Personalized Recommendation in Social Tagging Systems Using Hierarchical Clustering

Andriy Shepitsen, Jonathan Gemmell, Bamshad Mobasher, and Robin Burke
Center for Web Intelligence
School of Computing, DePaul University
Chicago, Illinois, USA
ashepits,jgemmell,mobasher,rburke@cs.depaul.edu

ABSTRACT

Collaborative tagging applications allow Internet users to annotate resources with personalized tags. The complex network created by many annotations, often called a folksonomy, permits users the freedom to explore tags, resources or even other user's profiles unbound from a rigid predefined conceptual hierarchy. However, the freedom afforded users comes at a cost: an uncontrolled vocabulary can result in tag redundancy and ambiguity hindering navigation. Data mining techniques, such as clustering, provide a means to remedy these problems by identifying trends and reducing noise. Tag clusters can also be used as the basis for effective personalized recommendation assisting users in navigation. We present a personalization algorithm for recommendation in folksonomies which relies on hierarchical tag clusters. Our basic recommendation framework is independent of the clustering method, but we use a context-dependent variant of hierarchical agglomerative clustering which takes into account the user's current navigation context in cluster selection. We present extensive experimental results on two real world dataset. While the personalization algorithm is successful in both cases, our results suggest that folksonomies encompassing only one topic domain, rather than many topics, present an easier target for recommendation, perhaps because they are more focused and often less sparse. Furthermore, context dependent cluster selection, an integral step in our personalization algorithm, demonstrates more utility for recommendation in multi-topic folksonomies than in single-topic folksonomies. This observation suggests that topic selection is an important strategy for recommendation in multi-topic folksonomies.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.3 Information Search and Retrieval—*Clustering, Information filtering*; H.2 [**Database Management**]: H.2.8 Database application—*Data mining*

General Terms

Algorithms

RecSys'08, October 23–25, 2008, Lausanne, Switzerland.

Keywords

Collaborative tagging, clustering, personalization, recommender systems

1. INTRODUCTION

Collaborative or social tagging systems allow Internet users to manage online resources. Central to collaborative tagging is the annotation; a user applies a personalized tag to a resource. Many annotations taken together results in a complex network of interrelated users, resources and tags commonly referred to as a folksonomy [10]. The ability to explore this large information space, free from a rigid predefined conceptual hierarchy, is central to the recent rise in popularity of collaborative tagging applications.

One of the most prominent applications is del.icio.us[1] in which users annotate URLs. Flickr[2], another popular tagging system, allows users to upload, share and annotate pictures. Last.FM[3] allows users to tag their music collection, and uses the wisdom of the crowd to generate recommendations. Other collaborative tagging applications focus on blogs, citations and wikis.

Collaborative tagging has proven to be intuitive for user to adopt. By selecting a previously-used tag, it is easy to retrieve annotated resources [6]. Resources can be categorized by several tags, rather than one directory or a single branch of a hierarchy [12]. It offers a sense of community not provided by other resource management or information retrieval tools. Users may share or discover resources through the collaborative network and connect to people with similar interests. Collaborative tagging applications benefit from the opinions of many users rather than a dominant view provided by a few. They are consequently more nimble, able to adjust to a changing vocabulary and absorb trends quickly [18].

From the system point of view, perhaps the greatest advantage offered by collaborative tagging applications is the richness of the user profiles. As users annotate resources, the system is able to monitor both their interest in resources as well as their vocabulary for describing those resources. These profiles are a powerful tool for recommendation algorithms [20]. The relatively low cost of generating a user profile can be dramatically overcome by the improved user experience that personalization allows.

While collaborative tagging applications have many benefits, they also present unique challenges. Unsupervised tagging, integral to the open nature of folksonomies, results in a wide variety of tags that can be redundant, ambiguous or entirely idiosyncratic. Tag redundancy, in which several tags have the same meaning, can obfuscate the similarity among resources. Redundant tags can hinder al-

[1] del.icio.us
[2] flickr.com
[3] last.fm

gorithms that depend on identifying similarities between resources. On the other hand, tag ambiguity, in which a single tag has many meanings, can falsely give the impression that resources are similar when they are in fact unrelated.

