An Analysis of Tag-Recommender Evaluation Procedures

Stephan Doerfel
University of Kassel
Wilhelmshöher Allee 73
34121 Kassel, Germany
doerfel@cs.uni-kassel.de

Robert Jäschke
L3S Research Center
Appelstraße 4
30167 Hannover, Germany
jaeschke@L3S.de

ABSTRACT

Since the rise of collaborative tagging systems on the web, the *tag recommendation task* – suggesting suitable tags to users of such systems while they add resources to their collection – has been tackled. However, the (offline) evaluation of tag recommendation algorithms usually suffers from difficulties like the sparseness of the data or the cold start problem for new resources or users. Previous studies therefore often used so-called *post-cores* (specific subsets of the original datasets) for their experiments. In this paper, we conduct a large-scale experiment in which we analyze different tag recommendation algorithms on different cores of three real-world datasets. We show, that a recommender's performance depends on the particular core and explore correlations between performances on different cores.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering

Keywords

Recommender; Core; Graph; Evaluation

1. INTRODUCTION

Recommender systems often have to deal with sparse data in which only little or nothing is known about many users or items. Alongside works that specifically tackle this task, it is common to focus on a dense subset of the data [12, 5] that contains enough information to produce helpful recommendations. For graph data, a commonly used technique are *generalized cores* [1] which comprise a dense sub-graph in which every vertex fulfills a specific constraint, e.g., the degree of each node is above a certain threshold. The influence of these cores on the performance of different recommendation algorithms has not been analyzed so far.

In this paper, we investigate cores that have been used in the evaluation of *tag recommender systems.* Tag recommenders are useful in *collaborative tagging systems* (whose underlying data structure is called a *folksonomy* – cf. [4] for the model we use) where users post resources and assign arbitrary keywords (*tags*) to them. During the posting process a recommender system typically suggests appropriate tags for annotation. The tag recommendation task is thus to find suitable tags for a given (user, resource) pair. Previously, tag recommender algorithms have often been evaluated on a special *post-core* of the raw dataset using the *LeavePostOut* method: A post is removed from the dataset and the recommended tags are evaluated against the true tags of the post.

We conduct a thorough experimental analysis of tag recommender rankings on three publicly available real-world datasets and critically compare the results on different cores. As post-cores are so-called "generalized cores" [1], we compare the performance of different algorithms on them to the performance on the original *graph-cores* and on the raw data.

This paper is organized as follows: In Section 2, we review related work. In Section 3, we describe the datasets and the experiments, whose results we discuss in Section 4.

2. RELATED WORK

In many recommendation scenarios, the sparsity of the data is a classical problem which has been tackled either by dealing with the sparseness in particular or by focusing on the dense part of the data (e.g., [12]). Most of the latter approaches are rather ad-hoc, e.g., defining some threshold for the minimal number of ratings an item or user should have. There are few theoretical considerations or experiments that investigate the implications of such thresholds on the performance of different recommender algorithms or the validity of the experiments. One widely applicable methodology to create dense subsets of graphs are the so-called *graph-cores* which were introduced by Seidman [13]. A *graph-core* is a sub-graph in which the degree of each node is above a pre-defined threshold. Upon this work build Batagelj and Zaveršnik [1, 2] who introduce *generalized cores.*

As part of the evaluation of recommender systems, cores have first been used in [5] to focus on the dense part of collaborative tagging systems. Experiments with different tag recommenders were conducted on subsets of such systems, constructed as generalized cores – so-called *post-cores.* A more detailed definition of these cores and extended experiments were presented in [7]. Post-cores were then commonly used in the evaluation of (tag) recommendation algorithms, e.g., in [11] – to compare different PageRank variants on post-cores at levels 5 and 20, in [8] – to evaluate a tag rec-

http://dx.doi.org/10.1145/2507157.2507222

Table 1: The number of users, tags, resources, tag assignments (tas), and posts of the datasets and the levels l chosen for the experiments.

	#users	#res.	#tags	#tas	#posts	chosen l
publ	4 777	94 427	57 639	397 081	109 984	2, 3, 4, 5, 10
book	4 959	231 907	80 603	1 032 037	268 589	2, 3, 4, 5, 6
deli	75 071	2 999 487	397 028	17 280 065	7 268 305	2, 3, 5, 10, 20

ommendation algorithm based on latent dirichlet allocation on a post-core at level 100 of a dataset from Delicious.

