Tag Recommendations Based on Tracking Social Bookmarking Systems

Szymon Chojnacki

Department of Artificial Intelligence, Institute of Computer Science,
Polish Academy of Sciences

Abstract. The purpose of this paper is to describe our approach to Tag Recommendation Task during ECML PKDD 2009 Challenge. The organizers supplied a training set of tagged webpages and publications from BibSonomy portal. Our goal was to build a model which can predict tags for new users bookmarking digital resources. Our strategy was based on an assumption that users tend to tag the same resources in various systems. Therefore, have we developed a tracking engine, which was adjusted to the profile of BibSonomy users in selection of RSS feeds and utilized the training data to optimize the list of tracked URLs. We had over 90 days to collect the data from the feeds, but this period did not overlap with the dates of posts from the training set. As a result we had to set manually parameters responsible for a trade-off between recall and accuracy of the model. We stored all downloaded feed entries in a searching engine. The recommendation was based on tags attached to the documents retrieved from the engine by means of typical information retrieval query.

Keywords: Information Retrieval, Searching Engines, Tag Recommendations.

1 Introduction

The development of collaborative society that we experience in recent years can be characterized by four principles: being open, peering, sharing and acting globally [6]. These principles determine the way we exchange information and organize the knowledge. Very important part of this phenomenon is the popularity of social classification, indexing and tagging. Attaching labels to common resources (webpages, blogs, music, videos, photos) can on one hand shed a new light on information retrieval problems, on the other hand poses new challenges concerning uncontrolled explosion of folksonomy size and its usability. The goal of our research is to build a tag recommendation system that would influence user's selection of tags and as a result enable us to reuse folksonomy entries in more efficient way than we observe currently

This paper describes our attempt to predict tags already chosen by BibSonomy users. This was the Task 1 in ECML PKDD 2009 Challenge. However, we believe

that our system is better suited for the third Task, in which the Teams have an opportunity to deliver recommendations online.

2 Related work

The growing interest of research community in the field of social bookmarking was fueled by last year's ECML challenge, during which the evaluation measures were standardized and benchmark data sets prepared. Thirteen solutions were submitted to the tag spam detection task and only five to tag recommendation task. We were inspired by the best teams in the Challenge, which relied on several external resources [7] and used only data available in title/description fields [5]. The team from the Aristotle University of Thessaloniki [3] reformulated the task as a multilabel classification problem.

3 Examined datasets

We used cleaned dump dataset which consisted of three tables: bibtex (158 924 records), bookmark (263 004 records) and tas (1 401 104 records). The dump contained all public bookmarks and publication posts of BibSonomy until (but not including) 2009-01-01. Posts from the user dblp (a mirror of the DBLP Computer Science Bibliography) as well as all posts from users which have been flagged as spammers have been excluded. Furthermore, the tags were cleaned. Java method was used to remove all characters which were neither numbers nor letters and removed those tags, which were empty after cleansing or matched one of the tags imported, public, systemimported, nn, systemunfiled.

The tas table (Tag Assignments) was a fact table with information about who attached which tag to which resource/content. The bookmark table consisted of following columns (content_id, url_hash, url, description, extended description and date). The bibtex table was described by following dimensions (content_id, journal, volume, chapter, edition, month, day, booktitle, howPublished, institution, organization, publisher, address, school, series, bibteXKey, url, type, description, annote, note, pages, key, number, crossref, misc, bibtexAbstract, simhash0, simhash1, simhash2, entrytype, title, author, edition, year).

4 Our approach

In this section we describe three main parts of our system. Firstly we focus on a selection of RSS feeds and the problems we encountered while downloading the posts. In the second part we define the vector space in which the posts were stored as well as main characteristics of deployed database. Finally we present the details of the tag recommendation algorithm. The algorithm is divided into four steps: searching of matching resources based on URL address, retrieval of the most similar cluster, selection of the post with highest overlap score and ranking of suggested tags.

4.1 RSS Feeds selection

Our strategy was to optimize a set of keywords that we were going to track in popular bookmarking systems as well as in a variety of domain portals. We analyzed distribution of most common tags in BibSonomy and Delicious and decided that tracking only the most recent posts would be biased (Table 1). We decided to enrich the most recent posts with a set of 100 most popular tags (out of 93 757 unique tags) in BibSonomy training data.