Because the information space is complicated by redundancy and ambiguity, personalized recommendation strategies are needed to aid the user as they interact with the system. For example, consider the user, Ozzie, who has tagged several Chicago White Sox Web resources with the ambiguous tag "sox." If Ozzie selects that tag, the system should not recommend resources concerning the Sarbanes-Oxley Act, abbreviated as "SOX", and certainly nothing to do with the Boston Red Sox. In addition, other White Sox fans may have used alternative tags: "Sox," "whiteSox," "chisox," etc. These resources may not have been tagged with "sox," but they should still be made available to Ozzie.

Recommendation strategies must also be adapted to deal with the unique structure of folksonomies. Typically, recommender systems have dealt with two dimensions: users and items. In folksonomies items are analogous to resources. In collaborative tagging applications, however, a third dimension is introduced: tags. Because the nature of folksonomies encourages sharing and exploration, recommender systems are needed not only to suggest resources, but tags and even other users as well.

Consider again the White Sox fan, Ozzie. After selecting "sox," the system may recommend resources, tag and other users related to that tag. He may notice, Reinsdorf, another fan of the White sox, and choose to view his profile. Finally, Ozzie may notice resources in Reinsdorf's profile dealing with the Chicago Bulls, and view one of these resources. The freedom to navigate through all three dimensions of the folksonomy is central to its utility. A primary concern of recommender systems in collaborative tagging is to present users with avenues for navigation that are most relevant to their information needs. However, this complex navigational structure can be viewed more clearly as a series of recommendations. In this paper we focus on the most common recommendation scenario; a user selects a tag and expects a recommendation of resources.

We previously studied the role of tag clustering in personalized search and navigation [4, 5]. We showed that data mining techniques such as clustering provide a means to overcome redundancy and ambiguity thereby facilitating recommendation. By clustering, redundant tags can be aggregated; the combined trend of a cluster can be more easily detected than the effect of a single tag. The effect of ambiguity can also be diminished, since a cluster imbues a tag with the meaning shared by the associated tags. Using the clusters as the nexus between users and resources, the relevance of a resource to a user can then be inferred and used for recommendation.

As seen at Figure 1 our cluster-based personalization algorithm accepts a user profile, a set of clusters, and a selected tag. As output, it recommends resources. It models users and resources as vectors over the set of tags. By measuring the importance of a tag cluster to a user, the user's interests can be better understood. By associating resources with tag clusters, the importance of a resource to the topic described by the cluster can be calculated. We use a context dependent hierarchical agglomerative algorithm for clustering tags. In contrast to the traditional agglomerative algorithm, our modification selects a subset of potential clusters related to the user's current navigational activity.

In this paper, we present the results of experiments on two real world dataset. As in previous experiments, the personalization algorithm provides improvement. However, the results demonstrate interesting differences. First, the recommendation task in a folksonomy encompassing only one subject domain appears to be easier, unclouded by the effect of additional domains present in larger folksonomies and aided by a denser dataset. Second, a recommendation strategy designed to reduce the topic space fairs well in both single topic and multi-topic folksonomies, but provides more significant improvement in a multi-topic folksonomy. We believe that by reducing the information space to focus on the user's interest, through our context dependent hierarchical clustering approach, we reduce noise by clarifying the user's intended meaning of ambiguous tags and filtering resources annotated with redundant tags.

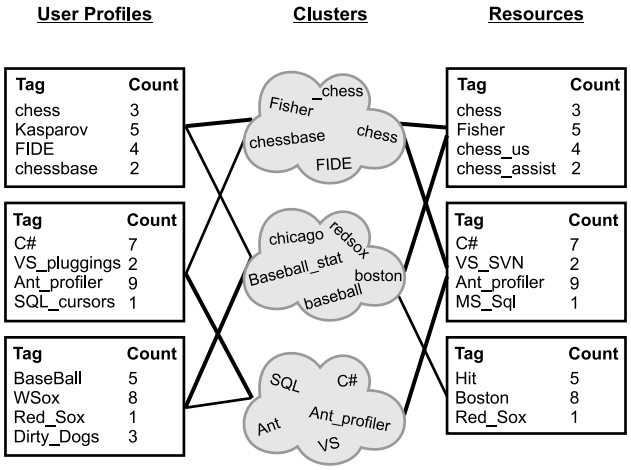

Figure 1: Clusters serve as intermediaries between users and resources.

2. RELATED WORK

Central to this paper is the ability of clustering algorithms to aggregate tags into topic domains. Previously, in [4, 5], we demonstrated how tag clusters serving as coherent topics can aid in the personalization of search and navigation. Further support for the utility of clustering is offered in [2] where improvement in search through clustering is theorized.

