Learning in Efficient Tag Recommendation

Marek Lipczak
Faculty of Computer Science
Dalhousie University
Halifax, Canada, B3H 1W5
lipczak@cs.dal.ca

Evangelos Milios
Faculty of Computer Science
Dalhousie University
Halifax, Canada, B3H 1W5
eem@cs.dal.ca

ABSTRACT

The objective of a tag recommendation system is to propose a set of tags for a resource to ease the tagging process done manually by a user. Tag recommendation is an interesting and well defined research problem. However, while solving it, it is easy to forget about its practical implications. We discuss the practical aspects of tag recommendation and propose a system that successfully addresses the problem of learning in tag recommendation, without sacrificing efficiency. Learning is realized in two aspects: adaptation to newly added posts and parameter tuning. The content of each added post is used to update the resource and user profiles as well as associations between tags. Parameter tuning allows the system to automatically adjust the way tag sources (e.g., content related tags or user profile tags) are combined to match the characteristics of a specific collaborative tagging system. The evaluation on data from three collaborative tagging systems confirmed the importance of both learning methods. Finally, an architecture based on text indexing makes the system efficient enough to serve in real time collaborative tagging systems with number of posts counted in millions, given limited computing resources.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*Web-based services*

General Terms

Design, Experimentation, Performance

Keywords

tag recommendation, collaborative tagging, folksonomies

1. INTRODUCTION

Collaborative tagging systems (e.g., BibSonomy, CiteULike, Delicious, Stack Overflow) are social data repositories, in which users store and share various kinds of web resources (e.g., references to scientific literature, bookmarks, descriptions of programming problems). Each resource is entered into the system in the form of a *post*, which combines the *resource*, the *user* who is posting it and a set of *tags*. The tags position the resource in *folksonomy*, a self-emerging structure that organizes all posts gathered in the system. Folksonomy is usually represented as a tripartite graph that connects users, resources and tags. From the perspective of our work it seems reasonable to extend the graph by a fourth element – words extracted from the resource title. The title is another, more traditional form of resource description. It is also a standard element of the collaborative tagging landscape; it is present in all collaborative tagging systems studied by us. Title words are highly overlapping with tags and are definitely one of the factors influencing the choice of tags [11], hence it seems natural to take them into consideration in the tag recommendation problem. To avoid the need of extending the general formalism established for folksonomies [6], we focus on *tag profiles*, a specific data structure that can be extracted from a folksonomy. Given an element of any type (i.e., user, resource, tag, title word), the tag profile is a set of tags that co-occurred with the element in any of the posts gathered in the repository. The profile contains also the information about the number of such co-occurrences (*frequency*).

1.1 Tag recommendation task

Despite all advantages of collaborative tagging as a way of organization and retrieval of web resources, tagging is a burden for a user, who has to come up with a set of tags for each posted resource. The objective of a tag recommender is to ease this burden and propose tags that are likely to be chosen by the user. Apart from its practical importance, tag recommendation is an interesting research problem. The complex structure and the large amount of information gathered in folksonomies create lots of potential solutions, which is reflected by the variety of proposed tag recommendation systems. The opportunity to compare these approaches was recently made by the ECML/PKDD Discovery Challenge 2009 [3]. The tag recommendation systems were compared in three tasks including online evaluation, in which the tags were judged by real users. Our own system [10] submitted to the challenge achieved two first places (content-based and online recommendation) and one third place (graph-based recommendation). The key to its success was simplicity and utilization of the combined advantages of various tag sources: resource title, resource profile and user profile.

RecSys2010, September 26–30, 2010, Barcelona, Spain.

1.2 Tag recommendation – practical aspects

Encouraged by the outcome of the challenge, we looked at our system from a broader, more practical perspective. The high quality of results produced in a hermetic testing environment (hard division between training set and test set) does not yet make a system practically usable. In this paper we present a tag recommendation system which meets three objectives that, in our opinion, define a practical tag recommendation system:

Generality – each collaborative tagging system has its own specific characteristics (e.g., personal or social character of posts). These differences are likely to have impact on tagging decisions made by users, hence they had to be taken into consideration while designing a tag recommendation system. Manual tuning of system parameters has obvious limitations; therefore, the recommendation system should be able to automatically adapt to these characteristics. A learning algorithm incorporated into our system performs automatic *parameter tuning* to optimize the results based on tags entered to the system by users.

Adaptability – tag recommendation is a dynamic process. Each recommendation is instantly followed by the real tags entered by the user. This *feedback loop* constantly brings new valuable information to the system. Our system uses the content of each entered post to update the resource and user profiles as well as the associations between title words and tags. The main advantage of such *online content adaptation* is the improved quality of tags extracted from a user profile, which adapts to current interests of a user.

Efficiency – the tag recommendation system should be able to handle the large size of repositories created collaboratively by users, the open-ended vocabulary of tags and still remain efficient enough to produce results in real time. At the same time tag recommendation is just an extension to the collaborative tagging system, so it should be able to operate with limited computing resources. Our system meets all these constraints thanks to its architecture based on a text indexing engine, with an additional cache layer.