3. EXPERIMENTAL SETUP

In the following, we describe the setup of our experiments to test different evaluation procedures with different cores, levels, and metrics for tag recommender algorithms.

3.1 Datasets

We use three publicly available datasets from two collaborative tagging systems (cf. Table 1): The *BibSonomy*[1] dataset (from 2012-01-01) is based on the regular dumps of the publicly available data.[2] The generation of the dataset is described in [6], including a more in-depth description of the data. BibSonomy supports tagging of both bookmarks and publication metadata, hence we split the data into two parts: book and publ. From *Delicious*[3] we use a dataset (deli) we obtained from July 27 to 30, 2005 [4] which is a subset of the Tagora dataset.[4] As Lipczak et al. [9] pointed out, tags from automatically imported posts are problematic for training and evaluating tag recommenders, since their provenance is unknown. The ability of a recommender to (not) predict such tags does not allow us to draw any conclusion about its performance on predicting user-generated tags. Hence, we applied a similar cleansing strategy as described in [9]: we removed sets of posts that were posted at exactly the same time by the same user. In addition, we cleaned all tags as described in [6], i.e., we ignored tag assignments with the tags *imported*, *public*, *system:imported*, *nn*, *system:unfiled*, converted all tags to lower case, and removed all characters which were neither numbers nor letters.

3.2 Evaluation Methodology

The dimensions of our experiments are the three datasets, the two different core types, the chosen levels (see Table 1), the recommendation algorithms, and the evaluation metrics.

Cores and LeavePostOut.

The construction of the graph-cores and post-cores is described in [7]. The data of a tagging system is modelled as a tripartite hypergraph, which has users, resources and tags as nodes and a hyperedge between a user u, a resource r and a tag t, if the user u assigned the tag t to the resource r. The *graph-core at level l* is the largest possible subgraph of that graph, such that each node (i.e., user, tag, resource) occurs in at least l tag assignments, i.e., has a node degree of at least l. In contrast, the *post-core at level l* is the largest subgraph, such that each node occurs in at least l posts. Since one post can consist of several tag assignments (all with the same user and resource, but with different tags), for users and resources, the condition to be part of the core is stronger in the post-core than in the graph-core.

For the experiments we used – besides the raw datasets (or 'cores at level 1') – both graph-cores and post-cores. For each dataset we chose five levels on which we conducted the experiments (see "chosen l" in Table 1). The difference in choice is due to the different characteristics of the datasets (size, unchanged cores over several levels, etc.).

To evaluate the recommenders we used the variant of the leave-one-out hold-out estimation [3] called *LeavePostOut* [5]. LeavePostOut is a standard approach for offline experiments and has been widely used in the literature. Given a dataset (or a core), for each user u one post p is selected at random. This post is eliminated from the dataset and the remaining data is used for training. The task for the recommender algorithm then is to produce tag recommendations (i.e., to predict the tags of p) given both the user and the resource of p, while the tags of p serve as gold-standard. A score is assigned that measures the prediction quality of the recommendation. The experiment is conducted for every user and the scores are averaged. To ensure statistical validity of the results, we repeated each experiment five times and use the averages of the resulting scores.

Evaluation Metrics.

We determine the quality of a recommender by measuring how successful an algorithm can predict the tags of the left-out post. We use the two common metrics *recall* and *precision* at a given cut-off level k (rec@k and pre@k). For a left-out post p, rec@k is the share of p's tags among the top k positions of a recommender's ranking and pre@k the share of the top k tags in the ranking that belong to p. In the experiments we let k run from 1 through 10. The *mean average precision* (MAP) takes the arithmetic mean of the precision at each new recall-level [10].

Algorithms.

The goal of our investigation is not to find the best tag recommendation algorithm, but rather to evaluate the experimental setups, in which algorithms usually are compared against each other. Therefore, we focus on a subset of well-studied tag recommendation algorithms, namely *most popular tags*, *most popular tags by resource*, *most popular tags by user*, *adapted PageRank*, and *FolkRank* [7].

