Testing and Evaluating Tag Recommenders in a Live System

Robert Jäschke, Folke Eisterlehner, Andreas Hotho, and Gerd Stumme
Knowledge & Data Engineering Group
University of Kassel
Wilhelmshöher Allee 73
34121 Kassel, Germany
http://www.kde.cs.uni-kassel.de/

ABSTRACT

The challenge to provide tag recommendations for collaborative tagging systems has attracted quite some attention of researchers lately. However, most research focused on the evaluation and development of appropriate methods rather than tackling the practical challenges of how to integrate recommendation methods into real tagging systems, record and evaluate their performance.

In this paper we describe the tag recommendation framework we developed for our social bookmark and publication sharing system BibSonomy. With the intention to develop, test, and evaluate recommendation algorithms and supporting cooperation with researchers, we designed the framework to be easily extensible, open for a variety of methods, and usable independent from BibSonomy. Furthermore, this paper presents a first evaluation of two exemplarily deployed recommendation methods.

Categories and Subject Descriptors

H.3.5 [**Information Systems**]: Online Information Services—*Web-based services*; H.2.8 [**Information Systems**]: Database Applications—*Data Mining*

General Terms

Design, Experimentation, Measurement

Keywords

Tag Recommender, Social Bookmarking, Framework

1. INTRODUCTION

Collaborative tagging systems are web based systems that allow users to assign keywords – so called *tags* – to arbitrary resources. Tags are used for navigation, finding resources and serendipitous browsing and thus provide an immediate benefit for users. These systems usually include tag recommendation mechanisms easing the process of finding good tags for annotating a resource. Recommending tags can serve various purposes, such as: increasing the chances of getting a resource annotated, reminding a user what a resource is about and consolidating the vocabulary across the users. Furthermore, as Sood et al. [9] point out, tag recommendations "fundamentally change the tagging process from generation to recognition" which requires less cognitive effort and time.

Our contributions with this paper are: (i) presenting and evaluating a tag recommendation framework deployed in BibSonomy, an open collaborative tagging system, (ii) providing researchers a testbed to test and evaluate their methods in a live system, and (iii) showing first results which indicate that the framework can be used to improve recommendation performance, e. g., by clever selection strategies.

2. APPLICATION

As foundation and testbed for our framework we use our social bookmark and publication sharing system *BibSonomy* [5]. Users of BibSonomy can organize their bookmarks and publication references by annotating them with tags. Plenty of features support them in their work: groups, tag editors, relations, various import and export options, etc. In particular, a REST-like API[1] eases programmatic interaction with BibSonomy and is the cornerstone of external cooperation with the presented tag recommendation framework. Technically, BibSonomy is based on several Java modules[2] which are merged in a Java Servlet/ServerPages based web application with an SQL database as backend.

The datastructure underlying most collaborative tagging systems and also BibSonomy is called *folksonomy*. It describes the assignment of *tags* by *users* to *resources*. Formally, a *folksonomy* is a tuple $\mathbb{F} := (U, T, R, Y)$ where U, T, and R are finite sets, whose elements are called *users*, *tags* and *resources*, resp., and Y is a ternary relation between them, i. e., $Y \subseteq U \times T \times R$, whose elements are called tag assignments (*tas* for short).[3]

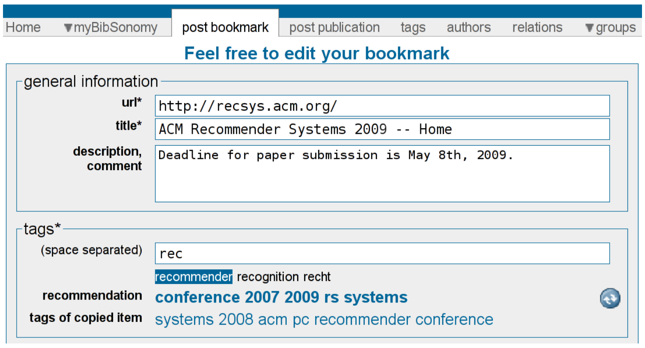
Currently, tag recommendations in BibSonomy appear in two situations: when the user edits a bookmark or publication post. Since the part of the user interface showing recommendations is very similar for both the bookmark posting and the publication posting page, we show in Figure 1 the

http://dx.doi.org/10.1145/1639714.1639790

[1] http://www.bibsonomy.org/help/doc/api.html

[2] Some of them are freely available at http://dev.bibsonomy.org/.