2. RELATED WORK

Research on tag recommendation systems can be divided into graph-based and content-based streams. Jäschke et al. [7] proposed a graph-based tag recommendation system based on FolkRank, an adaptation of PageRank to folksonomy graph. Given a resource-user pair the system increases their weights in the folksonomy graph and runs FolkRank to spread the weights in the graph. Tags with the highest weights are returned as recommendations. The process has to be run for each incoming post, which makes the system inefficient. Symeonidis at al. [14] used a generalization of Singular Value Decomposition to model the relations between users, resources and tags. Each of such triplets is assigned a probability value. Given a user and resource, the system simply returns the most probable tags related to them, hence the recommendation process is very efficient. The idea was extended by Randle et al. [13]. As both methods rely on tensor factorization, the efficiency and scalability of the training process is questionable. Apart from the efficiency problem, the main limitation of graph based methods is the sparsity of folksonomy graph. The commonly accepted approach to reduce this problem is graph pruning up to the point where all nodes have at least p edges (p-cores) [7]. The pruning process results in a greatly limited dataset which questions the practical usability of proposed systems. The way to bridge the gap between the need of data pruning and usability was proposed by Krestel et al. [9]. The authors applied the Latent Dirichlet Allocation to the dense core of a folksonomy to extract topics which are later used to recommend additional tags for infrequently tagged resources.

Content-based approaches extend the folksonomy graph by adding the resource content dimension. The content allows them to process posts, for which there is little information in the graph, making them more practical. Our system follows this tag recommendation approach. Among many proposed content-based recommenders we mention three that are most related to our work. Tatu el al. [15] proposed a system based on tags extracted from resource and user profile. The set of tags is extended using NLP techniques and later merged with content based tags. A system by Ju and Hwang [8] scans the content of previously tagged documents to evaluate the likelihood of a content word being used as a tag. The likelihood is later used as a score for words that occur in the content of currently posted resource. The content based tags are linearly combined with tags from resource and user profiles. Musto et al. [12] based their system on a search engine. The system retrieves resources, which content is related to the posted resource title and builds the recommendation based on prominent tags from their profiles. Specific attention is given to resources posted previously by the author of the current post.

The system presented in this paper is an extension of our submission to the ECML/PKDD Discovery Challenge 2009 [10]. The main difference from the previous system is the use of a text indexing engine, which replaced memory based data structures. The modification resolved scalability issues and made feasible the instant utilization of user feedback. The previous system was manually tuned to the challenge dataset, to improve its generality we added a learning module, which trains the system parameters to specific characteristics of processed data utilizing the feedback loop. The conceptual structure of the system remained relatively unchanged. However, the current system is simplified, as there is no need for the additional post-processing step that compensated for the low precision of tags extracted from user profile. As shown in Section 6.3 the ability of the system to adapt to newly added posts greatly improves the precision of this source of tags.

3. TAG RECOMMENDATION SYSTEM

Our system is built of five basic recommenders (Fig. 1): (a) title recommender extracts tags from resource title, (b) two profile-based recommenders are based on tags from resource and user profiles, (c) two graph-based recommenders take tags extracted from the title as input and run a *spreading activation* algorithm [2] using *title-to-tag* or *tag-to-tag* co-occurrence graphs. The main idea behind the design of the recommender is to utilize the specific advantages of each source of tags and combine the results produced by each of them. Each recommender produces a *tag recommendation set*, which is a set of proposed tags with assigned scores, $s \in [0, 1]$. The tag recommendation sets are combined by *mergers* which take two tag recommendation sets as input, linearly re-score the tags given the *merge coefficient* $p_{merge} \in [0, 1]$, representing the relative importance of both sources of tags, and merge the sets adding the scores of the

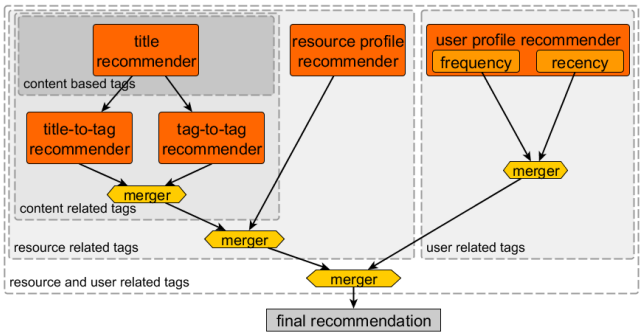


Figure 1: The tag recommendation system scheme. Tags from five basic recommenders are merged at different stages of processing.

tags that can be found in both input sets. The values of merging coefficients are learned based on real tags from user feedback. The learning approach is discussed in Section 6.2.

Processing stages.

Title recommender. The processing starts with tags extracted from the resource title. The title is a dense description of the resource, which is likely to influence the tagging decisions of users [11]. The score attached to each title word represents its use in previous posts, calculated as the the number of posts for which the word was both a title word and used as a tag, divided by the number of posts for which it was a title word. Words with low score are removed to increase precision. This step serves as a dataset and language independent stop-word filter. As the title is the only element of resource content that can be found in all collaborative tagging systems and is the most precise source of tags among content elements, it is used as the only source for **content based tags**.

