Demand-Driven Tag Recommendation

Guilherme Vale Menezes[1], Jussara M. Almeida[1], Fabiano Belém[1], Marcos André Gonçalves[1], Anísio Lacerda[1], Edleno Silva de Moura[2], Gisele L. Pappa[1], Adriano Veloso[1], and Nivio Ziviani[1]

[1] Universidade Federal de Minas Gerais, Belo Horizonte, Brazil
[2] Universidade Federal do Amazonas, Manaus, Brazil

Abstract. Collaborative tagging allows users to assign arbitrary keywords (or tags) describing the content of objects, which facilitates navigation and improves searching without dependence on pre-configured categories. In large-scale tag-based systems, tag recommendation services can assist a user in the assignment of tags to objects and help consolidate the vocabulary of tags across users. A promising approach for tag recommendation is to exploit the co-occurrence of tags. However, these methods are challenged by the huge size of the tag vocabulary, either because (1) the computational complexity may increase exponentially with the number of tags or (2) the score associated with each tag may become distorted since different tags may operate in different scales and the scores are not directly comparable. In this paper we propose a novel method that recommends tags on a demand-driven basis according to an initial set of tags applied to an object. It reduces the space of possible solutions, so that its complexity increases polynomially with the size of the tag vocabulary. Further, the score of each tag is calibrated using an entropy minimization approach which corrects possible distortions and provides more precise recommendations. We conducted a systematic evaluation of the proposed method using three types of media: audio, bookmarks and video. The experimental results show that the proposed method is fast and boosts recommendation quality on different experimental scenarios. For instance, in the case of a popular audio site it provides improvements in precision (p@5) ranging from 6.4% to 46.7% (depending on the number of tags given as input), outperforming a recently proposed co-occurrence based tag recommendation method.

1 Introduction

The act of associating keywords (tags) with objects is referred to as tagging. It has become a popular activity with the advent of Web 2.0 applications, which facilitated and stimulated end users to contribute with content created by themselves. This content, typically multimedia (e.g. audio, image, video), brings challenges to current information retrieval (IR) methods, not only due to the scale of the collections and speed of update, but also due to the (usually) poor quality of user-generated material. Tags offer a good alternative for personal or community-based organization, dissemination, and retrieval of Web 2.0 content. In fact, recent studies have demonstrated that tags are among the best textual features to be exploited by IR services, such as automatic classification [5].

In this context, tag recommendation services aims at improving the description of an object by suggesting tags that would correctly and more completely describe its

content. Possible sources of information for this recommendation task could be: (1) tags previously associated with objects in the collection; and (2) the textual content of other features (e.g., title, description, user comments) associated with the object for which the recommendation is expected. While in case of (2) there could be more input for the sake of recommendation, problems such as the lack of standardization in content format and the presence of noisy content (e.g., non-existing words [23]) benefit the use of recommendation methods that exploit solely tag co-occurrence information [6, 20].

In this paper we present a new co-occurrence based algorithm to recommend tags on a demand-driven fashion according to an initial set of tags applied to an object. We address the tag recommendation problem from the community-oriented perspective, in the sense that the set of recommended tags is useful both to the user and to a larger community of users. Typically, an initial set of tags $\mathcal{I}_o$ associated with an object o is provided to the recommendation method, which outputs a set of related tags $\mathcal{C}_o$, where $\mathcal{I}_o \cap \mathcal{C}_o = \emptyset$.

Since there is an unlimited number of ways to describe an object by choosing arbitrary keywords, an object may have several different tags associated with it. Our strategy is to treat each possible tag already existent in the system as a class for the object, modeling the problem of recommending tags as a *multi-label classification* problem. This approach is challenging since the vocabulary of tags in systems such as *YouTube*[3], *Delicious*[4] and *LastFM*[5] is large and current automatic classifiers cannot deal well with problems with many thousands of classes.

We present a Lazy Associative Tag REcommender, referred to as LATRE from now on, which has been developed to deal with large-scale problems with thousands of tags (or classes). LATRE exploits co-occurrence of tags by extracting association rules on a demand-driven basis by. These rules are the basic components of the classification model produced by LATRE. In this case, rules have the form $X \rightarrow y$, where X is a set of tags and y is the predicted tag.

Rule extraction is a major issue for co-occurrence based recommendation methods, such as [7, 20], since the number of extracted rules may increase exponentially with the number of tags. LATRE, on the other hand, extracts rules from the training data on the fly, at recommendation time. The algorithm projects the search space for rules according to qualitative information present in each test instance, allowing the generation of more elaborated rules with efficiency. In other words, LATRE projects/filters the training data according to the tags in $\mathcal{I}_o$, and extracts rules from this projected data. This ensures that only rules that carry information about object o (i.e., a test object) are extracted from the training data, drastically bounding the number of possible rules. In fact, the computational complexity of LATRE is shown to increase polynomially with the number of tags in the vocabulary. This efficiency enables LATRE to explore portions of the rule space that could not be feasibly explored by other methods (i.e., more "complex" rules).

After a set of rules is extracted for object o, LATRE uses them to sort/rank candidate tags that are more likely to be correctly associated with this object. Each extracted rule

[3] www.youtube.com
[4] www.delicious.com
[5] www.lastfm.com

$X \xrightarrow{\theta} y$ is interpreted as a vote given for tag y and the weight of the vote is given by θ, which is the conditional probability of object o being associated with tag y given that o contains all tags in X. Weighted votes for each tag are added, and tags that scored higher are placed on the beginning of the ranking. Usually, there are many candidate tags, and thus properly ranking them is also a difficult issue, since different candidate tags may operate in different scales (i.e., a popular tag may receive a large number of "weak" votes, and this tag is likely to be placed before a specific tag which received a small number of "strong" votes). In order to enforce all tags to operate in the same scale, so that they can be directly compared, we employed an entropy-minimization calibration approach to correct possible distortions in the scores of candidate tags.

Experimental results obtained from collections crawled from Delicious, LastFM and YouTube show that LATRE recommends tags with a significantly higher precision in all collections when compared to a recent co-occurrence based baseline proposed in [20]. The study of the effectiveness of LATRE on three different collections corresponding to different media types (i.e. Web pages, audio and video) is also an important contribution of our work, as most of the methods in the literature are tested only with one collection and one media type. Depending on the number of tags provided as input to LATRE, it obtained gains in precision (p@5) ranging from 10.7% to 23.9% for Delicious, from 6.4% to 46.7% for LastFM, and from 16.2% to 33.1% for YouTube.

