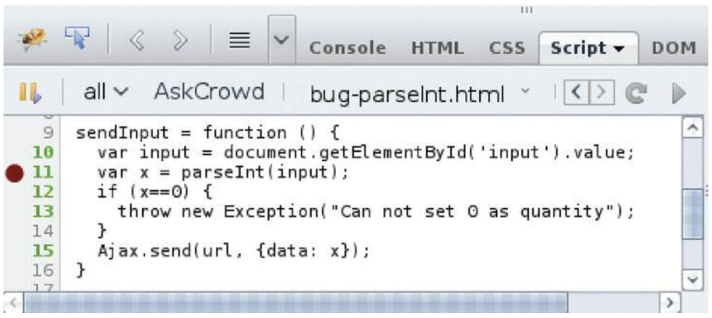

```
 9  sendInput = function () {
10    var input = document.getElementById('input').value;
11    var x = parseInt(input);
12    if (x==0) {
13      throw new Exception("Can not set 0 as quantity");
14    }
15    Ajax.send(url, {data: x});
16  }
```

Figure 1: A screenshot of our prototype crowd-enhanced JavaScript debugger. The button 'AskCrowd' selects the snippet surrounding the breakpoint (Line 11: red circle). A list of selected answers that closely match the problem are then automatically retrieved from StackOverflow. In this case, the first result is the solution: the prefix has a meaning for parseInt which must be taken into account.

opers. To investigate this question, we took a dataset that comprised of 70,060 StackOverflow Q&As that were determined to possibly relate to JavaScript crowd-bugs (dataset available on request). From this dataset, 1000 Q&As and their respective snippets were randomly extracted. We then performed 1,000 queries to the StackOverflow search engine, using the snippets only as an input. This analysis yielded the following results: 377 snippets were considered to be non-valid queries, 374 snippets yielded no results (i.e., the expected Q&A was not found) and finally, 146 snippets yield a perfect match (i.e., the expected Q&A was ranked #1).

These results indicate that StackOverflow does not handle code snippets inputted as query particularly well.

In response to this issue, we introduced pre-processing functions aimed at improving matching quality between the snippets being debugged and the ones on the Q&A repository. We experimented with different pre-processing functions and determined that the best one is based on a careful filtering of the abstract syntax tree of the snippet. Thus, using this pre-processing function, we repeated the same evaluation followed above and on this occasion, 511 snippets yielded a #1 ranked Q&A (full results can be found in our online technical report [1]). Consequently, we integrated this pre-processing function into our prototype crowd-bug debugger to maximize the chances of finding a viable solution.

Beyond this case, we are conducting further research which aims to leverage crowd wisdom to improve the automation of software repair and contribute to the overarching objective of achieving more resilient and self-healing software systems.

Links:
https://team.inria.fr/spirals
http://stackoverflow.com

Reference:
[1] M. Monperrus, A. Maia: "Debugging with the Crowd: a Debug Recommendation System based on Stackoverflow", Technical report #hal-00987395, INRIA, 2014.

Please contact:
Martin Monperrus
University Lille 1, France
Inria/Lille1 Spirals research team
Tel: +33 3 59 35 87 61
E-mail:
Martin.Monperrus@univ-lille1.fr

RiVal: A New Benchmarking Toolkit for Recommender Systems

by Alan Said and Alejandro Bellogín

RiVal is a newly released toolkit, developed during two ERCIM fellowships at Centrum Wiskunde & Informatica (CWI), for transparent and objective benchmarking of recommender systems software such as Apache Mahout, LensKit and MyMediaLite. This will ensure that robust and comparable assessments of their recommendation quality can be made.

Research on recommender systems often focuses on making comparisons of their predictive accuracy, i.e., the better the evaluation scores, the better the recommender. However, it is difficult to compare results between different recommender systems and frameworks, or even assess the quality of one system, due to the myriad of design and implementation options in the evaluation strategies. Additionally, algorithm implementations can diverge from the standard formulation due to manual tuning and modifications that work better in some situations. We have developed an open source evaluation toolkit for recommender systems (RiVal), which provides a set of standardised evaluation methodologies. This was achieved by retaining complete control of the evaluation dimensions being benchmarked (i.e., data splitting, metrics, evaluation strategies, etc.), independent of the specific recommendation strategies.