Title-to-tag recommender. To maintain the consistency of their profiles, users often choose to modify the tag that describes the concept represented by the title word [11]. For example, term *network* can be modified to *networks* or more specific *social-network*. To gain access to other forms of the term and related terms the system runs spreading activation algorithm on a directed co-occurrence graph of terms, which were used as title words or tags. The weight w_{ij} of the edge between two terms is equal to the number of posts in which they occurred together – the term i as a title word and term j as a tag, divided by the total number of occurrences of the term i as a title word. Spreading activation is a technique used broadly in various areas from Artificial Intelligence to Information Retrieval. Given a directed weighted graph, it can be used to search for related elements in the graph (e.g., concepts in a semantic network [2]). Starting with a set of input nodes with assigned output values O_i, the algorithm activates nodes connected to them assigning them an input value I_j, which is calculated based on a input function. In the general case, the input is then processed to form output values that are used in the next "pulse" of the algorithm. However, to avoid overgeneralization, our system accepts the output of the first pulse as the result of the process. The output value O_i is the score attached to a word by the title recommender. The input of an activated tag I_j is used as the score of the title-to-tag recommender. The input function uses the formula for the union of probabilities of independent events (Eq. 1) to ensure that the produced scores are in $[0, 1]$ range.

$$I_j = 1 - \prod_i (1 - O_i w_{ij}) \quad (1)$$

Tag-to-tag recommender. An analogous approach can be applied to a *tag-to-tag* graph. The graph captures the relations between tags that frequently co-occur in the same posts. Unlike the title-to-tag graph, this graph is not likely to represent connections between terms that convey similar meaning because most users try to avoid redundancy while tagging. The objective of this graph is to capture hypernymic relations between tags. The system runs spreading activation on the *tag-to-tag* graph using the set of tags extracted from the title. The tags extracted from both graphs are merged producing a set of **content related tags**.

Resource profile recommender. Tags related to the resource content are extended by the tags extracted from the resource profile (all tags previously used for the resource). The score of a tag extracted from the profile is its frequency (the number of posts in which the tag was found) divided by the total number of posts of the given resource. Collaborative effort of users makes resource profile a very precise source of tags. Unfortunately, this source is rarely usable as most of the resources added to collaborative tagging systems are unique [10]. This is why in our system resource profile tags are only a supplement to the content related tags. Together they create **resource related tags**.

User profile recommender. User profile is a very rich, but noisy source of potential tag recommendations. It is likely to contain tags representing different user's interests and activities, which change dynamically. Tags frequently used in the past are not necessary a good current recommendation. To adjust to this fact, the user profile recommender uses an additional recency-based scoring scheme, to complement the frequency-based one. It produces two identical sets of tags with different scores. The sets are later merged, so the final score is a linear combination of scores proposed by both schemes. The outcome of this recommender is a set of **user related tags**. Finally, resource related tags and user related tags are merged to produce the **final recommendation**.

As the system is a hybrid tag recommender which utilizes several tag sources, there is a large space of possible combinations of basic system components. The proposed system structure is based on two objectives. First, we wanted a system with a structure that can be easily explained to the users. Second, to reduce the complexity of parameter learning, we limited the number of merger inputs to two.

4. SCALABLE SYSTEM ARCHITECTURE

The main challenge in the efficient implementation of the presented tag recommendation system is the representation of the co-occurrence graphs and the tag profiles (for resources and users). In both cases it is clear that, given the amount of data and open-ended vocabulary, they cannot be stored in operational memory. Hence, an efficient method to extract these data structures from external memory is needed. To simplify the problem, the co-occurrence graph lookup can be reduced to the tag profile lookup task. A tag profile for a title word represents all tags that co-

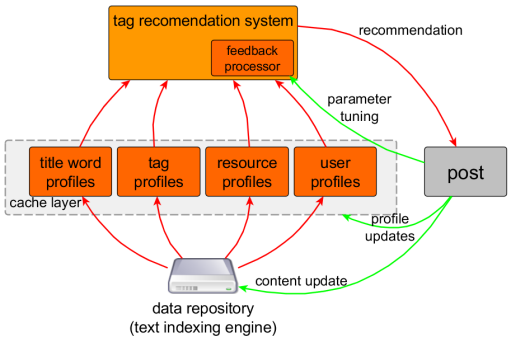


Figure 2: System architecture. The cache layer improves system efficiency. Utilization of the feedback loop allows online content adaptation.

occurred with it in any of the posts, while the frequency of co-occurrences can be used to calculate the weight of the connection. In this setting, to run a spreading activation algorithm on title-to-tag or tag-to-tag graph, given a set of tags extracted from resource title, the system has to extract tag profiles for each word and combine them. To extract a tag profile for a given post element (i.e., user, resource, tag or title word) the system uses a text indexing engine (Apache Lucene[1]). Each of the post elements is indexed separately. By accessing the Lucene index directly, the system is able to quickly retrieve a list of posts that contain a given element. As the extraction of posts is a much more time consuming task, we decided to limit the number of posts, based on which the profile is built. In experiments we used the threshold of the 1000 most recent posts that contain the element.

Cache layer.