This paper is organized as follows. In Section 2 we cover related work. In Section 3 we provide an in-depth description of our proposed method, whereas in Section 4 we describe our collections and discuss our experiments and results. Finally, in Section 5 we offer conclusions and possible directions for future work.

2 Related Work

Previous work has used co-occurrence information to expand an initial set of tags I_o associated with an object o with related tags [6, 7, 9, 20, 25, 26]. Heymann et al. [7] use association rules to expand a set of tags of an object. A confidence threshold is set to limit the rules used in expansion. Experiments have shown that the expanded set of tags can increase the recall of their tag recommendation method by 50% while keeping a high precision. In [20], authors use conditional probability and the Jaccard coefficient to recommend tags on Flickr. They aggregate these co-occurrence measures for all tags in I_o to obtain the set of related tags. The problem they studied is very similar to ours, i.e., output a ranking of tags using only community knowledge (not personal information). Another related work is presented in [6]. They study the problem of making personal recommendations using the history of all tags a user has applied in the past. The authors use a Naive Bayes classifier to learn a model specific for each user. They concluded that adding personal history can improve the effectiveness of co-occurrence tag-recommendation. Krestel et al. [9] use Latent Dirichlet Allocation to expand the set of tags of objects annotated by only a few users (the cold start problem). After a comparison with [7] they concluded that their method is more accurate and yields more specific recommendations. In [25], Wu et al. model tag recommendation in Flickr as a learn to rank problem using RankBoost. They use tag co-occurrence measures and image content information as features for the learn to rank algorithm.

A content-based approach to the expansion of a tag set is described in [19]. The authors use overlapping and duplicated content in videos to create links in a graph and propagate tags between similar videos. They use the expanded set for video clustering and classification, obtaining significant improvements.

Tag co-occurrence has also been used in contexts different from tag expansion. For example, the tag recommendation algorithm described in [26] uses co-occurrence information to select a small set of informative tags from the tags collectively used to describe an object. They give higher values to tags that have been used together by the same user (complementary tags) and lower value to different tags that have been used by different users to describe the same object (tags that describe the same concept). Another example is the identification of ambiguous tags using co-occurrence distributions. The method in [24] suggests tags that help to disambiguate the set of tags previously assigned to an object. The key observation is that very different distributions of co-occurring tags arise after adding each ambiguous tag. A third example is tag ranking [11], in which the authors used a random walk process based on tag co-occurrence information to generate a ranking, which is shown to improve image search, tag recommendation and group recommendation. Finally, tag translation using tag co-occurrence is described in [19]. The authors created a tag co-occurrence graph and used network similarity measures to find candidates for translation.

Classification algorithms have been used in tag recommendation in the past. In [7], the authors use a SVM classifier to predict whether a tag t is associated with an object o based on the textual content of o. Their approach is not scalable to many thousands of tags since they need to build a binary classifier for each tag. In their experiments they use only the top 100 tags of their Delicious collection. A second approach is used by [22]. They group objects into clusters using a graph partitioning algorithm, and they train a Naive Bayes classifier using the generated clusters as classes. When a new object arrives, their method classifies the object into one of the clusters, and use the tags of the cluster to generate a ranking of candidate tags. A third way is to consider each tag as a class, and model tag recommendation as a multi-label classification problem. In this case, tags are used as both features and labels, and the classification algorithm must be able to deal with a very large number of classes. This approach is discussed in [6, 21] and used in this work. While [6] uses a Naive Bayes classifier and [21] proposes a multi-label sparse Gaussian process classification to model tag recommendation, our work is based on associative classification. Between the many applications of social tags we can cite page clustering [12, 15], enhancing item search [3, 4, 16], enhancing item recommendation [8, 17, 18], uncovering user-induced relations among items [2], clustering users in communities [10] and automatic ontology generation [13].

3 Associative Recommendation

We have essentially modeled the tag recommendation task as a multi-label classification problem. In this case, we have as input the *training data* (referred to as $\mathcal{D}$), which consists of objects of the form $d = <\mathcal{I}_d, \mathcal{Y}_d>$, where both $\mathcal{I}_d$ and $\mathcal{Y}_d$ are sets of tags, and initially, $\mathcal{I}_d$ contains all tags that are associated with object d, while $\mathcal{Y}_d$ is empty. The *test set* (referred to as $\mathcal{T}$) consists of objects of the form $t = <\mathcal{I}_t, \mathcal{Y}_t>$, where both

$\mathcal{I}_t$ and $\mathcal{Y}_t$ are sets of tags associated with object t. However, while tags in $\mathcal{I}_t$ are known in advance, tags in $\mathcal{Y}_t$ are unknown, and functions learned from $\mathcal{D}$ are used to predict (or recommend) tags that are likely to be in $\mathcal{Y}_t$ based on tags in $\mathcal{I}_t$. We developed the LATRE method within this model. Recommendation functions produced by LATRE exploit the co-occurrence of tags in $\mathcal{D}$, which are represented by association rules [1].

Definition 1. *An association rule is an implication $\mathcal{X} \xrightarrow{\theta} y$, where the antecedent $\mathcal{X}$ is a set of tags, and the consequent y is the predicted tag. The domain for $\mathcal{X}$ is denoted as $\mathcal{I}$=$\{\mathcal{I}_1 \cup \mathcal{I}_2 \cup \ldots \cup \mathcal{I}_m\}$ (i.e., $\mathcal{X} \subseteq \mathcal{I}$), where m=$|\mathcal{D}| + |\mathcal{T}|$. The domain for y is $\mathcal{Y}$=$\{\mathcal{Y}_1 \cup \mathcal{Y}_2 \cup \ldots \cup \mathcal{Y}_m\}$ (i.e., $y \in \mathcal{Y}$). The size of rule $\mathcal{X} \rightarrow y$ is given by the number of tags in the antecedent, that is $|\mathcal{X}|$. The strength of the association between $\mathcal{X}$ and y is given by θ, which is simply the conditional probability of y being in $\mathcal{Y}_o$ given that $\mathcal{X} \subseteq \mathcal{I}_o$.*

We denote as $\mathcal{R}$ a rule-set composed of rules $\mathcal{X} \xrightarrow{\theta} y$. Next we present the major steps of LATRE: rule extraction, tag ranking, and calibration.