In [7], topic relevant partitions are created by clustering resources rather than tags. The most characteristic representatives of a cluster are recommended for users interested in a domain described by a cluster. Using clusters of resources, Flickr improves recommendation by distinguishing between alternative meanings of a query. For example, a user selecting the tag "apple" will receive several groups of resources. One group represents "fruit"; while another contains iPods, iMacs, and iPhones. A third cluster contains pictures of New York City. In [3] clusters of resources are shown to improve recommendation by categorizing the resources into topic domains. Consequently, the user may interactively disambiguate his query.

The utility of clustering extends beyond the scope of recommendation. In [8] hierarchical clustering is proposed to generate a taxonomy from a folksonomy. In [18], tag clusters are presumed to be representative of the resource content. Thus, a folksonomy of Web resources is used to move the Internet closer to the Semantic Web. Tag clustering can support tag recommendation, reducing annotation to a mouse click rather than a text entry. Well-chosen tags make the recovery process simple and offer some control over the tag-space diminishing tag redundancy and ambiguity to some degree. In [19], a group of tags are offered to the user based on several criteria (coverage, popularity, effort, uniformity) resulting in a cluster of a relevant tags.

In [1], ranking of web search was optimized using social annotations by taking in account the similarity of the query to the resources in del.icio.us. Their work is based on the assumption that folksonomies, such as del.icio.us, offer insights to the user's information needs. Our work shares this assumption as we seek to personalize the user recommendation.

In [9], a novel algorithm, FolkRank, for search and ranking in folksonomies is proposed that depends on interrelated tags, resources and users. The authors extend the commonly known PageRank algorithm to folksonomies under the assumption that users, resources and tags are important if they are connected to other important tags, resources and users in folksonomies. They use a weight passing scheme to derive the importance of an object in folksonomies. In this paper, we also adopt the idea of deriving the importance of resources to the users.

Integral to our algorithm for personalization is the measurement of relevance between a user and a resource. A similar notion was previously described in [13] in which an affinity level was calculated between a user and a set of tag clusters. A collection of resources was then identified for each cluster based on tag usage. Resources were recommended to the user based on the user's affinity to the clusters and the associated resources.

3. RECOMMENDATION ALGORITHM

A folksonomy can be described as a four-tuple. There exists a set of users, U; a set of resources, R; a set of tags, T; and a set of annotations, A. We denote the data in the folksonomy as D and define it as:

$$D = \langle U, R, T, A \rangle \tag{1}$$

The annotations, A, are represented as a set of triples containing a user, tag and resource defined as:

$$A \subseteq \{\langle u, r, t \rangle : u \in U, r \in R, t \in T\} \tag{2}$$

A folksonomy can, therefore, be viewed as a tripartite hypergraph [11] with users, tags, and resources represented as nodes and the annotations represented as hyper-edges connecting one user, tag and resource.

3.1 Standard Recommendation Methods in Folksonomies

Collaborative tagging applications may vary in the way they handle recommendation. Possible methods include recency, authority, linkage, popularity, or vector space models. In this work we focus on the vector space model [15] adapted from the information retrieval discipline to work with folksonomies. Each user, u, is modeled as a vector over the set of tags, where each weight, $w(t_i)$, in each dimension corresponds to the importance of a particular tag, t_i.

$$\vec{u} = \langle w(t_1), w(t_2) ... w(t_{|T|}) \rangle \tag{3}$$

Resources can also be modeled as a vector over the set of tags. In calculating the vector weights, a variety of measures can be used. The *tag frequency*, *tf*, for a tag, *t*, and a resource, *r*, is the number of times the resource has been annotated with the tag. We define *tf* as:

$$tf(t,r) = |\{a = \langle u, r, t \rangle \in A : u \in U\}| \tag{4}$$

Likewise, the well known *term frequency * inverse document frequency* [14] can be modified for folksonomies. The *tf*idf* multiplies the aforementioned frequency by the relative distinctiveness of the tag. The distinctiveness is measured by the log of the total number of resources, *N*, divided by the number of resources to which the tag was applied, n_t. We define *tf*idf* as:

$$tf*idf(t,r) = tf(t,r) * \log(N/n_t) \tag{5}$$

With either term weighting approach, a similarity measure between a query, *q*, represented as a vector over the set of tags, and a resource, *r*, also modeled as a vector over the set of tags, can be calculated. However, in this work, since a user often initiates a request for recommendations by selecting a single tag, we assume the query is a vector with only one tag.