All post elements produce heavy-tailed frequency distributions [11]. Hence, for each element type we can expect a small number of frequently used elements and a large number of infrequently used elements. In addition, it was shown that resources are often added to the system in bursts [5]. We should expect the same for user activity. All these observations suggest that an extracted tag profile is likely to be reused, so it should be cached in operational memory. The system contains a layer of caches over the Lucene index (Fig. 2). Each element type has a separate tag profile cache. The size of the cache is set individually for each element to optimize performance given limited memory resources. If the system hits the profile in the cache, it does not have to refer to the index. In case of a miss, the profile is built based on the information extracted from the index and, in the cache, the profile with the lowest value of replacement function is replaced by the new profile. The replacement policy is a combination of Least Frequently Used and Least Recently Used policies. For computational simplicity we decided to use the simplest combination of both factors (Eq. 2). In addition, we put a hard constraint on the maximal size of a profile. Tags, which are relatively rare in the profile or have not been entered into it recently, are not likely to become prominent enough in the recommendation process to reach the top of the list that is finally presented to the user, so they can be omitted. To retain potentially useful tags, the profile is implemented as a cache as well, using the same cache replacement policy (Eq. 2). All the constraints put a hard limit on the memory required to store the profiles in cache layer. To test the impact of the constraints on the quality of recommended tags, we set the tag recommendation system in the configuration used during the ECML/PKDD Discovery Challenge 2009 and evaluated it in identical test conditions, obtaining comparable results. The detailed results of this preliminary evaluation step are omitted due to space constraints.

$$rf(item) = \frac{frequency(item)}{currentTime - lastTimeUsed(item)} \qquad (2)$$

Online content adaptation and learning.

Whenever a new post is added to the system, it is stored in the text index and the tags entered by the user are used to update each of the relevant profiles in the cache layer (Fig. 2). This approach solves the cache synchronization problem without the need for additional information extraction from the index. User tags are also passed to the *feedback processing module*, which is responsible for tuning of the mergers. The module stores the input sets used by all mergers while processing the post. Given user tags, it is able to reproduce the merging process for different values of merge coefficient and learn the optimal value online.

5. DATASETS

To evaluate the presented tag recommendation system we used datasets from three collaborative tagging systems: *BibSonomy*[2] – a repository of webpage bookmarks and references to scientific publications, *Delicious*[3] – a popular social bookmarking site and *Stack Overflow*[4] – a "questions and answers" forum for programmers. The three datasets represent a broad range of collaborative tagging systems both in terms of the character of posted information and size. In this section we introduce each dataset, present general statistics (Table 1) and discuss potential sources of noise.

BibSonomy dataset – We used the BibSonomy dataset that was made available for the ECML/PKDD Discovery Challenge 2009. The dataset contains all public posts entered to the system before July 2009 as well as the metadata information of the posted resources. The tags available in the dataset are normalized – all non-alphabetical characters were removed.

Delicious dataset – Despite the fact that *Delicious* does not make its dataset publicly available for research purposes, its size and popularity makes it a frequent object of crawling. To evaluate our system we used a combination of two *Delicious* snapshots, which contain full post profiles of over 13,000 users [1] and 900,000 users [16]. As the former does not contain post time-stamps and the latter does not contain resource titles, both snapshots had to be merged. The detailed description of the merging process can be found in [11]. In the experiments we used around nine million posts for which all needed information was known. The posts were entered into *Delicious* between September 2003 and April 2007.

[1]http://lucene.apache.org/

[2]http://www.bibsonomy.org/

[3]http://delicious.com/

[4]http://stackoverflow.com/

Table 1: Dataset statistics. *Top freq* is the number of occurrences of the most frequent element.

	posts	tags			resources		users	
	total	total	distinct	top freq	distinct	top freq	distinct	top freq
BibSonomy	465,708	1,567,771	111,343	53,183	415,738	71	4,752	52,344
Delicious	8,890,876	29,807,506	601,547	399,927	4,172,960	2,673	13,079	24,176
Stack Overflow	487,829	1,467,578	32,269	60,725	487,829	1	83,693	809

Stack Overflow dataset – Stack Overflow makes all information gathered in the system publicly available monthly. The dataset we used contains posts entered into the system before February 2010. We used only the "question" posts, which were tagged by their authors. Each post contains a short description of a programming problem and its title. To keep the data structure obtained from all systems consistent, we used only the title as the post content.

5.1 Preliminaries

There are three sources of noise, which can bias the training and evaluation of tag recommendation system:

Spammers – Although *Delicious* snapshots are usually strongly contaminated by spammers, our dataset seems to be free of this problem due to the fact that one of the source datasets was crawled based on *fan* links between users, who were not likely to link to spammers. *BibSonomy* data was cleaned from spam before the challenge. *Stack Overflow* is manually controlled by the users and moderators.

Imported posts – Collaborative tagging systems allow users to import their resources from external repositories (e.g., browser bookmarks) or other collaborative tagging systems. In most cases the posts are given the same automatically created set of tags. It is especially important to remove such posts because they can strongly bias the results of tag recommendation evaluation. We eliminated posts which contained tags that likely marked the imported posts (e.g., "firefoxbookmarks"). In addition, we removed large groups of resources posted by a single user in a short time period. This technique is effective for *BibSonomy*, but not for *Delicious* data because while importing posts from browser's bookmarks folder *Delicious* copies the original time-stamp and uses sub-folders names as tags, making these posts hard to distinguish from real posts.