4. RESULTS

When we compare recommenders' scores on different cores and levels with different metrics we first observe that they tend to yield better scores on the post-cores than on the graph-cores of the same level (in 88.6% of the experiments). The performance of the algorithms on the graph-cores and post-cores, is better than that on the raw datasets in 95.5% and 99.2% of the cases, respectively. This increase of the scores raises the question whether the choice of the core has an influence on the comparison of different algorithms against each other. We will therefore investigate correlations between such rankings on different cores in Section 4.2. Before, we have a look at the performance on different core-levels and particular subsets of the data to find out, whether even using the same core, choosing particular posts can yield better results.

[1]http://www.bibsonomy.org/
[2]http://www.kde.cs.uni-kassel.de/bibsonomy/dumps/
[3]http://www.delicious.com/
[4]http://www.tagora-project.eu/data/#delicious

4.1 Recommendation Performance Depends on Core Type and Level

The most prominent observation is that the performance at different core levels depends both on the dataset and on the algorithm. In Figure 1, we see exemplarily the pre@5 scores for the five algorithms over different levels for the graph-core of the datasets publ and deli. Although most often the scores rise with increasing level, there are exceptions like *most popular tags by user* on publ and *adapted PageRank* on deli with (sometimes) sinking scores. The same plots for book (omitted due to space limitations) are similar to publ. The results for rec@5 are similar to those of pre@5.

Further, we leveraged the property that the graph-core always contains the post-core at the same level: Next to the scores on the graph-cores ($\times$) we plotted the scores of the same experiments with only a slight modification of LeavePostOut's post selection process. Where we usually choose one post per user at random, we now choose posts randomly such that they are also contained in the post-core ($+$). Comparing the scores on arbitrarily chosen posts to those from the post-core, we see that in particular for low levels it is easier to predict tags for posts from the post-core than for arbitrarily chosen posts. There are however some exceptions, in particular *FolkRank* and *most popular tags by user*. We conclude that focusing on posts from the the dense part of the data often overestimates the performance of recommendation algorithms.

4.2 Recommender Ranking Correlation

Evaluating recommender systems usually has the goal to determine one algorithm that performs best on one or several datasets and therefore several algorithms are ranked according to their performance. Since several setups for experiments are possible – several core types, levels and metrics – the question arises, whether the ranking of recommenders varies depending on the chosen setup. To investigate this question we determine the algorithm rankings, where the algorithms are ranked according to their recommendation quality. A ranking can be computed on the raw datasets, and on each of the two core types at all chosen levels.[5] Between two rankings (on two different setups) we can determine Pearson's correlation coefficient r, as a measure of how likely the score rankings of the recommenders are (linearly) correlated. The coefficient ranges from -1 (anti-correlation) through 0 (no correlation) to 1 (perfect correlation). As Pearson's r takes the particular score values (the value describing one recommender's performance on one setup) of the algorithms into account, we additionally use another metric that only considers the order of the algorithms in a ranking: the *number of discordant pairs d*.[6] Given two rankings, the algorithms A and B are discordant, when in one ranking A performs better than B while in the other ranking B is better than A. Thus in our case of five algorithms, d is between 0 (the rankings agree completely) and 10 (one ranking is the reverse of the other).

Table 2 shows the mean pairwise (averaged over any pair of two different setups) values of r and d together with the standard deviations exemplarily for the metrics pre@5, rec@5, and MAP. We can observe that on no dataset we get

[5] That is, 11 different setups (see Table 1).
[6] This value is closely related to the ranking correlation measure *Kendall's τ*.