3.1 Demand-Driven Rule Extraction

The search space for rules is huge. Existing co-occurrence based recommendation methods, such as the ones proposed in [20], impose computational cost restrictions during rule extraction. A typical strategy to restrict the search space for rules is to prune rules that are not sufficiently frequent (i.e., minimum support). This strategy, however, leads to serious problems because the vast majority of the tags are usually not frequent enough. An alternate strategy is to extract only rules $\mathcal{X} \rightarrow y$ such that $|\mathcal{X}| \leq \alpha_{max}$, where α_{max} is a pre-specified threshold which limits the maximum size of the extracted rules. However, in actual application scenarios, methods such as [20] are only able to efficiently explore the search space for rules if α_{max}=1. When the value of α_{max} is increased, the number of rules extracted from $\mathcal{D}$ increases at a much faster pace (i.e., there is a combinatorial explosion).

The obvious drawback of this approach (i.e., α_{max}=1) is that more complex rules (i.e., rules with $|\mathcal{X}|$>1) will be not included in $\mathcal{R}$. These complex rules may provide important information for the sake of recommendation, and thus, the immediate question is how to efficiently extract rules from $\mathcal{D}$ using arbitrary values of α_{max}. One possible solution for this question is to extract rules on a demand-driven basis, but before discussing this solution we need to present the definition of useful association rules.

Definition 2. *A rule $\{\mathcal{X} \rightarrow y\} \in \mathcal{R}$ is said to be useful for object t =<$\mathcal{I}_t, \mathcal{Y}_t$> if $\mathcal{X} \subseteq \mathcal{I}_t$. That is, rule $\{\mathcal{X} \rightarrow y\} \in \mathcal{R}$ can only be used to predict tags for object $t \in \mathcal{T}$ if all tags in $\mathcal{X}$ are included in $\mathcal{I}_t$.*

The idea behind demand-driven rule extraction is to extract only those rules that are useful for objects in $\mathcal{T}$. In this case, rule extraction is delayed until an object t =<$\mathcal{I}_t, \mathcal{Y}_t$> is informed. Then, tags in $\mathcal{I}_t$ are used as a filter which configures $\mathcal{D}$ in a way that only rules that are useful for object t can be extracted. This filtering process produces a projected training data, $\mathcal{D}_t$, which is composed of objects of the form d^t =< $\mathcal{I}_d^t, \mathcal{Y}_d^t$ >, where $\mathcal{I}_d^t$=$\{\mathcal{I}_d \cap \mathcal{I}_t\}$ and $\mathcal{Y}_d^t$=$\{\mathcal{I}_d - \mathcal{I}_d^t\}$.

The process is illustrated in Tables 1 and 2. There are 5 objects $\{<\mathcal{I}_{d_1}, \mathcal{Y}_{d_1}>; <\mathcal{I}_{d_2}, \mathcal{Y}_{d_2}>; <\mathcal{I}_{d_3}, \mathcal{Y}_{d_3}>; <\mathcal{I}_{d_4}, \mathcal{Y}_{d_4}>; <\mathcal{I}_{d_5}, \mathcal{Y}_{d_5}>\}$ in $\mathcal{D}$, and one object $<\mathcal{I}_{t_1}, \mathcal{Y}_{t_1}>$ in $\mathcal{T}$. Table 2 shows $\mathcal{D}$ after being projected according to $\mathcal{I}_{t_1}$. In this case, $\mathcal{I}_{d_1}^{t_1}$=$\{\mathcal{I}_{t_1} \cap \mathcal{I}_{d_1}\}$=$\{$unicef$\}$, and $\mathcal{Y}_{d_1}^{t_1}$=$\{\mathcal{I}_{d_1} - \mathcal{I}_{d_1}^{t_1}\}$=$\{$children, un, united, nations$\}$. The same procedure is repeated for the remaining objects in $\mathcal{D}$, so that $\mathcal{D}^{t_1}$ is finally obtained. For an arbitrary object $t \in \mathcal{T}$, we denote as $\mathcal{R}_t$ the rule-set extracted from $\mathcal{D}_t$.

Table 1. Training data and test set.

		$\mathcal{I}$	$\mathcal{Y}$
	d_1	unicef children un united nations	$\emptyset$
	d_2	un climatechange summit environment	$\emptyset$
$\mathcal{D}$	d_3	climatechange islands environment	$\emptyset$
	d_4	children games education math	$\emptyset$
	d_5	education children unicef job	$\emptyset$
$\mathcal{T}$	t_1	unicef education haiti	?

Table 2. Projected training data for object t_1.

		$\mathcal{I}^t$	$\mathcal{Y}^t$
	$d_1^{t_1}$	unicef	children un united nations
$\mathcal{D}_{t_1}$	$d_4^{t_1}$	education	children games math
	$d_5^{t_1}$	unicef education	children job

Lemma 1. *All rules in $\mathcal{R}_t$ are useful for object t =$<\mathcal{I}_t, \mathcal{Y}_t>$.*

Proof. Let $\mathcal{X} \rightarrow y$ be an arbitrary rule in $\mathcal{R}_t$. In this case, $\mathcal{X} \subseteq \mathcal{I}_t$. Thus, according to Definition 2, this rule must be useful for object t. □

Lemma 1 states that any rule extracted from $\mathcal{D}_{t_1}$ (i.e., Table 2) is useful for object t_1. Examples of rules extracted from $\mathcal{D}_{t_1}$ include:

- unicef $\xrightarrow{\theta=1.00}$ children
- $\{$unicef$\wedge$education$\}$ $\xrightarrow{\theta=1.00}$ children
- education $\xrightarrow{\theta=0.50}$ math

Since $\{$unicef,education$\} \subseteq \mathcal{I}_{t_1}$, all these rules are useful for object t_1. An example of rule that is useless for object t_1 is "climatechange $\xrightarrow{\theta=1.00}$ environment", and it is easy to see that this rule cannot be extracted from $\mathcal{D}_{t_1}$, since tag "climatechange" is not present in $\mathcal{D}_{t_1}$.