Table 1. The most common tags in Delicious and Bibsonomy

	Tag	Delicious[1]	BibSonomy[2]
1	design	1.69%	27
2	blog	1.29%	13
3	tools	1.05%	10
4	software	0.96%	4
5	webdesign	0.92%	54
6	programming	0.89%	5
7	tutorial	0.85%	44
8	art	0.75%	83
9	reference	0.72%	33
10	video	0.72%	3
11	inspiration	0.71%	587
12	music	0.66%	25
13	web2.0	0.65%	7
14	education	0.63%	17
15	photography	0.52%	166

We had to face different problems in case of bookmarking systems and domain portals. We used Google Reader to search for top 10 domain portals and their RSS URLs for each chosen keyword. Because some feeds appeared in different searching results we end up with 734 feeds.

An example of feeds recommended by Google Reader for a keyword “linux” is presented in Table 2. Even though numerous feeds use the most recent RSS or Atom standard and we could easily parse the content of XML files, it is uncommon to fill in the category field by feed editors. We can see in the Table 2, that out of 10 sources: one did not contain proper URL, four did not deliver information about category, one marked each feed entry with the same category.

[1] Relative frequency of a tag in a random collection of 603 750 downloaded from the Delicious.
[2] Rank of a corresponding tag in the BibSonomy.

Table 2. Results of searching first ten feeds for "Linux" keyword in Google Reader.

RSS Feed	Categories of updated entries
Linux Insider[3]	| Community | Community | Distros | Licensing | Financial News | Mobile | Community | Community | Mobile
Linux Magazine[4]	No Categories
DistroWatch.com: News[5]	No Categories
Linux Today[6]	No Categories
Slashdot: Linux[7]	| programming | xwindows | google | gui | software | microsoft | portables | os | storage | linuxbusiness | security | gnu | education | caldera | portables
Linux.com :: Features[8]	URL broken
HowtoForge - Linux Howtos and Tutorials[9]	| Ubuntu | Debian | Ubuntu | Desktop | Debian | Lighttpd | Ubuntu | Desktop | Virtualization | Ubuntu | Desktop | Security | Ubuntu | CentOS | Samba | Ubuntu | Desktop | Linux | Ubuntu | Security | Ubuntu | Desktop | Fedora | Security
Linux and Open Source - RSS Feeds[10]	No Categories
LinuxQuestions.org[11]	| Linux - Newbie | Linux - Newbie | Linux - Newbie | Linux - Software | Programming | Red Hat | Linux - General | Linux - General | Linux - Laptop and Netbook | Puppy | Ubuntu | Linux - Desktop | Ubuntu | Ubuntu | Linux - Security
LXer Linux News[12]	| linux | linux | linux | linux | linux | linux | linux | linux | linux | linux | linux | linux | linux | linux | linux | linux | linux | linux | linux | linux

On the other hand the problem with typical bookmarking systems is the fact that when we subscribe most recent posts for a given keyword we get only tags of a particular user who bookmarked the resource. As a consequence we need to crawl a service in order to find out about most typical tags for a given resource. The problem of

[3] http://www.linuxinsider.com/perl/syndication/rssfull.pl
[4] http://www.linux-mag.com/cache/rss20.xml
[5] http://distrowatch.com/news/dw.xml
[6] http://linuxtoday.com/backend/biglt.rss
[7] http://rss.slashdot.org/Slashdot/slashdotLinux
[8] http://www.linux.com/index.rss
[9] http://www.howtoforge.com/node/feed
[10] http://rssnewsapps.ziffdavis.com/eweeklinux.xml
[11] http://www.linuxquestions.org/syndicate/lqlatest.xml
[12] http://lxer.com/module/newswire/headlines.rss

connection limits arises when we want to crawl every out of 100 entries downloaded for a given keyword. Because of this, we decided to verify if we can cluster tags based on their cooccurence score.

Table 3 contains 20 pairs of tags with highest symmetric Jaccard cooccurance coefficient calculated as a division of number of posts with both tags by a number of all posts with any of the tags.

Table 3. Co-occurances of pairs of tags, occurances and normalized Jaccard coefficient.

