RiVal – A Toolkit to Foster Reproducibility in Recommender System Evaluation

Alan Said
TU-Delft
The Netherlands
alansaid@acm.org

Alejandro Bellogín
Universidad Autónoma de Madrid
Spain
alejandro.bellogin@uam.es

ABSTRACT

Currently, it is difficult to put in context and compare the results from a given evaluation of a recommender system, mainly because too many alternatives exist when designing and implementing an evaluation strategy. Furthermore, the actual implementation of a recommendation algorithm sometimes diverges considerably from the well-known ideal formulation due to manual tuning and modifications observed to work better in some situations. RiVal – a recommender system evaluation toolkit – allows for complete control of the different evaluation dimensions that take place in any experimental evaluation of a recommender system: data splitting, definition of evaluation strategies, and computation of evaluation metrics. In this demo we present some of the functionality of RiVal and show step-by-step how RiVal can be used to evaluate the results from any recommendation framework and make sure that the results are comparable and reproducible.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - Information filtering; H.5.1 [**Multimedia Information Systems**]: Evaluation/methodology

General Terms

Experimentation; Documentation; Measurement; Performance

Keywords

Recommender Systems; Evaluation; Benchmarking; Reproducibility; Recommendation Frameworks; Experiments

1. RECOMMENDATION AND EVALUATION

The recommender system research community has access to multiple open source recommendation frameworks, e.g. Apache Mahout[1], LensKit[2], MyMediaLite[3]. One of the emerging problems with having many recommendation frameworks is the difficulty when comparing results across frameworks, i.e. the reported accuracy of an algorithm in one framework will often differ from the same algorithm in a different framework. There are multiple causes for this, some of these are related to minor differences in algorithmic implementation, data management, and evaluation [1]. An example of this is shown in Table 1 which illustrates the different RMSE and nDCG values obtained through each framework's internal evaluation mechanisms (under the same evaluation conditions) using the same algorithms in Apache Mahout (AM), LensKit (LK) and MyMediaLite (MML) on the same dataset – Movielens100k. The table highlights the vast differences between the different frameworks showing that the same algorithm, dataset, and metric can differ several orders of magnitude across frameworks.

This demonstration shows a cross-framework recommender system evaluation toolkit – RiVal[4]. RiVal provides a transparent evaluation setting, allowing the practitioner complete control of the various evaluation settings.

(a) nDCG for AM and LK

Alg.	F.W.	nDCG
Item-based	AM	0.005169231
	LK	0.924546132
SVD 50	AM	0.105427298
	LK	0.943464094
User-based	AM	0.169295451
	LK	0.948413562

(b) RMSE for LK and MML.

Alg.	F.W.	RMSE
Item-based	LK	1.05018614
	MML	0.92933246
SVD 50	LK	1.01209290
	MML	0.93074012
User-based	LK	1.02545490
	MML	0.93419026

Table 1: Root-mean-square error (RMSE) and normalized Discounted Cumulative Gain (nDCG) for item-based and user-based (kNN, k=50) collaborative filtering algorithms using Pearson correlation, and matrix factorization (50 dimensions, FunkSVD in Mahout and Lenskit; SVD++ in MyMediaLite).

2. RIVAL – A TOOLKIT FOR EVALUATION

RiVal is an open source Java toolkit which allows for fine-grained control of the complete evaluation methodology. We have defined the following four stages in the recommendation-evaluation process *i*) *data splitting*; *ii*) *item recommendation*; *iii*) *candidate item generation*; *iv*) *performance measurement*.

[1] `http://mahout.apache.org`
[2] `http://lenskit.grouplens.org`
[3] `http://mymedialite.net`
[4] `http://rival.recommenders.net`

RecSys'14, October 6–10, 2014, Foster City, Silicon Valley, CA, USA.

ACM 978-1-4503-2668-1/14/10.
http://dx.doi.org/10.1145/2645710.2645712 .

Since RiVal is not a recommendation framework, step (*iii*) is not performed by RiVal, but can be performed by any of the three integrated frameworks (Mahout, LensKit and MyMediaLite), or outside of the RiVal pipeline. In this case, steps (*i*), (*iii*), and (*iv*) are performed in the toolkit, whereas in step (*ii*), the preferred recommendation framework is given the data splits generated in the previous step and the recommendations produced by the framework are then given as input to step (*iii*) of RiVal.