The next theorem states that LATRE efficiently extracts rules from $\mathcal{D}$. The key intuition is that LATRE works only on tags that are known to be associated to each other, drastically narrowing down the search space for rules.

Theorem 1. *The complexity of LATRE increases polynomially with the number of tags in the vocabulary.*

Proof. Let n be the number of tags in the vocabulary. Obviously, the number of possible association rules that can be extracted from $\mathcal{D}$ is 2^n. Also, let $t = <\mathcal{I}_t, \mathcal{Y}_t>$ be an arbitrary object in $\mathcal{T}$. Since $\mathcal{I}_t$ contains at most k tags (with $k \ll n$), any rule useful for object t can have at most k tags in its antecedent. Therefore, the number of possible rules that are useful for object t is $(n\text{-}k) \times (k + \binom{k}{2} + \ldots + \binom{k}{k}) = O(n^k)$ (since $k \ll n$), and thus, the number of useful rules increases polynomially in n. Since, according to Lemma 1, LATRE extracts only useful rules for objects in $\mathcal{T}$, then the complexity of LATRE also increases polynomially in n. □

An important practical aspect of LATRE is that the projection of the dataset (as shown in the examples in Tables 1 and 2) greatly reduces the size of both n and k, since we only consider candidate tags that co-occur at least once with any tag in the test object (n is reduced), while the size of the test object is small in practice (k is reduced). For instance, in Tables 1 and 2 we have $k_1 = 1$, $k_2 = 0$, $k_3 = 0$, $k_4 = 1$ and $k_5 = 2$ for the projected dataset. Therefore, the average k per object is $(1 + 0 + 0 + 1 + 2)/5 = 4/5 = 0.8$. This number is much smaller than the upper bound in the number of tags in an object, which is 5 in the example.

3.2 Tag Ranking

In order to select candidate tags that are more likely to be associated with object $t \in \mathcal{T}$, it is necessary to sort tags by combining rules in $\mathcal{R}_t$. In this case, LATRE interprets $\mathcal{R}_t$ as a poll, in which each rule $\mathcal{X} \xrightarrow{\theta} y \in \mathcal{R}_t$ is a vote given by tags in $\mathcal{X}$ for candidate tag y. Votes have different weights, depending on the strength of the association they represent (i.e., θ). The weighted votes for each tag y are summed, giving the score for tag y with regard to object t, as shown in Equation 1 (where y_i is the i-th candidate tag, and $\theta(\mathcal{X} \rightarrow y_i)$ is the value θ assumes for rule $\mathcal{X} \rightarrow y_i$):

$$s(t, y_i) = \sum \theta(\mathcal{X} \rightarrow y_i), \text{ where } \mathcal{X} \subseteq \mathcal{I}_t \tag{1}$$

Thus, for an object t, the score associated with tag y_i is obtained by summing the θ values of the rules predicting y_i in $\mathcal{R}_t$. The likelihood of t being associated with tag y_i is obtained by normalizing the scores, as expressed by the function $\hat{p}(y_i|t)$, shown in Equation 2:

$$\hat{p}(y_i|t) = \frac{s(t, y_i)}{\sum_{j=0}^{n} s(t, y_j)} \tag{2}$$

Candidate tags for object t are sorted according to Equation 2, and tags appearing first in the ranking are finally recommended.

3.3 Calibration

According to Equation 1, the score associated with a tag is impacted by two characteristics: (1) the number of votes it receives, and also (2) the strength of these votes. While

both characteristics are intuitively important to estimate the likelihood of association between tags and objects, it may be difficult to decide which one is more important. In some cases, the scores associated with different tags cannot be directly compared, because they operate in different scales (i.e., the score associated with popular tags are likely to be higher than the scores associated with specific tags, simply because they receive a large number of votes). This means that the same value of score can be considered either high or low, depending on the tag.

An approach for this problem would be to inspect the expected likelihood $\hat{p}(y|t)$, in order to make scores associated with different tags directly comparable. The obvious problem with this approach is that the correct value for $\hat{p}(y|t)$ is not known in advance, since $t \in \mathcal{T}$, and thus we cannot verify if $y \in \mathcal{Y}_t$. An alternative is to use a validation set (denoted as $\mathcal{V}$), which is composed of objects of the form $v = <\mathcal{I}_v, \mathcal{Y}_v>$, where both $\mathcal{I}_v$ and $\mathcal{Y}_v$ are sets of tags associated with object v, and $\{\mathcal{I}_v \cap \mathcal{Y}_v\}=\emptyset$. That is, the validation set essentially mimics the test set, in the sense that $\mathcal{Y}_v$ is not used for the sake of producing rules, but only to find possible distortions in the value of $\hat{p}(y|v)$.

The key intuition of our approach is to contrast values of $\hat{p}(y|v)$ for which $y \notin \mathcal{Y}_v$, and values of $\hat{p}(y|v)$ for which $y \in \mathcal{Y}_v$. In an ideal case, for a given tag y, there is a value f_y such that:

- if $\hat{p}(y|v) \leq f_y$, then $y \notin \mathcal{Y}_v$
- if $\hat{p}(y|v) > f_y$, then $y \in \mathcal{Y}_v$

Once f_y is calculated, it can be used to determine whether a certain value of $\hat{p}(y|v)$ is low or high, so that the score associated with different tags can be directly compared. However, more difficult cases exist, for which it is not possible to obtain a perfect separation in the space of values for $\hat{p}(y|v)$. Thus, we propose a more general approach to calculate f_y. The basic idea is that any value for f_y induces two partitions over the space of values for $\hat{p}(y|v)$ (i.e., one partition with values that are lower than f_y, and another partition with values that are higher than f_y). Our approach is to set f_y with the value which minimizes the average entropy of these two partitions. In the following we present the basic definitions in order to detail this approach.