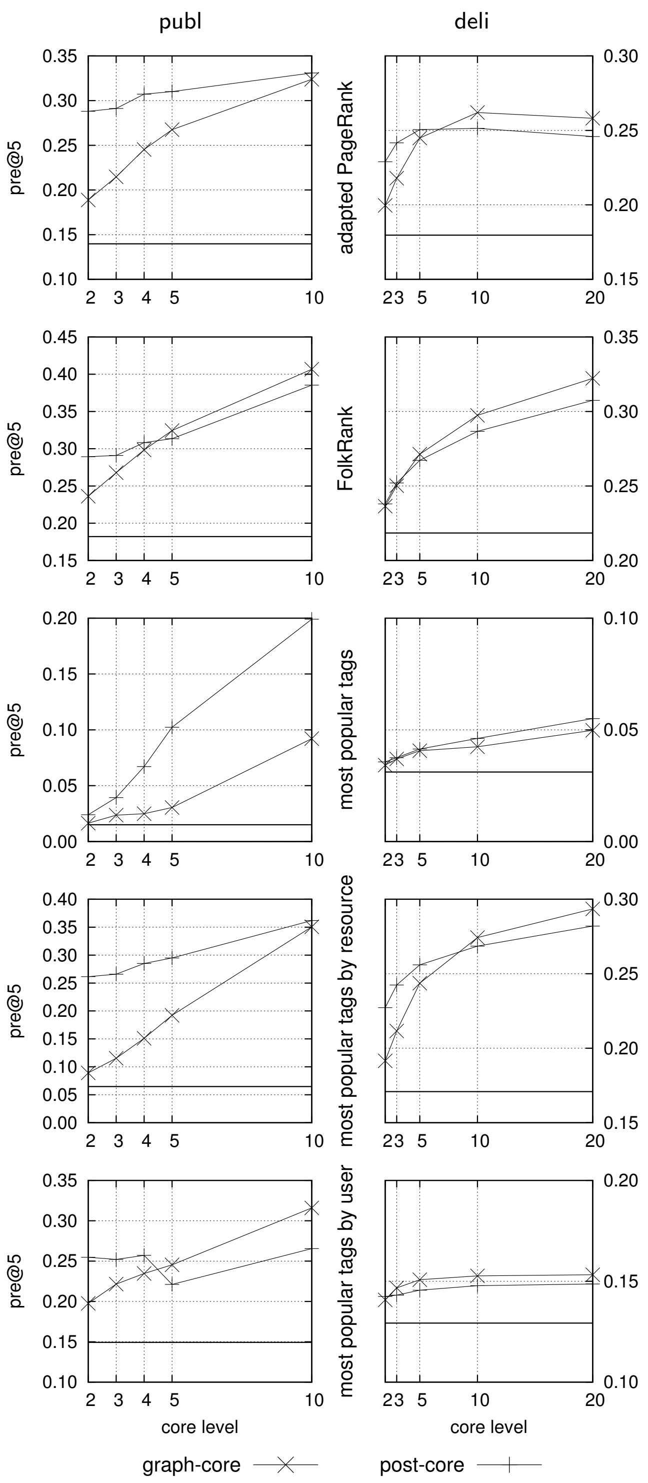

Figure 1: The pre@5 scores over the core level l for publ (left) and deli (right) for the five recommenders using modifications of LeavePostOut. The horizontal line depicts the pre@5 value for the raw dataset.

perfect correlations. Generally, the correlations are rather high, but we clearly see that the rankings are inconsistent. The most stable are the rankings on deli. Here, only in every second pair of setups two recommenders change their

Table 2: The mean pairwise Pearson's r, the number of discordant pairs d in the algorithm rankings on different cores, and their standard deviation σ.

dataset	metric	avg. r	σ	avg. d	σ
publ	MAP	0.890	0.093	1.491	1.069
publ	pre@5	0.886	0.101	1.636	1.007
publ	rec@5	0.894	0.099	1.564	1.014
book	MAP	0.899	0.093	1.491	1.069
book	pre@5	0.870	0.116	1.564	1.151
book	rec@5	0.902	0.091	1.455	1.068
deli	MAP	0.989	0.011	0.545	0.503
deli	pre@5	0.987	0.012	0.545	0.503
deli	rec@5	0.988	0.011	0.545	0.503

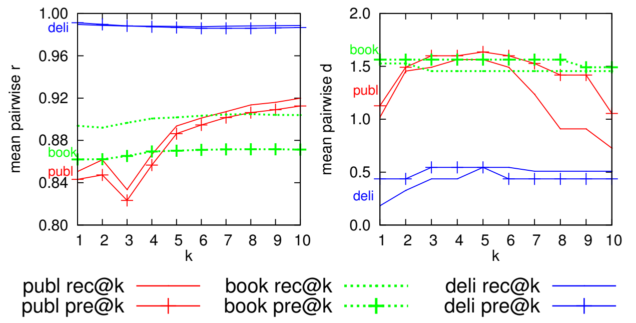

Figure 2: The mean pairwise Pearson's correlation r and number of discordant pairs d over the cut-level k for the metrics rec@k and pre@k.

