experimental results of an algorithm is very difficult when using a different library because of different settings and parameters that exist between them. However, it is shown that if a set of carefully selected guidelines is followed with the use of the same library then the results can be replicated with a very small and non-noticeable different in the output value.

To assist in solving the problem of reproducibility of experiments with the use of the same recommendation library we have:

1. Developed an explanation based approach that can be used towards this direction.
2. Experimentally evaluated the above approach using a publicly available recommendation library and a real dataset.

The rest of the paper is organized as follows: Section 2 provides the relevant background, section 3 delivers the proposed approach, section 4 presents the experiments and section 5 contains the conclusions.

2 Background

Evaluating recommender systems in offline environments can be done using prediction accuracy or information retrieval metrics. However, the problem arises when in a research output of a new algorithm the source code is not made publicly available or when the exact settings for replicating the code and the experiments are missing. In the literature there are related works that have done important steps towards the solution of the reproducibility problem. In [1] a very good analysis of the main problems is identified, which include the name and the source code of the recommendation library, the details of the algorithm, the dataset used and the details of how the dataset has been used. Moreover, in the same work a set of guidelines is proposed that can be followed to assist in the reproducibility. Another similar work that identifies a set of best practices for recommender systems can be found in [5], while in [6] the importance of the reproducibility of experiments in recommender systems evaluation was highlighted with the organization of a workshop in 2013. Furthermore, the outcome of this workshop can be found in a relevant report with its future directions being theoretical only [7]. One other relevant approach can be found in [8] and it is about the improvement of statistical power of the 10-fold cross validation scheme in recommender systems. A more relevant but more software oriented approach is Rival [9]. In this approach a toolkit provided different stages in the process such as data splitting, item recommendation and evaluation. It is not however a framework or a library but a toolkit that can be used in Apache Mahout, LensKit and MyMediaLite and it provides a user interface. Other researchers however having knowing about the reproducibility problem have decided to develop and propose their own evaluation metrics. For example, in [10] the authors proposed a general evaluation metric that operate over a set of sessions, while another proposed metric can be found in [11] where the authors propose the modified Reciprocal Hit Rand Metric (mRHR) which is a hit rank metric.

In addition to the related works, the most common recommendation method is Collaborative Filtering (CF) and the most know CF method is Pearson Correlation Coefficient (PCC). PCC is defined in equation 1 and 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 $\acute{r}a$ and $\acute{r}b$ represent the user's average ratings. P is the set of all products. Moreover, the similarity value ranges from -1 to 1 and higher is better.

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

Furthermore, to measure the prediction error, MAE it typically be used and is defined in equation 2 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.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |pi - ri| \ (2)$$

However, there are numerous settings found in a recommendation library that can affect the result, such as the number of the nearest neighbors, if a cross-fold evaluation took place or the dataset what split into a training and testing part and the minimum ratings per item or if a threshold of minimum ratings that a user has submitted for an item will be applied. In table 1 we can see the results of PCC using different neighborhood size, the MovieLens 1 million dataset [12], 80% training and 20% testing evaluation and the Recommender101 library. Furthermore, in table 2 it is shown that if the minimum number of ratings per user is different the output can vary significantly on a 5-fold cross validation.

	Number of k nearest neighbours					
	60	80	100	200	300	400
PCC	0.870	0.862	0.841	0.811	0.785	0.761

Table 1. MAE results for Recommender101

Settings	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

Table 2. 5-fold cross-fold with different settings MAE results based on Recommender101

3 Proposed approach

In previous research it has been shown that it is very difficult to reproduce results using different evaluation libraries due to differences that exist between the implementations of algorithms and metrics [1]. However, with the use of the same library the possibility of reproducing correctly an algorithm and an experimental evaluation is high if the same settings and parameters are used.

Thus, for our proposed approach we use the Recommender101 library in combination with a set of explanations that accompany the output log file of the result. The library is comprised from a set of components for offline evaluation as shown in figure 1. The settings used such as the algorithm used, the number nearest neighbors, type of validation (cross fold or test/train) and the algorithm evaluated are passed in an external configuration file. Moreover, it supports well known metrics such as MAE, RMSE, Precision, Recall, NDCG among others and when the experiment is finished the result is printed on the screen and saved in a log file. We extend the Recommender101 library to print on the screen and also save in the log file a set of explanations in simple language that can be used to guide a future researcher to reproduce an experiment.

In addition to the settings used it should be noted that it is difficult to exactly replicate an experiment since in most cases a dataset is randomly divided to training and testing parts. However and despite of this minor issue if the other settings and parameters are properly applied then the result will be close to identical.