[3] In the original definition [6], we introduced additionally a subtag/supertag relation, which we omit here.



Feel free to edit your bookmark

general information

url* http://recsys.acm.org/

title* ACM Recommender Systems 2009 -- Home

description, comment: Deadline for paper submission is May 8th, 2009.

tags*

(space separated) rec

recommender recognition recht

recommendation **conference 2007 2009 rs systems**

tags of copied item systems 2008 acm pc recommender conference

Figure 1: BibSonomy's recommendation interface on the bookmark posting page.

relevant part of the 'postBookmark' page only.

The 'tags' box contains a text input field where the user can enter the (space separated) tags, tags suggested for autocompletion, the tags from the recommender (bold), and the tags from the post the user just copies. To the very right of the recommendation is a small icon depicting the *reload* button. It allows the user to request a new tag recommendation if he is unsatisfied with the one shown or wants to request further tags.

Besides triggering autocompletion with the tabulator key during typing, users can also click on tags with their mouse. They are then added to the input box. When the user copies a resource from another user's post, the tags the other user used to annotate the resource are shown below the recommended tags ('tags of copied item'). They are also regarded for autocompletion.

The *tag recommendation task* is: Given a resource r and a user u who wants to annotate r, the recommender shall return a set of recommended tags $T(u,r) := \{t_1,\ldots,t_k\}$ together with a *scoring function* $f\colon T(u,r) \to [0,1]$ which assigns to each tag a score.[4] The value of k is fixed to 5 throughout this paper.

3. RELATED WORK

Although having a different recommendation target (resources rather than tags), the REFEREE framework described by Cosley et al. [4] is most closely related to our work. It provided recommendations for the CiteSeer (formerly ResearchIndex) digital library. Besides the different recommendation target, the focus of the work is more on the evaluation of several different strategies than on the details of the framework. A powerful, open, and well documented framework for recommendations is the Duine Framework[5] developed by Novay. It is based on work by van Setten [10] and has a focus on explicit user ratings and non re-occuring items, e. g., like in a movie recommendation scenario where one does not recommend movies the user has already seen. This is in contrast to tag recommendations, where re-occuring tags are a crucial requirement of the system. Similar to what we present in Section 4.2, the framework implements various hybrid recommenders (for a survey on hybrid recommender see e. g., [2]).

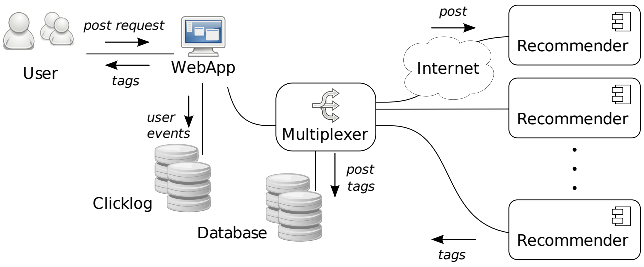


Figure 2: A schematic posting process.

The topic of tag recommendations in social bookmarking systems has attracted quite a lot of attention in the last years. Most related work describes recommendation approaches which could be used within our framework. The existent approaches usually lay in the collaborative filtering and information retrieval areas [8, 3, 9]. Xu et al. [12] identify properties of good tag recommendations like high coverage of multiple facets, high popularity, or least-effort and introduce a collaborative tag suggestion approach. Further examples include Basile et al. [1], suggesting an architecture of an intelligent tag recommender system, and Vojnovic et al. [11], trying to imitate the learning of the true popularity ranking of tags for a given resource during the assignment of tags by users.

4. A RECOMMENDATION FRAMEWORK FOR BIBSONOMY

Figure 2 gives an overview on the components of BibSonomy involved in a recommendation process. The web application receives the user's HTTP request and queries the multiplexer (cf. Sec. 4.4) for a recommendation – providing it post information like URL, title, user name, etc.. In addition, click events are logged in a database. The multiplexer then requests the active recommenders to produce recommendations and selects one of the results. The suggested tags together with the post are then logged in a database and the selected recommendation is returned to the user.