Tag suggestions – At the time the *BibSonomy* dataset was created the system used two basic tag recommenders interchangeably. They recommended title words or the most frequent tags from user profile. The *Delicious* posting interface displays three set of tags: a short list of tags from the resource profile, full list of user profile tags and a recommendation set which is an intersection of resource and user profile tags. *Stack Overflow* has no tag recommendation system. All systems use auto-completion which, given first letters of a tag, suggests possible choices based on the most frequent tags in the system. Recommendation and auto-completion certainly have impact on user decisions; however, the suggested tags have to be accepted by the user, hence they must be good descriptors of the resource.

As some of the bias factors cannot be eliminated it is important to evaluate tag recommenders on diverse datasets. Consistent performance should be an indication that the system is in fact producing high quality tags and not taking advantage of data aberrations.

6. EVALUATION

The system was evaluated using the three datasets to assess the impact of the learning techniques (merge coefficients learning and online content adaptation) on the quality of produced results. In addition, we ran efficiency tests to confirm the practical usability of the system.

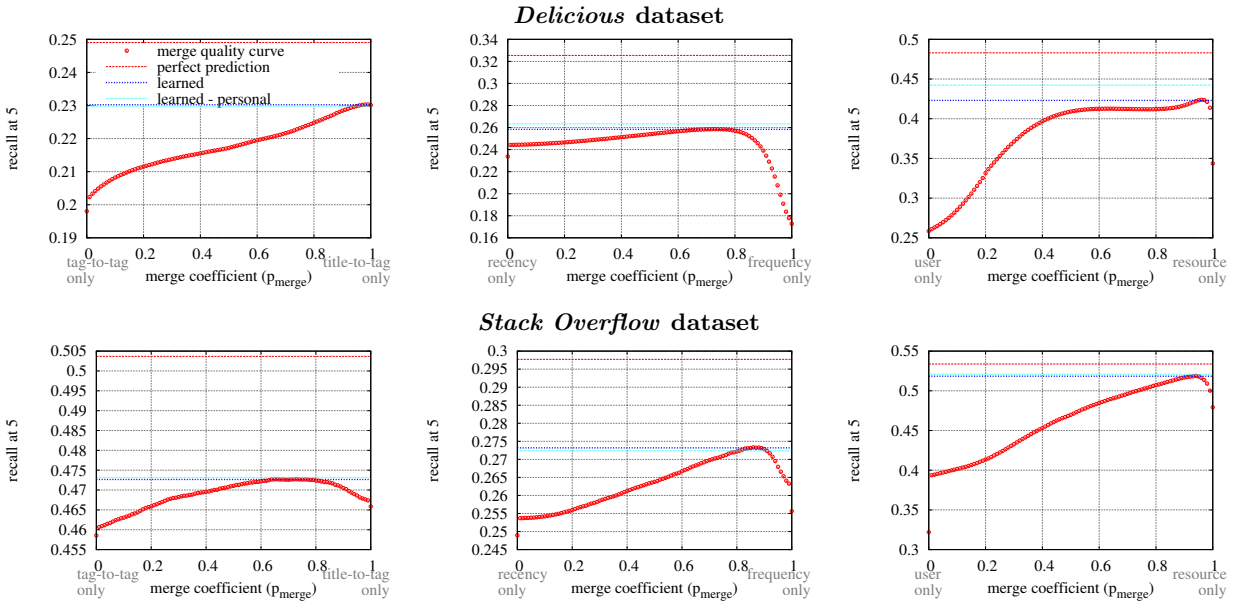
6.1 Methodology and measures

We sorted the posts chronologically and separated roughly 20% of the latest posts to test the system; 20% posts that precede them were used to tune the system parameters (to guard against overfitting). For the *BibSonomy* dataset, only the 12% of latest posts were used for testing to match the test period used in the challenge. The division was made just to clarify the presentation of the process. In the system, indexing, parameter tuning and recommendation are done simultaneously in a single run.

While evaluating the system we assumed that all and only correct tags were provided by the user. We decided to adapt this approach because of its simplicity and popularity, although it has certain limitations [4]. Following this assumption, we compared a ranked list of recommended tags with the set of real tags provided by a user. The task is analogous to standard Information Retrieval tasks, hence the basic IR measures are used to evaluate tag recommenders (e.g., precision, recall, f1 score [3], mean average precision [9]). We found that these measures are inconsistent (e.g., increasing the recall of top five recommended tags, we are likely to decrease the precision of the top tag). In our opinion, a tag recommendation system should provide the user the maximal possible number of correct tags, given a hard constraint on the number of tags recommended (usually five). If the limit of tags presented to the user is five, a system that gives the user five tags, the last three of which are correct is better than a system, which proposes two tags only, even if both of them are correct. This is why we decided to adopt *recall*@5 as the main quality criterion. We also use this measure to tune the parameters of the tag recommendation system.