Definition 3. *Let y be an arbitrary tag, and let $v = <\mathcal{I}_v, \mathcal{Y}_v>$ be an arbitrary object in $\mathcal{V}$. In this case, let $o(y, v)$ be a binary function such that:*

$$o(y, v) = \begin{cases} 1 \text{ if } y \in \mathcal{Y}_v \\ 0 \text{ otherwise} \end{cases}$$

Definition 4. *Consider $\mathcal{O}(y)$ a list of pairs $<o(y, v), \hat{p}(y|v)>$, sorted in increasing order of $\hat{p}(y|v)$. That is, $\mathcal{O}(y)$=$\{\ldots, <o(y, v_i), \hat{p}(y|v_i)>, <o(y, v_j), \hat{p}(y|v_j)>, \ldots\}$, such that $\hat{p}(y|v_i) \leq \hat{p}(y|v_j)$. Also, consider c a candidate value for f_y. In this case, $\mathcal{O}_c(y, \leq)$ is a sub-list of $\mathcal{O}(y)$, that is, $\mathcal{O}_c(y, \leq)$=$\{\ldots, <o(y, v), \hat{p}(y|v)>, \ldots\}$, such that for all pairs in $\mathcal{O}_c(y, \leq)$, $\hat{p}(y|v) \leq c$. Similarly, $\mathcal{O}_c(y, >)$=$\{\ldots, <o(y, v), \hat{p}(y|v)>, \ldots\}$, such that for all pairs in $\mathcal{O}_c(y, >)$, $\hat{p}(y|v) > c$. In other words, $\mathcal{O}_c(y, \leq)$ and $\mathcal{O}_c(y, >)$ are two partitions of $\mathcal{O}(y)$ induced by c.*

Definition 5. *Consider $N_0(\mathcal{O}(y))$ the number of elements in $\mathcal{O}(y)$ for which $o(y, v)$=0. Similarly, consider $N_1(\mathcal{O}(y))$ the number of elements in $\mathcal{O}(y)$ for which $o(y, v)$=1.*

The first step of our entropy-minimization calibration approach is to calculate the entropy of tag y in $\mathcal{O}(y)$, as shown in Equation 3.

$$E(\mathcal{O}(y)) = -\left(\frac{\mathrm{N}_0(\mathcal{O}(y))}{|\mathcal{O}(y)|} \times \log \frac{\mathrm{N}_0(\mathcal{O}(y))}{|\mathcal{O}(y)|}\right) - \left(\frac{\mathrm{N}_1(\mathcal{O}(y))}{|\mathcal{O}(y)|} \times \log \frac{\mathrm{N}_1(\mathcal{O}(y))}{|\mathcal{O}(y)|}\right) \quad (3)$$

The second step is to calculate the sum of the entropies of tag y in each partition induced by c, according to Equation 4.

$$E(\mathcal{O}(y), c) = \frac{|\mathcal{O}_c(y, \leq)|}{|\mathcal{O}(y)|} \times E(\mathcal{O}_c(y, \leq)) + \frac{|\mathcal{O}_c(y, >)|}{|\mathcal{O}(y)|} \times E(\mathcal{O}_c(y, >)) \quad (4)$$

The third step is to set f_y to the value of c which minimizes the difference $E(\mathcal{O}(y)) - E(\mathcal{O}(y), c)$. Now, the final step is to calibrate each $\hat{p}(y|t)$ (note that $t \in \mathcal{T}$) using the corresponding f_y (which was obtained using the validation set). The intuition is that f_y separates values of $\hat{p}(y|t)$ that should be considered high (i.e., $\hat{p}(y|t) > f_y$) from those that should be considered low (i.e., $\hat{p}(y|t) \leq f_y$). Thus, a natural way to calibrate $\hat{p}(y|t)$ is to calculate how many times $\hat{p}(y|t)$ is greater than f_y. This can be easily done as shown in Equation 5. The values of $\hat{c}(y|t)$ are directly comparable, since the corresponding values of $\hat{p}(y|t)$ were normalized by f_y. Thus, $\hat{c}(y|t)$ is used to sort candidate tags that are more likely to be associated with object t:

$$\hat{c}(y|t) = \frac{\hat{p}(y|t)}{f_y} \quad (5)$$

In some cases, calibration may drastically improves recommendation performance, as we will show in the next section.

4 Experimental Evaluation

In this section we empirically analyze the recommendation performance of LATRE. We employ as the basic evaluation measures precision at x (p@x), which measures the proportion of relevant tags in the x first positions in the tag ranking, and MRR [6, 20], which shows the capacity of a method to return relevant tags early in the tag ranking. We first present the baseline and collections employed in the evaluation, and then we discuss the recommendation performance of LATRE on these collections.

4.1 Baseline

The baseline method used for comparison is described in [20]. It is a co-occurrence method which also employs association rules. It gives less weight to candidate tags that are either too popular or too rare. Furthermore, its algorithm gives more weight to candidate tags that are higher in each candidate tag list with the goal of smoothening the co-occurrence values decay. Other related methods in Section 2 were not considered as baselines because they use additional information, such as the user history [6], image features [25] and the page content [7, 19].

4.2 Collections

Differently from related work that present results restricted to a single collection [6, 7, 20], in this paper we experiment with several collections, namely, Delicious, LastFM and YouTube.

Delicious

Delicious is a popular social bookmarking application that permits users to store, share and discover bookmarks. Users can categorize their bookmarks using tags, which serve as personal indexes so that a user can retrieve its stored pages. The assignment of tags in Delicious is collaborative, i.e., the set of tags associated with a page is generated by many users in collaboration.

For the Delicious crawl we used its "Recent Bookmarks" page, which is a public timeline that shows a subset of the most recently bookmarked objects. We collected unique bookmarked page entries and extracted the set of most frequently used bookmarks for a page (i.e., its "top bookmarks"). Delicious makes available as many as 30 "top bookmarks". Therefore, this is the maximum number of tags per object in our crawled dataset.

The crawl was performed in October 2009. We obtained 560,033 unique object entries and 872,502 unique tags. The mean number of tags per object is 18.27. The number of tags per object ranged from 1 to 30. The first, second and third quartile of the distribution of tags per page is, respectively, 8, 17 and 27.

LastFM

LastFM is a Web 2.0 music website and Internet radio. It allows users to collaboratively contribute with tags to categorize and describe the characteristics of artists, such as the music genre. LastFM was crawled using a snowball approach, which collects a set of seed artists and follows links to related artists. The artists used as seeds are the ones associated with the system most popular tags.