4.1 Recommender Interface

One central element of the framework is the recommender interface. It specifies which data is passed from a recommendation request to one of the implemented recommenders and how they shall return their result. The *getRecommendedTags* method returns – given a post – a sorted set of tags; *addRecommendedTags* adds to a given (not necessarily empty) collection of tags further tags. Since – given a post and an empty collection – *addRecommendedTags* should return the same result as *getRecommendedTags*, the latter can be implemented by delegation to the former. For measuring and thus potentially improving its performance, the final post, as it is stored in the database, is given to the recommender via the *setFeedback* method. Two further classes augment the interface: The *RecommendedTag* class basically extends the *Tag* class as used in the BibSonomy API by adding a floating point *score* attribute. A corresponding *RecommendedTagComparator* can be used to compare tags, e. g., for sorted sets.

[4] Although, of course, f also depends on u and r, we will omit those two variables to simplify notation. Since f always appears together with $T(u,r)$, it should be clear from context, which f is meant.

[5] http://duineframework.org/

Our implementation is based on Java and all described classes are contained in the module *bibsonomy-model*, which is available online as JAR file in a Maven2 repository.[6] However, implementations are not restricted to Java – using the remote recommender (see Sec. 4.3) one can implement a recommender in any language which is then integrated using XML over HTTP requests.

4.2 Meta Recommender

Meta or *hybrid recommenders* [2] do not generate recommendations on their own but instead call other recommenders and modify or merge their results. Since they implement the same interface, they can be used like any other recommender. More formally, given n recommendations $T_1(u,r),\dots,T_n(u,r)$ and corresponding scoring functions $f_1,\dots,f_n$, a meta recommender produces a merged recommendation $T(u,r)$ with scoring function f.

4.2.1 First Weighted By Second

As an example of a cascade hybrid, the idea behind this recommender is to re-order the tags of one recommendation using scores from another recommendation. More precisely, given recommendations $T_1(u,r)$ and $T_2(u,r)$ and corresponding scoring functions f_1 and f_2, this recommender returns a recommendation $T(u,r)$ with scoring function f, which contains all tags from T_1 which appear in T_2 (with $f(t) := f_2(t)$) plus all the remaining tags from T_1 (with lower f but respecting the order induced by f_1). If $T_1(u,r)$ does not contain enough recommendations, T is filled by the not yet used tags from $T_2(u,r)$ – again with f being lower than for the already contained tags and respecting the order induced by f_2.

4.2.2 Weighted Merging

This weighted hybrid recommender enables merging of recommendations from different sources and weighting of their scores. Given n recommendations $T_1(u,r),\dots,T_n(u,r)$, corresponding scoring functions $f_1,\dots,f_n$, and (typically fixed) weights $\rho_1,\dots,\rho_n$ (with $\sum_{i=1}^n \rho_i = 1$), the weighted merging recommender returns a recommendation $T(u,r) := \bigcup_{i=1}^n T_i(u,r)$ and a scoring function $f(t) := \sum_{i=1}^n \rho_i f_i(t)$ (with $f_i(t) := 0$ for $t \notin T_i(u,r)$).

4.3 Remote Recommender

The remote recommender retrieves recommendations from an arbitrary external service using HTTP requests in REST-based interaction. Therefore, it uses the XML schema of the BibSonomy REST-API.[7] This recommender has three advantages: it allows us to distribute the recommendation work over several machines, it opens the framework to include recommenders from auxilliary partners, and it enables programming language independent interaction with the framework.

4.4 Multiplexing Tag Recommender

Our framework's technical core component is the so called *multiplexing tag recommender* (see Fig. 2). Implementing BibSonomy's tag recommender interface, it provides the web application with tag recommendations, using one of the recommenders available. All recommendation requests and each recommender's corresponding result are logged in a database. For this purpose, every tag recommender is registered during startup and assigned to a unique identifier. For technical reasons, we differentiate between locally installed and remote recommenders (cf. Sec. 4.3).