Tag 1	Tag 2	$\lvert t_1 \cap t_2 \rvert$	$\lvert t_1 \rvert$	$\lvert t_2 \rvert$	$\frac{\lvert t_1 \cap t_2 \rvert}{\lvert t_1 \cup t_2 \rvert}$
ccp	jrr	4294	4294	4294	1.00
algorithms	genetic	5775	6220	5888	0.91
aaaemulation-topgames	emulation-videogames	3653	3653	4576	0.80
emulationgames	emulation-videogames	4576	6055	4576	0.76
aaaemulation-topgames	classicemulated-remakeretrogames	2472	3653	2472	0.68
aaaemulation-topgames	emulationgames	3653	3653	6055	0.60
classicemulated-remakeretrogames	emulation-videogames	2472	2472	4576	0.54
genetic	programming	5262	5888	9491	0.52
journal	medical	1693	2566	2448	0.51
algorithms	programming	5303	6220	9491	0.51
classicemulated-remakeretrogames	emulationgames	2472	2472	6055	0.41
book	nlp	1230	2614	2027	0.36
education	learning	2143	5021	4751	0.28
media	texts	1998	7149	2012	0.28
analysis	data	1187	3352	2589	0.25
folksonomy	tagging	1027	2561	3083	0.22
emulationgames	zzztosort	2844	6055	11839	0.19
audio	music	919	1857	4142	0.18
howto	tutorial	850	2876	2798	0.18
bookmarks	indexforum	9164	52795	9183	0.17

We can see that “ccp” and “jrr” always appear together. Also “genetic”, “algorithms” and “programming” create a cloud of tags. Four tags “emulationgames”, “emulationvideogames”, “aaaemulationgames”, “classicemulatedremakeretrogames” create another cloud. However, the Jaccard coefficient drops very fast below 20% level and therefore we decided not to abandon the idea of tag clustering.

4.2 Data storage

In order to recommend tags online we needed a fast engine that does not need to be taught every time we get need posts from a scratch. The Beatca system (developed in our Institute [1,4]) is an example of such engine. It performs online incremental hierarchical clustering of documents and proved very effective in the field of intelligent Information Retrieval. Soft classification of documents and construction of conceptual closeness graph is based on large-scale Bayesian networks. Optimal document map search and document clustering is based on SOM (self-organizing maps), AIS (artificial immune systems), and GNG (growing neural gas).

Each post is defined as a point in a multidimensional space in which coordinates represent frequency of a token appearing in a post's title or description. Because some tokens are very common and others are present in only few posts we selected only the most informative tokens as coordinates in our vector space. The dictionary optimization was based on a entropy-like quality measure $Q(t_i)$ of a token t_i:

$$Q(t_i) = \frac{N_i}{N} \cdot \frac{-\sum_{j=1}^{N} \frac{N_{ij}}{N_i} \cdot \log \frac{N_{ij}}{N_i}}{\log N_i} \tag{1}$$

where N_{ij} is the number of occurrences of term t_i in document d_j, N_j is the number of documents that contains term t_i and N is the total number of documents. We removed tokens with $Q(t_i)$ measure below 0.01 or above 0.95.

We implemented term frequency inverse document frequency weighting scheme. According to the scheme we divided term frequency in a single document by the number of documents in which the term appears.

4.3 Tag recommendations

Our tag recommendation consisted of four steps. If we had a positive result in the first step then we went directly to the final fourth step.

Step One

In the first step we checked if a post is present in the BibSonomy training set or an URL of the post is among downloaded RSS entries. If the answer was true then we selected all tags attached to these resources and moved to the Step Four.

Step Two

In the second step we retrieved a group (cluster) of documents that was the most similar to the post's description or title field. The similarity was measured as a cosine of an angel between vectors $x=\{x_1,\ldots,x_n\}$ and $y=\{y_1,\ldots,y_n\}$ representing the resources in our database and the post (Eq. 2). For example, one of the posts had following title: "Attribute Grammar Based Programming and its Environment". The query consisting

of the first five informative tokens from the above title returned a cluster of four documents:

1. “A purely functional high order attribute grammar based system” from Deliciuos.com; tagged with [lisp;rag;compilers]
2. “AspectAG 0.1.1 strong typed attribute grammar implemented using type-level programming” from Reedit.com; tagged with [haskell]
3. “A 2D game programming environment based around the Ruby programming language” from Dzone.com; tagged with [frameworks,games,ruby]
4. “Attribute grammar based language extension for Java” from CiteULike.org tagged with [metaprogramming]

$$\cos(\alpha) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} \quad (2)$$

A cluster of all the retrieved posts was transferred to the next step.

Step Three

For all the posts retrieved in the second step we calculated normalized overlap score and chosen the post with the highest score. The overlap was defined as a maximum length of n-gram appearing in both posts. In order to compute the score we used all the words from title/description fields (not only the most informative tokens). The overlap score was divided by the length of title/description field of the candidate posts. For example, normalized overlap score between “Attribute Grammar Based Programming and its Environment” and “Attribute grammar based language extension for Java” equals to 3/7=0.42. The post with highest score was transferred to the final fourth step if the value of a score was greater than 0.6 threshold.