The crawl was performed in October 2008. We obtained 99,161 unique object entries and 109,443 unique tags. The mean number of tags per object is 26.88. The number of tags per object ranged from 1 to 210. The first quartile of the distribution of tags per object is 7, the second quartile is 14, and the third quartile is 31.

YouTube

YouTube is the largest video sharing application on the Web. Users that upload videos to YouTube can provide a set of tags that describe them for indexing purposes. YouTube is different from Delicious and LastFM in that it is non-collaborative, that is, only the video uploader can provide tags.

YouTube was crawled using a snowball approach, following links between related videos. The all-time most popular videos were used as seeds. Our sample was obtained in July 2008. We obtained 180,778 unique objects, and they were described by 132,694 unique tags. The mean number of tags per object is 9.98. The number of tags per object ranged from 1 to 91. The first quartile of the distribution of tags per object is 6, the second quartile is 9 and the third quartile is 14.

4.3 Pre-Processing Steps and Setup

In order to assess the recommendation performance of the evaluated methods, we equally divided the tags associated with test object $t = <\mathcal{I}_t, \mathcal{Y}_t>$: half of the tags is included in $\mathcal{I}_t$, and the other half is included in $\mathcal{Y}_t$ and used to assess the performance. This division is made by shuffling the tags and including the first half in $\mathcal{I}_t$ and the last half in $\mathcal{Y}_t$. A similar approach has been adopted in [6] and [20]. We applied Porter's stemming algorithm [14] to avoid trivial recommendations such as plurals and small variations of the same input word.

We split each collection into three subsets: the first subset is composed of objects with a large number of tags, the second subset is composed of objects with a moderate number of tags, and the third subset is composed of objects with a small number of tags. The range of tags per object was selected in a way that the corresponding subsets have approximately the same number of objects. These subsets are shown in Table 3.

Then, we randomly selected 20,000 objects from each of the subsets. We divided each group of selected objects into 5 partitions of 4,000 objects each, and we used 5-fold cross validation to assess recommendation performance. We use the validation set to find the best parameters for each evaluated method.

Table 3. We divided each collection into three subsets according to the number of tags per object. We show the number of objects in each of these subsets and the average number of tags per object (and its standard deviation) in each subset. Note that we excluded all objects associated with a single tag, since we need at least one tag in $\mathcal{I}_t$ and one tag in $\mathcal{Y}_t$.

Collection	Range	# Objects	Avg. # Tags
Delicious	2 to 6 tags/object	188,173	3.94 ± 1.38
	7 to 12 tags/object	167,613	9.50 ± 1.73
	13 to 30 tags/object	170,708	15.92 ± 2.53
LastFM	2 to 6 tags/object	29,622	3.96 ± 1.39
	7 to 16 tags/object	30,215	10.55 ± 2.77
	17 to 152 tags/object	31,492	44.99 ± 25.30
YouTube	2 to 5 tags/object	56,721	3.63 ± 1.09
	6 to 9 tags/object	53,284	7.39 ± 1.11
	10 to 74 tags/object	59,285	13.60 ± 5.02

4.4 Results

All experiments were performed on a Linux PC with an Intel Core 2 Duo 2.20GHz and 4GBytes RAM. In the following subsections we discuss the effectiveness and the computational efficiency of LATRE.

4.5 Precision

Tables 4, 5 and 6 show the results for p@x for each subset of the three collections. We varied x from 1 to 5, following the analysis performed in previous work [6, 20].

The reason is that we are interested in the performance of the methods on the top of the ranking (i.e. the first 5 recommendations), since in tag recommendation the user is not likely to scan a large number of tags before choosing which ones are relevant. Furthermore, it is better to recommend good tags earlier in the ranking (e.g. p@1), so that the user has to scan fewer tags. We executed three algorithms over the subsets: the baseline, LATRE without calibration (referred to as LATNC) and LATRE.

Statistical tests have shown that LATRE performs significantly better ($p < 0.05$) than the baseline in all scenarios we experimented with. LATRE has shown gains in p@5 from 6.4% in LastFM to 23.9% in Delicious if we consider only the lower ranges; considering only the middle ranges, LATRE has shown gains in p@5 from 10.7% in Delicious to 28.9% in YouTube; and considering only the upper ranges, LATRE has shown gains in p@5 from 17.2% in Delicious to 46.7% in LastFM. It is important to note that the absolute precision values shown in this paper are underestimated, since there may be additional tags that are relevant to the user and that were not used by he/she to describe the object (and thus are not in $\mathcal{Y}_t$), as discussed in [6].

One interesting conclusion we could draw from the experiments is that calibration has its best performance in the lower ranges, indicating that the distortions described in Section 3 have a more damaging effect in these ranges. It is specially difficult to perform well in the lower ranges since there is very little information to work with, e.g., when there are two tags associated with an object, only one tag can be used as input and only one tag can be considered to be the correct answer. Several applications that use tag co-occurrence could benefit from calibration in these cases, such as in tag expansion for the cold start problem [9, 7] (see Section 2).

As the number of tags per object increases, the benefit of using more elaborated rules becomes clearer. The gains in precision in the middle and upper ranges are mainly due to LATNC (i.e., LATRE without calibration), and the reason is that there are more opportunities for producing complex rules in these ranges, i.e., there is more information available. Applications that use tag co-occurrence in objects with a large number of tags could benefit from these elaborated rules, such as tag expansion for index enrichment [20, 7], tag ranking [11] or tag translation [19].

It is interesting to notice that in the range 17-152 of LastFM, the baseline has achieved a recommendation performance which is lower than its performance in the range 7-16 (p@1=0.40 vs. p@1=0.54). The baseline does not perform well in range 17-152 of LastFM because this subset has a high number of tags per object (see Table 3), and the algorithm tends to recommend tags that are too general, such as “music” and “listen”.

Furthermore, Tables 4, 5 and 6 show that the absolute precision values for Delicious are lower than the corresponding values in LastFM and YouTube. The reason is that Delicious has a much more diverse set of objects, since Web pages can contain or refer to any kind of data, information or media. As an example, YouTube and LastFM pages can also be stored as a bookmark in Delicious. In fact, the number of distinct tags in Delicious in the five partitions used in our experiment (20,000 objects) is higher than in YouTube and LastFM. In the lower ranges, Delicious has 13,247 unique tags, while LastFM and YouTube have 5,164 and 6,860, respectively. The same relative proportions were found in the middle and higher ranges.