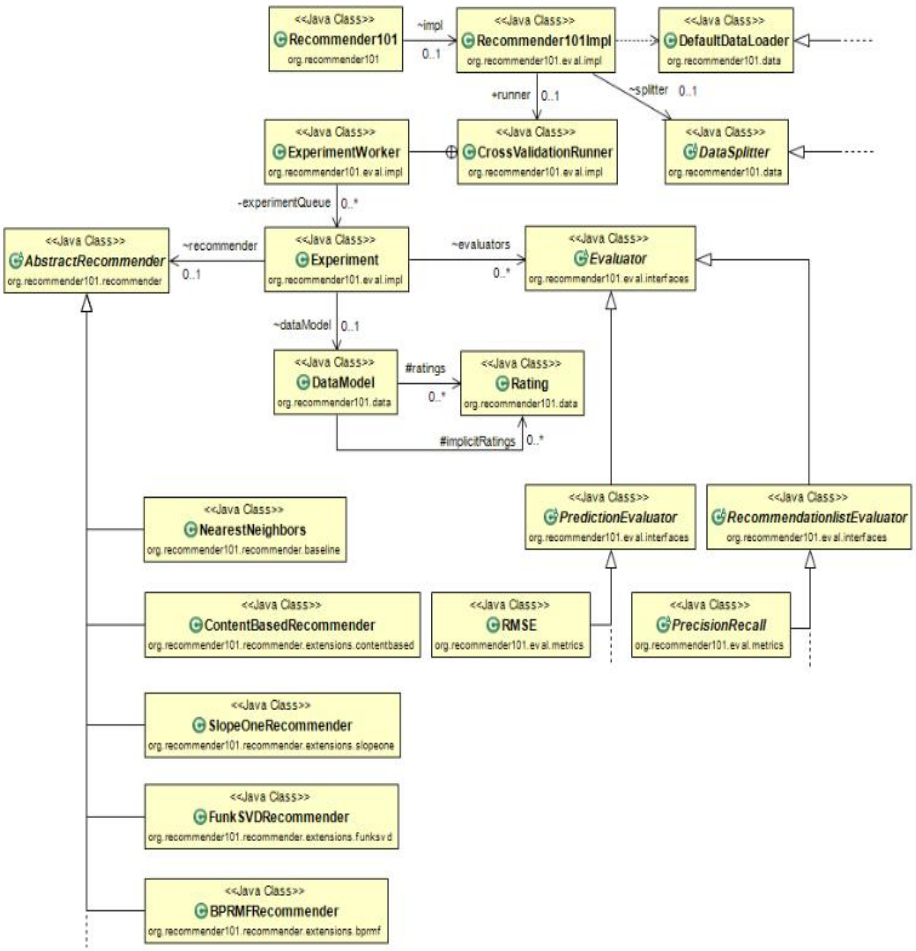

Fig 1. Recommender101 [4]

3.1 Explanations

We define explanations as a set of details that accompany the output result, thus making it clear to the research what needs to be included in a research output. In Recommender101 a number of settings and parameters are available in the configuration file (recommender101.properties under the conf directory). If these parameters are not properly mentioned by researchers in their work then the output result could vary significantly [1].

3.2 The proposed approach

In the proposed approach we:

1. Retrieve information from the configuration file
2. Write the information in the log file along with evaluation result and explain what this is

The settings retrieved from the configuration file are the following and are presented in the same way that are saved in the log file:

1. The configuration parameters and settings can be set at the configuration file recommender101.properties that be found under the conf directory of Recommender101
2. The filename of the dataset is (name of the file goes here)
3. The minimum number of ratings per user to be considered is (number)
4. The minimum number of ratings per considered item is (number)
5. This experiment has used all users OR This experiments has used (number) users
6. The minimum rating value applied is (number e.g. between 1 to 5)
7. The maximum rating value applied is (number e.g. between 1 to 5)
8. This experiment is based on a (number e.g. 5 or 10) cross fold validation OR this experiment is based on a training/test approach using (number %) for training and (number %) for testing
9. The number of nearest neighbors used is (number)
10. The algorithm used is (name)
11. The metrics used for this experiments are (This is already implemented in recommender101)
12. The results are (This is already implemented in recommender101)

4 Experimental evaluation

The experimental evaluation has been based on the MovieLens 1 million dataset [12], which consists of 6040 users, 4000 movies and 1 million ratings in a 1-5 scale. Furthermore the Recommender101 library has been used [4]. Furthermore, we have used PCC as the algorithm and MAE as the evaluation metric to perform an experiment with 80% of the dataset used for training and 20% for testing and reproduce the result. Furthermore for each user to be considered a threshold of 20 ratings was applied.

For the experiments we asked two different researchers to perform an experiment each. Both were instructed to download and install Recommender101 in Eclipse. The first one was instructed to perform an experiment and the second one was instructed to use the log file of the first and reproduce the experiment. The results of the first experiment are presented in table 3 and the results of the second in table 4. The log file included the MAE result using 100, 200 and 300 k nearest neighbors and all the settings explained in section 3.2.

	Number of k nearest neighbors		
	100	200	300
PCC	0.841	0.811	0.785

Table 3. First MAE experiment

	Number of k nearest neighbors		
	100	200	300
PCC	0.842	0.810	0.784

Table 4. Second MAE experiment

5 Conclusions and future work

In this paper we highlighted the problem of reproducibility in recommender systems evaluation. Although, it is shown in previous research that it is difficult to reproduce results using different offline evaluation libraries, the reproducibility of results becomes achievable if the correct settings and parameters are used within the same library. Thus, we have proposed an approach that is based on explanations that can be used to assist researchers in reproducing the results of an experimental evaluation. The initial evaluation results are promising and can assist towards this direction and our approach can be straightforwardly implemented by researchers in other libraries. Furthermore, in our future work we aim to provide a visualized approach of the explanations.

References

1. Polatidis, N., Kapetanakis, S., Pimenidis, E., & Kosmidis, K. Reproducibility of experiments in recommender systems evaluation 14th International Conference on Artificial Intelligence Applications and Innovations, AIAI 2018. IFIP AICT 519 pp.
2. Said, A., Bellogín, A.: Comparative recommender system evaluation. Proc. 8th ACM Conf. Recomm. Syst. - RecSys '14. 129–136 (2014).
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. 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).
5. 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).

6. 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).
7. 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).
8. 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).
9. 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).
10. Hernández del Olmo, F., Gaudioso, E.: Evaluation of recommender systems: A new approach. Expert Syst. Appl. 35, 790–804 (2008).
11. 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).
12. Harper, F.M., Konstan, J.A.: The MovieLens Datasets. ACM Trans. Interact. Intell. Syst. 5, 1–19 (2015).