Step four

In the last step we ordered the tags of selected post according to their count in BibSonomy training set. Top five tags were selected as predictions in the Challenge.

5 Evaluation

The F1-Measure common in Information Retrieval was used to evaluate the recommendations. The precision and recall were first computed for each post in the test data by comparing the recommended tags against the tags the user has originally assigned to this post [2]. Then the average precision and recall over all posts in the test data was used to calculate the F1-Measure as f1 = (2 * precision * recall) / (precision + recall). The number of tags one can recommend was not restricted. However, the organizers regarded the first five tags only. We computed both

precision and recall measures for various levels of a threshold parameter from step three in our recommendation algorithm (Fig. 1). According to these simulations optimum level of the threshold is approximately 0.6 and yields F1-measure between 3% and 4%.

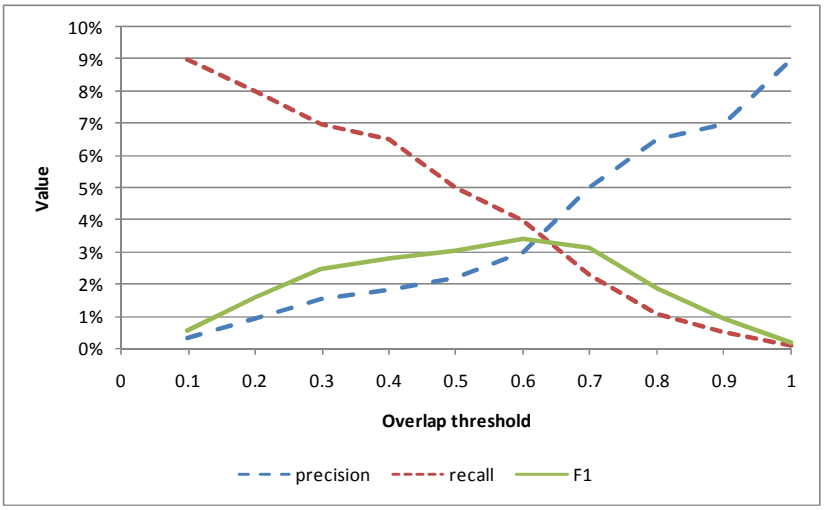

Fig. 1. Values of precision, recall and F1 measures for different levels of overlap threshold.

During the challenge we obtained overall F1-measure of 4,6%, which was slightly better than in our simulations, but incomparable to the results of the best teams.

6 Conclusions

We must admit that the way we approached the problem needs substantial computing power and disc space. Unfortunately the quality of our tag recommendations was below an average and probably this direction of research in the field of tag recommending systems is not a promising one. However we believe that there are certain situations in which best tags are not a function of words contained in title of a post and in our future research we would like to focus on such examples. Despite of unsatisfactory result in the first Task of ECML PKDD 2009 Challenge we are going to verify our recommendations within the third online recommendation task.

References

1. Ciesielski, Draminski, Klopotek, Kujawiak, Wierzchon, "On some clustering algorithms for document maps creation", in: Proceedings of the Intelligent Information Processing and Web Mining Conference (IIS:IIPWM-2005), Gdansk, Advances in Soft Computing, Springer-Verlag, (2005).
2. Jäschke, Marinho, Hotho, Schmidt-Thieme, Stumme, "Tag Recommendations in Social Bookmarking Systems", in: AI Communications , Amsterdam: IOS Press (2008)
3. Katakis, Tsoumakas, Vlahavas, "Multilabel Text Classification for Automated Tag Suggestion", in: Proceedings of ECML PKDD Discovery Challenge (RSDC08) (2008).
4. Klopotek, Wierzchon, Ciesielski, Draminski, Kujawiak, "Coexistence of Fuzzy and Crisp Concepts in Document Maps", in: Proceedings of the International Conference on Artificial Neural Networks (ICANN 2005), Lecture Notes in Artificial Intelligence, LNAI 3697, Springer-Verlag, 2005
5. Lipczak, "Tag Recommendation for Folksonomies Oriented towards Individual Users", in: Proceedings of ECML PKDD Discovery Challenge (RSDC08) (2008).
6. Tapscott, Williams, "Wikinomics", Atlantic Books, London, (2008).
7. Tatu, Srikanth, D'Silva, RSDC'08: "Tag Recommendations using Bookmark Content", in: Proceedings of ECML PKDD Discovery Challenge (RSDC08) (2008).