Mean Reciprocal Rank (MRR)

Tables 4, 5 and 6 also show the values of MRR for all ranges. MRR shows the capacity of a method to return relevant tags early in the tag ranking. Statistical significance tests were performed ($p < 0.05$) and results that are statistically different from the baseline are shown in bold face. We can see that LATRE has results significantly better than the baseline for all ranges in all datasets.

Table 4. Delicious: results for p@1, p@3, p@5 and MRR. Statistically significant differences ($p < 0.05$) are shown in (1) bold face and (2) marked with an asterisk (*), representing respectivelly (1) cases in which LATNC and/or LATRE performs better than the baseline and (2) cases in which LATNC performs worse than the baseline. LATRE relative gains are shown in the last row.

	2-6 tags/object				7-12 tags/object				13-30 tags/object			
	p@1	p@3	p@5	MRR	p@1	p@3	p@5	MRR	p@1	p@3	p@5	MRR
Baseline	.106	.063	.046	.158	.307	.196	.150	.417	.506	.362	.285	.627
LATNC	.105	.059*	.045*	.154	**.328**	**.214**	**.160**	**.426**	**.544**	**.414**	**.335**	**.659**
LATRE	**.129**	**.077**	**.057**	**.188**	**.330**	**.218**	**.166**	**.435**	**.543**	**.413**	**.334**	**.659**
[% gain]	**21.7**	**22.2**	**23.9**	**19.0**	**7.5**	**11.2**	**10.7**	**4.3**	**7.3**	**14.1**	**17.2**	**5.1**

Table 5. LastFM: results for p@1, p@3, p@5 and MRR.

	2-6 tags/object				7-16 tags/object				17-152 tags/object			
	p@1	p@3	p@5	MRR	p@1	p@3	p@5	MRR	p@1	p@3	p@5	MRR
Baseline	.313	.174	.125	.403	.539	.366	.279	.646	.400	.328	.289	.560
LATNC	.320	**.180**	.128	.409	**.574**	**.410**	**.314**	**.670**	**.564**	**.476**	**.425**	**.695**
LATRE	**.327**	**.187**	**.133**	**.418**	**.575**	**.411**	**.316**	**.672**	**.564**	**.475**	**.424**	**.695**
[% gain]	**4.5**	**7.5**	**6.4**	**3.7**	**6.7**	**12.3**	**13.3**	**4.0**	**41.0**	**44.8**	**46.7**	**24.1**

Table 6. YouTube: results for p@1, p@3, p@5 and MRR.

	2-5 tags/object				6-9 tags/object				10-74 tags/object			
	p@1	p@3	p@5	MRR	p@1	p@3	p@5	MRR	p@1	p@3	p@5	MRR
Baseline	.288	.153	.105	.356	.405	.273	.204	.646	.534	.419	.347	.560
LATNC	**.328**	**.171**	**.113**	**.385**	**.485**	**.356**	**.254**	**.556**	**.626**	**.528**	**.459**	**.704**
LATRE	**.350**	**.184**	**.122**	**.411**	**.491**	**.365**	**.263**	**.567**	**.627**	**.530**	**.462**	**.706**
[% gain]	**21.5**	**20.3**	**16.2**	**15.5**	**21.2**	**33.7**	**28.9**	**14.1**	**17.4**	**26.5**	**33.1**	**10.7**

Computational Efficiency

We evaluated LATRE efficiency by measuring the average execution time per object. Table 7 shows the results for each subset. For subsets with few tags per object, the average and maximum tagging time are in the order of few milliseconds. As expected, the average tagging time increases with the ratio of tags per object. However, even for objects that are associated with many tags, the average time spent per object is never

greater than 1.8 seconds. This makes LATRE specially well-suited for real-time tag recommendation [22].

Table 7. Tagging time in seconds. We also show the standard deviation of the average tagging time.

Collection	Subset	Avg. Time	Max. Time
Delicious	2-6	0.0023 ± 0.0019	0.016
	7-12	0.067 ± 0.047	0.33
	13-30	0.47 ± 0.24	1.23
LastFM	2-6	0.0062 ± 0.0055	0.039
	7-16	0.20 ± 0.15	1.18
	17-152	1.77 ± 0.34	2.44
YouTube	2-5	0.0023 ± 0.0023	0.037
	6-9	0.027 ± 0.026	0.32
	10-74	0.31 ± 0.27	1.56

The last set of experiments aims at verifying the increase in the number of extracted rules as a function of α_{max}. According to Theorem 1, the number of rules extracted by LATRE increases polynomially. Table 8 contrasts the number of rules extracted by LATRE with the number of rules that would be extracted by the baseline. We only show the results for subsets of the LastFM collection, but the same trends are also observed for the subsets of the other two collections. Clearly, the number of rules extracted by the baseline increases exponentially, while the number of rules extracted by LATRE increases at a much slower pace.

Table 8. Number of extracted rules for each subset of LastFM.

α_{max}	2-6 tags/object		7-16 tags/object		17-152 tags/object	
	Baseline	LATRE	Baseline	LATRE	Baseline	LATRE
1	2.10^7	3.10^6	5.10^7	2.10^7	5.10^7	5.10^7
2	1.10^{11}	3.10^6	3.10^{11}	3.10^7	4.10^{11}	6.10^7
3	5.10^{14}	3.10^6	2.10^{15}	4.10^7	3.10^{15}	6.10^7

5 Conclusions

In this paper we have introduced LATRE, a novel co-occurrence based tag recommendation method. LATRE extracts association rules from the training data on a demand-driven basis. The method projects the search space for rules according to qualitative information in test objects, allowing an efficient extraction of more elaborate rules. LATRE interprets each extracted rule as a vote for a candidate tag. After all votes are summed, tags are sorted according to their scores. Finally, LATRE calibrates the scores in order to correct possible distortions in the final ranked list of tags.