Whenever the *getRecommendedTags* method is invoked, the corresponding recommendation request is delegated to each recommender, spawning separate threads for each recommender. After a timeout period of 100 ms, one of the collected recommendations is selected, applying a preconfigured *selection strategy*:

For our evaluation process we implemented a '*sampling with replacement*' strategy, choosing exactly one recommender i and all of its recommended tags $T_i(u,r)$ together with its scoring function f_i. If the user requests tag recommendations more than once (e. g., using the 'reload' button), this process is repeated independently from previous requests.

4.5 Example Recommender Implementations

Here we describe two of the recommenders which are currently active in BibSonomy. The short names in parentheses are for later reference.

4.5.1 Most Popular ρ-Mix (MPρ-mix)

We implemented a variant of the *most popular ρ-mix* recommender described in [7]. The recommender has been implemented as a combination of three recommenders, using a value of $\rho = 0.6$: a) the *most popular tags by resource* recommender which returns the k tags $T_1(u,r)$ which have been attached to the resource most often, b) the *most popular tags by user* recommender which returns the k tags $T_2(u,r)$ the user has used most often (with $f_2(t) := \frac{|Y \cap \{u\}\times\{t\}\times R|}{|Y\cap\{u\}\times T\times R|}$, i. e., the relative tag frequency), and c) the *weighted merging* meta recommender described in Section 4.2.2 which merges the tags of the two former recommenders, with weights $\rho_1 = \rho = 0.6$ and $\rho_2 = 1 - \rho = 0.4$.

4.5.2 Title Tags Weighted by User Tags (TbyU)

This method ranks tags extracted from the resource's title using the frequency of the tags used by the user. Technically, this is again a combination of three recommenders: a) a simple *content based recommender*, which extracts k tags $T_1(u,r)$ from the title of a resource, cleans them and checks against a multilingual stopword list, b) the *most popular tags by user* recommender as described in the previous section – here returning *all* tags $T_2(u,r)$ the user has used (by setting $k = \infty$), and c) the *first weighted by second* meta recommender described in Section 4.2.1 which weights the tags from the content based recommender by the frequency of their usage by the user as given by the second recommender.

5. EVALUATION

As performance measures we use precision, recall, and f1-measure (f1m). For each post (u, T_{ur}, r) we compare the recommended tags $T(u,r)$ with the tags T_{ur} the user has finally assigned. Then, precision and recall of a recommendation are defined as $\text{recall}(T(u,r)) = \frac{|T_{ur}\cap T(u,r)|}{|T_{ur}|}$ and $\text{precision}(T(u,r)) = \frac{|T_{ur}\cap T(u,r)|}{|T(u,r)|}$. We then average these values over all posts in the given set and compute the f1-measure as $\text{f1m} = \frac{2\cdot\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}}$.

[6] http://dev.bibsonomy.org/maven2/org/bibsonomy/bibsonomy-model/

[7] http://www.bibsonomy.org/help/doc/xmlschema.html

Before intersecting T_{ur} with $T(u, r)$, we clean the tags by ignoring their case and removing all characters which are neither numbers nor letters. Finally, we ignore tags which are 'empty' after normalization (i. e., they neither contained a letter nor number) or which are equal to the strings *imported*, *public*, *systemimported*, *nn*, *systemunfiled*.

We store in a database for each recommendation process the corresponding bookmark or BibTeX entry, each recommender's recommendation, as well as the applied selection strategy together with the recommenders and tags selected are stored. Additionally, the user interaction is tracked by logging mouse click events using JavaScript. Each click on one of BibSonomy's web pages is logged using AJAX into a separate logging table. Information like the shown page, the DOM path of the clicked element, the underlying text, etc is stored.

6. RESULTS

For space reasons we provide only a brief analysis of the framework on data from posting processes between April 8th and May 8th 2009. Only public posts from users not flagged as spammer were taken into account.