6.2 Learning the merge coefficients

One of the crucial aspects of the proposed tag recommendation system is merging of tag recommendation sets that come from different components of the system. There are four processing stages, at which two input tag sets are merged (Fig. 1). Each merger has its individual merge coefficient (p_{merge}), which allows for adjustment of the relative importance of the input sets. Tags from both sets are linearly re-scored by p_{merge} and (1-p_{merge}) respectively. To understand how the choice of merge coefficient influences the quality of the result set, we discretized the range of p_{merge} into 101 values ($p_{merge} = 1$ represents tags from the first input set only and $p_{merge} = 0$ represents tags from the second input set only). We used the 80% of the posts with the



(a) content related (b) user related (c) resource and user related

Figure 3: Merge quality curves represent quality score (recall@5) as a function of merge coefficient. Various characteristics of mergers can be discovered and utilized by the proposed parameter tuning approach.

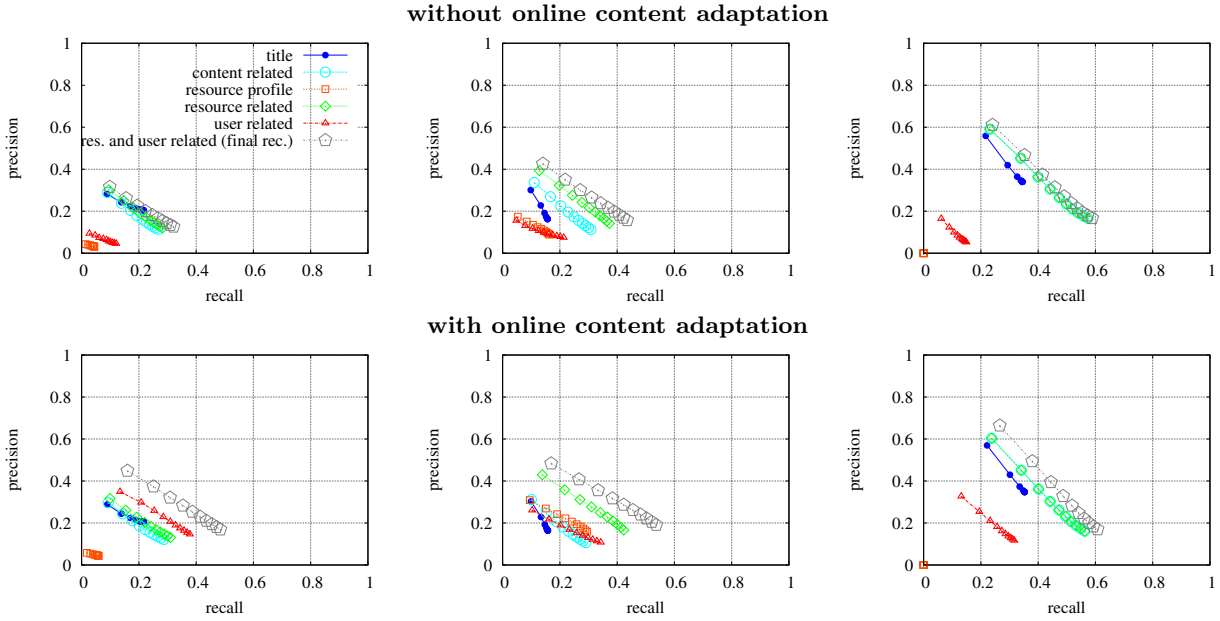
earliest time-stamps to build the folksonomy and then we iteratively, in timestamp order, added the remaining 20% of test posts to the repository calculating the average quality score (i.e., *recall*@5) for each value of p_{merge}.

We present the results for three mergers and two datasets, omitting the merger that produces resource related tags and all mergers for the *BibSonomy* dataset because of space constraints (Fig. 3). The *merge quality curve*, which is the quality score (recall@5) as a function of merge coefficient, shows that each merger has its specific characteristic, which is also dependent on the dataset. In many cases (e.g., the merger producing user related tags for *Delicious* dataset) the optimal value of p_{merge} is closer to the input set with the lower quality, which is counterintuitive. In general, the optimal value of p_{merge} tends to be close, but not equal, to one of the extremes. These facts show that the value of merge coefficient should not be related to the difference between the accuracy of input sets, which seems to be the most intuitive approach. The observations of intermediate results of the experiment suggest another solution to this problem. The shape of the merge quality curve stabilizes after some number of processed posts and the optimal value of p_{merge} stays constant or fluctuates in a limited range. The shape of the curve is smooth, in the sense that a small change in p_{merge} is not able to make a dramatic change in the tag ranking, hence it cannot affect the score. Therefore, it is possible to choose a nearly optimal value of p_{merge} from a discrete number of choices. Given these two observations, parameter tuning becomes a simple optimization task, which can be solved by recording the average quality of the merger given a limited set of p_{merge} values and use the value that has the highest quality. We used this learning approach to tune the merge coefficient to a value that overall produces results with the highest average quality. The parameter learning was done on the 20% of posts that precede the test posts. The learning method is able to discover nearly optimal values of p_{merge} in all cases ("learned" in Fig. 3).