The toolkit can either be used as Maven dependencies, or ran as a standalone program for each of the steps. When running the toolkit in standalone mode, the type of evaluation to perform, recommendation algorithms, and framework to use are specified in property files which instantiate the necessary setup and execute each step. Listing 1 shows an example of a configuration file which sets up RiVal to prepare a set of datasets to perform cross validation on. The configuration instantiates the MovielensParser, which assumes the input data has a structure similar to the Movielens datasets (tab- or colon-separated columns). The resulting data splits will be written in the `ml100kcv` folder, separating training and test files through prefixes and/or suffixes, e.g. `mov100k_fold_1_global.train` would be the training file for the first cross validation fold.

Listing 1: Example of data splitter configuration

```
dataset.file=dataset.csv
dataset.parser=net.recommenders.rival.split.\
       parser.MovielensParser
dataset.splitter=net.recommenders.rival.split.\
       splitter.CrossValidationSplitter
split.peruser=false
split.seed=2014
split.cv.nfolds=5
split.output.folder=./ml100kcv/
split.training.prefix=mov100k_fold
split.test.prefix=mov100k_fold
split.training.suffix=_global.train
split.test.suffix=_global.test
```

Analogously, Listing 2 shows an example configuration of a recommendation step performing user-based recommendation using cosine similarity with a neighborhood size set to 50 with the LensKit recommendation framework. The evaluation step, which encompasses both candidate item generation and performance measurement is configured in a similar fashion (not included here due to space constraints). Examples of all configurations are found in the source code of RiVal[5].

Listing 2: Example of recommendation configuration

```
recommender=org.grouplens.lenskit.knn.user.\
       UserUserItemScorer
similarity=org.grouplens.lenskit.vectors.\
       similarity.CosineVectorSimilarity
neighborhood=50
training=./trainset.scv
test=./testset.csv
output=./results.csv
framework=lenskit
```

An example of an evaluation (RMSE) performed on a set of different algorithms (user/item-based CF, SVD), a

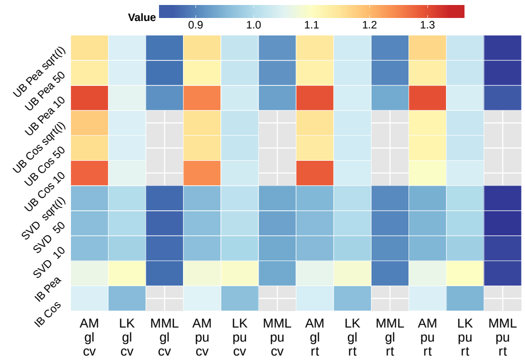

Figure 1: RMSE for a controlled evaluation. IB and UB refer to item- and user-based respectively; Pea and Cos to Pearson and Cosine; gl and pu to global and per user splitting; cv to cross validation and rt to ratio; and AM, LK, MML to the frameworks. (NB: To be viewed in color.)

set of different data splits (cross-validation, per-user, global random) and frameworks (Mahout, LensKit, MyMediaLite) is shown in Fig. 1. The figure highlights that not only does the internal evaluation of each framework differ, even when the evaluation is fully controlled, the results should not be directly compared across frameworks.

3. DEMONSTRATION

In the demonstration, we will be showing how to quickly set up RiVal and run cross-framework comparison (benchmarking) using LensKit, MyMediaLite, and Mahout as recommendation frameworks. Each step in the evaluation protocol (data splitting, recommendation, candidate item generation, performance measurement) will be shown in detail and comparisons across frameworks and different evaluation strategies will be shown in order to highlight the importance of a transparent evaluation setup.

We will also be showing the evaluation setup used by all participants in the 2014 ACM RecSys Challenge[6] as RiVal is the tool used by both participants and organizers in order to measure the performance of the algorithms developed in the scope of the challenge.

4. ACKNOWLEDGMENTS

This work was partly carried out during the tenure of an ERCIM "Alain Bensoussan" Fellowship Programme. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreements n°246016 and n°610594, and the Spanish Ministry of Science and Innovation (TIN2013-47090-C3-2).

5. REFERENCES

[1] A. Said and A. Bellogín. Comparative recommender system evaluation: Benchmarking recommendation frameworks. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys, New York, NY, USA, 2014. ACM.

[5] `http://github.com/recommenders/rival`

[6] `http://2014.recsyschallenge.com`