We start with some general numbers: In the analysed period, 4,372 posting processes (2,168 for BibTeX, 2,204 for bookmarks) have been provided with tag recommendations. The MPρ-mix recommender served recommendations for 2,276 postings, the TbyU recommender for 2,251. In general, the f1-measure is rather low: For the MPρ-mix recommender it increases from 0.162 for one tag to 0.244 for five tags; for the TbyU recommender from 0.171 to 0.222.

Looking at the influence of the 'reload' button we discovered that in 669 (258 bookmark, 411 BibTeX) of the 4,372 posting processes the users requested to reload the recommendation. Thus, in around 15 % of all posting processes users requested another recommendation.

Next we evaluate the data from the log which records when a user clicked on a recommended tag. Evaluation revealed, that although clicks are rather sparse (in only 802 of the 4,372 posting processes users clicked on a recommendation), a large fraction of correctly recommended tags has been clicked instead of typed.

An investigation of the average f1-measure of each recommender shows that the performance for most of the users does not vary much. However, there are users where one of the two recommenders performed better than the other, even for users with higher post counts. Once such a user is identified, one could primarily select recommendations from the user's preferred recommender.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we presented the tag recommendation framework we developed for BibSonomy. It allows us to not only integrate and judge recommendations from various sources but also to develop clever selection strategies. A strength of the framework is its ability to log all steps of the recommendation process and thereby making it traceable. The framework will be the cornerstone of this year's ECML PKDD Discovery Challenge,[8] where one task requires the participants to deliver live recommendations for BibSonomy.

Acknowledgement. Part of this research was funded by the European Union in the Tagora (FET-IST-034721) project and by the DFG in the project "Info 2.0 – Informationelle Selbstbestimmung im Web 2.0".

8. REFERENCES

[1] P. Basile, D. Gendarmi, F. Lanubile, and G. Semeraro. Recommending smart tags in a social bookmarking system. In *Bridging the Gap between Semantic Web and Web 2.0 (SemNet 2007)*, pages 22–29, 2007.

[2] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, Nov. 2002.

[3] A. Byde, H. Wan, and S. Cayzer. Personalized tag recommendations via tagging and content-based similarity metrics. In *Proceedings of the International Conference on Weblogs and Social Media*, March 2007.

[4] D. Cosley, S. Lawrence, and D. M. Pennock. REFEREE: an open framework for practical testing of recommender systems using ResearchIndex. In *VLDB '02: Proceedings of the 28th international conference on Very Large Data Bases*, pages 35–46. VLDB Endowment, 2002.

[5] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. BibSonomy: A social bookmark and publication sharing system. In A. de Moor, S. Polovina, and H. Delugach, editors, *Proceedings of the Conceptual Structures Tool Interoperability Workshop at the 14th Int. Conf. on Conceptual Structures*, Aalborg, Denmark, July 2006. Aalborg University Press.

[6] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Information retrieval in folksonomies: Search and ranking. In Y. Sure and J. Domingue, editors, *The Semantic Web: Research and Applications*, volume 4011 of *Lecture Notes in Computer Science*, pages 411–426, Heidelberg, June 2006. Springer.

[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] G. Mishne. Autotag: a collaborative approach to automated tag assignment for weblog posts. In *WWW '06: Proceedings of the 15th International Conference on World Wide Web*, pages 953–954, New York, NY, USA, 2006. ACM Press.

[9] S. Sood, S. Owsley, K. Hammond, and L. Birnbaum. TagAssist: Automatic tag suggestion for blog posts. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM 2007)*, 2007.

[10] M. van Setten. *Supporting people in finding information : hybrid recommender systems and goal-based structuring.* PhD thesis, University of Twente, Enschede, The Netherlands, Dec. 2005.

[11] M. Vojnovic, J. Cruise, D. Gunawardena, and P. Marbach. Ranking and suggesting tags in collaborative tagging applications. Technical Report MSR-TR-2007-06, Microsoft Research, 2007.

[12] Z. Xu, Y. Fu, J. Mao, and D. Su. Towards the semantic web: Collaborative tag suggestions. In *Proceedings of the Collaborative Web Tagging Workshop at the WWW 2006*, Edinburgh, Scotland, May 2006.

[8] `http://www.kde.cs.uni-kassel.de/ws/dc09`