The merge quality curve is just a general characteristic of a merger over a large number of posts. Each of these posts had a specific range of merge coefficient values, for which the optimal result could be produced. Given the real tags, it is possible to calculate the score that could be obtained assuming the perfect prediction of p_{merge} for each post ("perfect prediction" in Fig. 3). It is unlikely that this value could ever be reached, given the unpredictability of user decisions, but certainly there is room for improvement. In the future we plan to investigate the impact of personalized parameter tuning on the quality of recommendation. Preliminary results show potential improvement in most, but not all cases (e.g., user related tags for *Stack Overflow* dataset) ("learned-personal" in Fig. 3).

6.3 Online content adaptation

To observe the impact of the online content adaptation on the results and provide a baseline for the system we ran a series of experiments in which this feature was turned off. The parameters of the system were re-trained to tune it to the new conditions. The adaptation improves the results of the recommendations for all tested datasets (Table 2). The statistical significance of the difference was confirmed by a Wilcoxon signed-rank test ($P < 0.001$). We present the plots of recall and precision for each stage of the recommendation process, without and with adaptation, to show how they contributed to the final result (Fig. 4). Although the importance of each source of tags is specific to the dataset, in all cases the successive stages of the recommendation process are able to improve the quality of the recommended tags. The results produced at each stage of the system can be used as additional baselines for the final recommendation. The *recall*@5 value of the system with online content



(a) *BibSonomy* dataset (b) *Delicious* dataset (c) *Stack Overflow* dataset

Figure 4: Precision/recall plots for $k \in [1, 10]$ top tags produced at each stage of tag recommendation process. Comparison between the system without and with online content adaptation shows the positive impact of adaptation on the quality of user related tags, resource related tags and the final recommendation.

adaptation is significantly better than any of its baselines. The online content adaptation has a clear impact on the relative importance of different tag sources. For all datasets the largest improvement is noticed for the user related tags. It is specifically important for *Stack Overflow* data, which is completely unbiased by imports and recommendations. Adaptation allows the system to extend the repository of user related tags by tags that describe user's recent interests. Large improvement can also be noticed for tags extracted from resource profile for the *Delicious* dataset (Fig. 4(b)). It seems that the availability of a large number of newly added posts allows resource profiles to overcome the problem of cold start [9] – the noisiness of profiles of infrequently posted resources. Finally, the adaptation seems to have little or no impact on the content related tags extracted from the co-occurrence graphs. The associations between tags are well established at the time of the evaluation and they are not changed by the adapted content. In this case the adaptation is likely to be useful in the early stage of folksonomy formulation only.

6.4 Efficiency

We based the evaluation of system efficiency solely on the *Delicious* dataset, because of its size. We also decided not to use a server machine, as it is hard to control the impact of its configuration (e.g., RAID, hard-drive cache) on the processing efficiency. The tests were performed on a personal computer with a 32-bit, dual-core, 1.73 GHz CPU and 7200 RPM, 16 MB Cache, SATA (3 GB/s) hard drive and with the system restricted to 1.5 GB of memory. We iteratively added the posts to the index in chronological order, restarting the system twice: after indexing 60% of posts (over 5 million) and 80% of posts (over 7 million). After the restart, full recommendation was run for 8 hours. We logged the

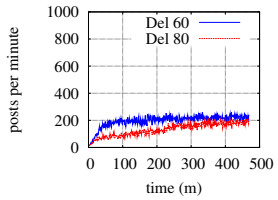
Table 2: Recall@5 for the final recommendation. For all data online content adaptation gives statistically significant improvement ($P < 0.001$, Wilcoxon test).

dataset	**no** adapt.	**with** adapt.	**increase**
BibSonomy	0.247	**0.387**	0.140
Delicious	0.342	**0.423**	0.081
Stack Overflow	0.490	**0.518**	0.028

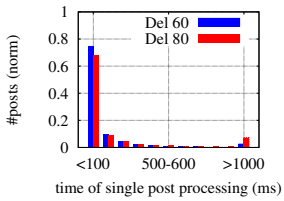
number of posts processed per minute (Fig. 5(a)) and the recommendation time for each processed post (Fig. 5(b)). To demonstate their impact on efficiency, the caches were not pre-fetched. With 5 million posts in the index the system needed around 50 minutes to fill the caches with useful profiles. After that it demonstrated stable throughput of around 200 posts processed per minute (which adds up to 288,000 posts per day). As shown in the histogram of single post recommendation times, most of the recommendations were finished in less than 100 milliseconds, which makes processing unnoticable for a user. Both results are well above the practical usability level for collaborative tagging systems of a size of BibSonomy or CiteULike. Comparison of the results recorded after 60% and 80% of posts shows only minor decrease in performace. The investigation of system's scalability limit has to be deferred until larger datasets are available to us.

7. CONCLUSIONS AND FUTURE WORK

Although tag recommendation is a very interesting theoretical problem, it has also great practical importance, which should be taken into account while proposing a solution. We



(a) Number of posts processed per minute (includes indexing).



(b) Histogram of recommendation times per single post.