order. On the two BibSonomy datasets, the values are similar: here, on average, in two rankings one or two pairs of recommenders have different order. Thus it is evident, that different setups lead to different conclusions about the ranking of algorithms. We computed which of the cores yield the ranking that is most consistent with the raw data. For all datasets these are the graph-cores at levels 2 and 3, i.e., the two largest cores.

The consistency of the rankings also depends on the particular metric that is employed. In Figure 2, we see the mean pairwise values of r and d for rec@k and pre@k with k running from 1 through 10. We see that for deli the consistency is quite stable for both metrics precision and recall. However, for the other datasets the values vary and the highest consistency is achieved for $k = 10$.

5. CONCLUSION

We have analyzed the use of cores for the evaluation of tag recommendations. In the experiments, we have shown that using cores for offline evaluation has its pitfalls as recommenders perform differently in different core setups of the same dataset. Focusing on one particular core can produce non-stable results, evaluating the performance of recommenders on another core type or at another core level might yield a different ranking of the algorithms. Thus, we have established, that next to the choice of the dataset(s), the evaluation procedures and measures, the choice of the core is of significance to the outcome of experiments. To minimize this influence, any evaluation should be performed either directly on the raw data or on several core types and levels. On the other hand, we could observe that even cores at higher levels yield correlated results to those of the raw-data. For a preliminary comparison of recommenders it is therefore possible to use several smaller cores to get a quick first impression of their overall performance.

6. ACKNOWLEDGEMENTS

Part of this research was funded by the DFG in the project "Info 2.0 – Informationelle Selbstbestimmung im Web 2.0".

7. REFERENCES

[1] V. Batagelj and M. M. Zaveršnik. Generalized cores. *CoRR*, cs.DS/0202039, 2002.

[2] V. Batagelj and M. Zaveršnik. Fast algorithms for determining (generalized) core groups in social networks. *Advances in Data Analysis and Classification*, 5(2):129–145, 2011.

[3] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.

[4] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Information retrieval in folksonomies: Search and ranking. In *The Semantic Web: Research and Applications*, volume 4011 of *LNCS*, pages 411–426, Berlin/Heidelberg, 2006. Springer.

[5] R. Jäschke, L. Balby Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag recommendations in folksonomies. In *Proc. 11th European Conf. Principles and Practice of Knowledge Discovery in Databases*, volume 4702 of *LNCS*, pages 506–514, Berlin/Heidelberg, 2007. Springer.

[6] R. Jäschke, A. Hotho, F. Mitzlaff, and G. Stumme. Challenges in tag recommendations for collaborative tagging systems. In *Recommender Systems for the Social Web*, volume 32 of *Intelligent Systems Reference Library*, pages 65–87. Springer, Berlin/Heidelberg, 2012.

[7] R. Jäschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag recommendations in social bookmarking systems. *AI Communications*, 21(4):231–247, 2008.

[8] R. Krestel, P. Fankhauser, and W. Nejdl. Latent dirichlet allocation for tag recommendation. In *Proc. 3rd Conf. on Recommender Systems*, pages 61–68, New York, NY, USA, 2009. ACM.

[9] M. Lipczak, Y. Hu, Y. Kollet, and E. Milios. Tag sources for recommendation in collaborative tagging systems. In *ECML PKDD Discovery Challenge*, volume 497 of *CEUR-WS.org*, pages 157–172, 2009.

[10] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, 2008.

[11] M. Ramezani. Improving graph-based approaches for personalized tag recommendation. *Journal of Emerging Technologies in Web Intelligence*, 3(2), 2011.

[12] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proc. 10th Int. Conf. World Wide Web*, pages 285–295, New York, NY, USA, 2001. ACM.

[13] S. B. Seidman. Network structure and minimum degree. *Social Networks*, 5(3):269 – 287, 1983.