Recommender systems are a popular means of assisting users of a range of online services in areas, such as music (e.g., Spotify, Last.fm), movies and videos (e.g., Netflix, YouTube) or other items (e.g., Amazon, eBay) [1]. In recent years, research in this field has grown exponentially and today most top-tier research venues feature tracks on recommendation. There has been a parallel development in industry and now, many data science positions place a significant emphasis on candidates possessing expertise in recommendation techniques. This gain in popularity has led to an overwhelming growth in the amount

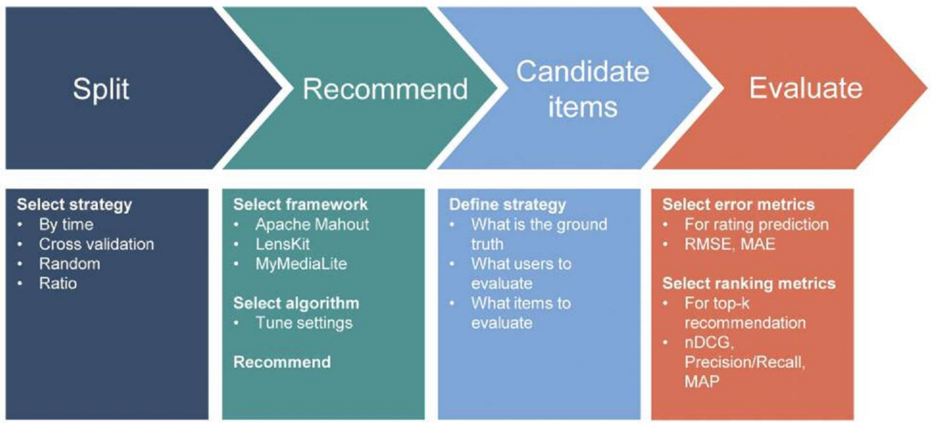

Figure 1: The RiVal evaluation pipeline. The toolkit's modular design means each module can be executed individually (i.e., only the evaluation module or only the data splitting module) or alternatively, the complete pipeline can be executed within RiVal.

of available literature, as well as a large set of algorithms to be implemented. With this in mind, it is becoming increasingly important to be able to benchmark recommendation models against one another to objectively estimate their performance.

Usually, each implementation of an algorithm is associated with a recommendation framework or software library, which in turn, must provide additional layers to access the data, report performance results, etc. An emerging problem associated with having numerous recommendation frameworks is the difficulty in comparing results across software frameworks, i.e., the reported accuracy of an algorithm in one framework will often differ from the same algorithm in a different framework. Minor differences in algorithmic implementation, data management and evaluation are among the number of causes of this problem. To properly analyse this problem, we have developed RiVal, a software toolkit that is capable of efficiently evaluating recommender systems, RiVal can test the various functionalities of recommender systems while simultaneously being agnostic to the actual algorithm in use. It does not incorporate recommendation algorithms but rather, provides bindings or wrappers to the three recommendation frameworks most common at the moment, Apache Mahout, LensKit and MyMediaLite.

RiVal provides a transparent evaluation setting which gives the practitioner complete control of the various evaluation steps. More specifically, it is composed of three main modules: data splitting, candidate item generation and performance measurement. In addition, an item recommendation module is also provided that integrates the three common recommendation frameworks (listed above) into the RiVal pipeline (Figure 1). RiVal is now available on GitHub and further development information can be found on its Wiki and manual pages (see Links section below). The toolkit's features are also outlined in detail in a paper [2] and demo [3]. The toolkit can be used programmatically as Maven dependencies, or by running it as a standalone program for each of the steps.

By using RiVal, we have been able to benchmark the three most common recommendation algorithms implemented in the three aforementioned frameworks using three different datasets. We also generated a large amount of results by using our controlled evaluation protocol, since it consisted of four data splitting techniques, three strategies for candidate item generation, and five main performance metrics. Our results point to a large discrepancy between the same algorithms implemented in different frameworks. Further analyses of these results [2] indicate that many of these inconsistencies were a result of differences in the implementation of the algorithms. However, the implementation of the evaluation metrics and methods also differed across the frameworks, which makes the objective comparison of the recommendation quality across the frameworks impossible when using a framework-internal evaluation.

The RiVal toolkit enables practitioners to perform completely transparent and objective evaluations of recommendation results, which will improve the selection of which recommendation framework (and algorithm) should be used in each situation. Providing an evaluation system, which is highly configurable, unbiased by framework-dependent implementations, and usable across frameworks and datasets, allows both researchers and practitioners to assess the quality of a recommender system in a wider context than current standards in the area allow.

This research was developed while both authors were ERCIM fellows at CWI.

Links:
RiVal: http://rival.recommenders.net
Apache Mahout:
https://mahout.apache.org
LensKit: http://lenskit.org
MyMediaLite:
http://www.mymedialite.net

References:
[1] F. Ricci, L. Rokach, B. Shapira, P.B. Kantor: "Recommender Systems Handbook", Springer, 2011
[2] A. Said, A. Bellogín: "Comparative Recommender System Evaluation: Benchmarking Recommendation Frameworks", in ACM RecSys, 2014
[3] A. Said, A. Bellogín: "RiVal – A Toolkit to Foster Reproducibility in Recommender System Evaluation", in ACM RecSys, 2014.

Please contact:
Alan Said
TU-Delft, The Netherlands
E-mail: alansaid@acm.org

Alejandro Bellogín
Universidad Autónoma de Madrid, Spain
E-mail: alejandro.bellogin@uam.es