Figure 5: Processing efficiency. Both post throughput and recommendation times are well above usability levels for practical tag recommendation.

presented a concept of a tag recommendation system together with its architecture based on a text indexing engine. The system successfully addresses three important practical aspects of the tag recommendation task: generality, adaptability and efficiency. Automatic parameter tuning makes the system generally applicable to a broad range of collaborative tagging systems. This is an important feature as all systems have specific characteristics which determine the most useful sources of potential tag recommendation and ratios, by which they should be combined. The proposed system takes advantage of the feedback loop, which makes the real tags entered by the user available instantly after the recommendation. The adaptation to newly added posts dramatically changes the character of the tag recommendation problem. It reveals the importance of the user profile as the source of high quality tags, whereas, the lack of adaptation forces the system to rely mostly on the resource title, a precise but limited source of tags. Finally, the system architecture based on a text indexing engine with an additional cache layer makes the system efficient enough to process large collaborative tagging systems, counted in millions of posts, in real time.

In the future we plan to continue the work on personalized recommendation. We also want to investigate the use of a database system instead of the underlying text indexing engine. Although the character of the problem intuitively makes it more suitable for a database, preliminary experiments with Apache Derby and Berkeley DB resulted in much higher storage space usage and extended processing time.

The presented system operates as a part of *BibSonomy*. The system, its source code and sample experiment are available at: `http://www.cs.dal.ca/~lipczak/tr.php`

8. ACKNOWLEDGMENTS

We would like to thank Tom Crecelius and Robert Wetzker as well as the administrators of BibSonomy and Stack Overflow for making their datasets available to us. The research was funded by the Natural Sciences and Engineering Research Council of Canada and the MITACS NCE.

9. REFERENCES

[1] M. Bender, T. Crecelius, M. Kacimi, S. Michel, T. Neumann, J. X. Parreira, R. Schenkel, and G. Weikum. Exploiting social relations for query expansion and result ranking. In *Data Engineering for Blogs, Social Media, and Web 2.0, ICDE 2008 Workshops*, pages 501–506, 2008.

[2] P. R. Cohen and R. Kjeldsen. Information retrieval by constrained spreading activation in semantic networks. *Inf. Process. Manage.*, 23(4):255–268, 1987.

[3] F. Eisterlehner, A. Hotho, and R. Jäschke, editors. *ECML PKDD Discovery Challenge 2009 (DC09)*, volume 497 of *CEUR-WS.org*, 2009.

[4] J. Gemmell, M. Ramezani, T. Schimoler, L. Christiansen, and B. Mobasher. The impact of ambiguity and redundancy on tag recommendation in folksonomies. In *RecSys '09: Proc. the Third ACM Conference on Recommender Systems*, pages 45–52. ACM, 2009.

[5] S. A. Golder and B. A. Huberman. Usage patterns of collaborative tagging systems. *J. Inf. Sci.*, 32(2):198–208, 2006.

[6] A. Hotho, R. Jäschke, C. Schmitz, and G. Stumme. Trend detection in folksonomies. *Semantic Multimedia*, pages 56–70, 2006.

[7] R. Jäschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag recommendations in folksonomies. *Knowledge Discovery in Databases: PKDD 2007*, pages 506–514, 2007.

[8] S. Ju and K.-B. Hwang. A weighting scheme for tag recommendation in social bookmarking systems. In *Proc. the ECML/PKDD 2009 Discovery Challenge Workshop*, pages 109–118, 2009.

[9] R. Krestel, P. Fankhauser, and W. Nejdl. Latent dirichlet allocation for tag recommendation. In *RecSys '09: Proc. the Third ACM Conference on Recommender Systems*, pages 61–68. ACM, 2009.

[10] M. Lipczak, Y. Hu, Y. Kollet, and E. Milios. Tag sources for recommendation in collaborative tagging systems. In *Proc. the ECML/PKDD 2009 Discovery Challenge Workshop*, pages 157–172, 2009.

[11] M. Lipczak and E. Milios. The impact of resource title on tags in collaborative tagging systems. In *HT'10: Proc. the 21th ACM Conference on Hypertext and Hypermedia*, pages 179–188. ACM, 2010.

[12] C. Musto, F. Narducci, M. de Gemmis, P. Lops, and G. Semeraro. STaR: a social tag recommender system. In *Proc. the ECML/PKDD 2009 Discovery Challenge Workshop*, pages 215–227, 2009.

[13] S. Rendle, L. B. Marinho, A. Nanopoulos, and L. Schmidt-Thieme. Learning optimal ranking with tensor factorization for tag recommendation. In *KDD '09: Proc. the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 727–736. ACM, 2009.

[14] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. Tag recommendations based on tensor dimensionality reduction. In *RecSys '08: Proc. the 2008 ACM Conference on Recommender Systems*, pages 43–50. ACM, 2008.

[15] M. Tatu, M. Srikanth, and T. D'Silva. Rsdc'08: Tag recommendations using bookmark content. In *Proc. the ECML/PKDD 2008 Discovery Challenge Workshop*, pages 96–107, 2008.

[16] R. Wetzker, C. Zimmermann, and C. Bauckhage. Analyzing social bookmarking systems: A del.icio.us cookbook. In *Mining Social Data (MSoDa) Workshop Proceedings*, pages 26–30, 2008.