Our experiments involve objects belonging to different media types, namely Web pages from Delicious, audio from LastFM, and videos from YouTube. We have shown that LATRE recommends tags with a significantly higher precision in all subsets when compared against the baseline. While our proposed calibration mechanism has its best performance in subsets with a few number of tags, the use of more elaborate rules improves precision in subsets with a larger number of tags. We have proved that LATRE is able to efficiently extract such elaborate rules from the training data. LATRE achieved improvements in precision (p@5) from 10.7% to 23.9% for Delicious, from 6.4% to 46.7% for LastFM, and from 16.2% to 33.1% for YouTube.

As future work, we will investigate other textual features of the Web 2.0, and how these features may improve tag recommendation. Further, we will extend LATRE, so that recommended tags will be used as additional information which can be exploited to improve recommendation effectiveness.

6 Acknowledgements

We thank the partial support given by the Brazilian National Institute of Science and Technology for the Web (grant MCT/CNPq 573871/2008-6), Project InfoWeb (grant MCT/CNPq/CT-INFO 550874/2007-0), and authors' individual grants and scholarships from CNPq and FAPEMIG.

References

1. R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 207–216, 1993.
2. C.-m. Au Yeung, N. Gibbins, and N. Shadbolt. User-induced links in collaborative tagging systems. In *CIKM '09: Proc. of the 18th ACM Conference on Information and Knowledge Management*, pages 787–796, 2009.
3. K. Bischoff, C. S. Firan, W. Nejdl, and R. Paiu. Can all tags be used for search? In *CIKM '08: Proc. of the 17th ACM Conference on Information and Knowledge Management*, pages 193–202, 2008.
4. M. J. Carman, M. Baillie, R. Gwadera, and F. Crestani. A statistical comparison of tag and query logs. In *SIGIR '09: Proc. of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 123–130, 2009.
5. F. Figueiredo, F. Belém, H. Pinto, J. Almeida, M. Gonçalves, D. Fernandes, E. Moura, and M. Cristo. Evidence of quality of textual features on the web 2.0. In *CIKM '09: Proc. of the 18th ACM Conference on Information and Knowledge Management*, pages 909–918, 2009.
6. N. Garg and I. Weber. Personalized, interactive tag recommendation for flickr. In *RecSys '08: Proc. of the 2008 ACM Conference on Recommender Systems*, pages 67–74, 2008.
7. P. Heymann, D. Ramage, and H. Garcia-Molina. Social tag prediction. In *SIGIR '08: Proc. of the 31nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 531–538, 2008.
8. I. Konstas, V. Stathopoulos, and J. M. Jose. On social networks and collaborative recommendation. In *SIGIR '09: Proc. of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 195–202, 2009.

9. R. Krestel, P. Fankhauser, and W. Nejdl. Latent dirichlet allocation for tag recommendation. In *RecSys '09: Proc. of the 2009 ACM conference on Recommender systems*, pages 61–68, 2009.
10. X. Li, L. Guo, and Y. E. Zhao. Tag-based social interest discovery. In *WWW '08: Proc. of the 17th International Conference on World Wide Web*, pages 675–684, 2008.
11. D. Liu, X.-S. Hua, L. Yang, M. Wang, and H.-J. Zhang. Tag ranking. In *WWW '09: Proc. of the 18th International Conference on World Wide Web*, pages 351–360, 2009.
12. C. Lu, X. Chen, and E. K. Park. Exploit the tripartite network of social tagging for web clustering. In *CIKM '09: Proc. of the 18th ACM Conference on Information and Knowledge Management*, pages 1545–1548, 2009.
13. A. Plangprasopchok and K. Lerman. Constructing folksonomies from user-specified relations on flickr. In *WWW '09: Proc. of the 18th International Conference on World Wide Web*, pages 781–790, 2009.
14. M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.
15. D. Ramage, P. Heymann, C. D. Manning, and H. Garcia-Molina. Clustering the tagged web. In *WSDM '09: Proc. of the Second ACM International Conference on Web Search and Data Mining*, pages 54–63, 2009.
16. R. Schenkel, T. Crecelius, M. Kacimi, S. Michel, T. Neumann, J. X. Parreira, and G. Weikum. Efficient top-k querying over social-tagging networks. In *SIGIR '08: Proc. of the 31nd international ACM SIGIR conference on Research and development in information retrieval*, pages 523–530, 2008.
17. S. Sen, J. Vig, and J. Riedl. Tagommenders: connecting users to items through tags. In *WWW '09: Proc. of the 18th International Conference on World Wide Web*, pages 671–680, 2009.
18. A. Shepitsen, J. Gemmell, B. Mobasher, and R. Burke. Personalized recommendation in social tagging systems using hierarchical clustering. In *RecSys '08: Proc. of the 2008 ACM conference on Recommender systems*, pages 259–266, 2008.
19. S. Siersdorfer, J. San Pedro, and M. Sanderson. Automatic video tagging using content redundancy. In *SIGIR '09: Proc. of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 395–402, 2009.
20. B. Sigurbjörnsson and R. van Zwol. Flickr tag recommendation based on collective knowledge. In *WWW '08: Proc. of the 17th International Conference on World Wide Web*, pages 327–336, 2008.
21. Y. Song, L. Zhang, and C. L. Giles. A sparse gaussian processes classification framework for fast tag suggestions. In *CIKM '08: Proc. of the 17th ACM Conference on Information and Knowledge Management*, pages 93–102, 2008.
22. Y. Song, Z. Zhuang, H. Li, Q. Zhao, J. Li, W.-C. Lee, and C. L. Giles. Real-time automatic tag recommendation. In *SIGIR '08: Proc. of the 31nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 515–522, 2008.
23. F. M. Suchanek, M. Vojnovic, and D. Gunawardena. Social tags: meaning and suggestions. In *CIKM '08: Proc. of the 17th ACM Conference on Information and Knowledge Management*, pages 223–232, 2008.
24. K. Q. Weinberger, M. Slaney, and R. Van Zwol. Resolving tag ambiguity. In *MM '08: Proc. of the 16th ACM International Conference on Multimedia*, pages 111–120, 2008.
25. L. Wu, L. Yang, N. Yu, and X.-S. Hua. Learning to tag. In *WWW '09: Proc. of the 18th International Conference on World Wide Web*, pages 361–370, 2009.
26. Z. Xu, Y. Fu, J. Mao, and D. Su. Towards the semantic web: Collaborative tag suggestions. In *WWW '06: Proc. of the Collaborative Web Tagging Workshop*, 2006.