Several techniques exist to calculate the similarity between vectors such as the Jaccard similarity coefficient or Cosine similarity [16]. Cosine similarity is a popular measure defined as:

$$cos(q,r) = \frac{\sum_{t \in T} tf(t,q) * tf(t,r)}{\sqrt{\sum_{t \in T} tf(t,q)^2} * \sqrt{\sum_{t \in T} tf(t,r)^2}} \tag{6}$$

In this paper we presume that the user interacts with the system by selection a query tag and expects to receive resource recommendations. Therefore, a query is a unit vector consisting of a single tag, and the equation may be further simplified.

$$cos(q,r) = \frac{tf(q,r)}{\sqrt{\sum_{t \in T} tf(t,r)^2}} \tag{7}$$

To recommend resources, we can calculate the similarity of the selected tag to each resource and recommend the top n.

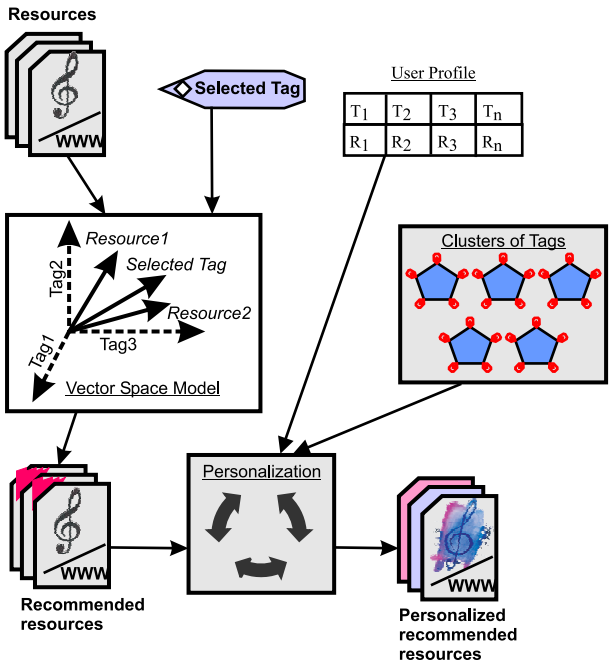

Figure 2: Steps in the personalization algorithm

3.2 The Personalized Recommendation Framework

The standard recommendation strategy outlined above does not take into account the user profile: identical results will be produced

regardless of who the user may be. While personalization has been shown to increase the utility of Web applications, the need for it in folksonomies is even more critical. Noise in the folksonomy such as tag redundancy and tag ambiguity obfuscate patterns and reduce the effectiveness of basic recommendation strategies. Recall Ozzie, the White Sox fan, and assume he has a friend that is a Red Sox fan. If the recommender application is not able to distinguish between these two users' meaning behind the tag "sox," both users will receive identical results.

Our personalization algorithm is illustrated in Figure 2. The process proceeds in two stages. First, given a user's click on a tag, the standard non-personalized recommendation algorithm is applied to produce a set of recommended resources. This set is then personalized by taking the user profile and tag clusters into account and re-ranking the results accordingly. Note that we assume the existence of a set of tag clusters obtained through the offline clustering phase. The details of our clustering approach are discussed in Section 4.

The personalized recommendation process is described in more detail below.

Step 1: *Calculate Cosine Similarity*. A basic search is performed on the query tag, q, using the similarity metric in Equation 7. A similarity, $S(q, r)$, is calculated for every resource $r \in R$ using this measure. As an output, this stage of the algorithm will produce a subset of resources, R', that have some similarity to the query tag and similarity scores for each.

Step 2: *Calculate Relevance of all $r \in R'$ to u*. In this step the clusters serve as a nexus between users and resources enabling the recommender to identify resources that mirror the user's interest. The input to this stage of the algorithm is the user profile, and the subset of resources, R' found in step 1. As output, the algorithm will return the relevance of each $r \in R'$ to u.

Step 2.1: *Calculate the user's interest in each cluster*. For each cluster, c, the user's interest is calculated as the ratio of times the user, u, annotated a resource with a tag from that cluster over the total number of annotations by that user. We denote this weight as *uc_w(u,c)* and defined it as:

$$\textit{uc\_w(u,c)} = \frac{|\{a = \langle u, r, t\rangle \in A : r \in R, t \in c\}|}{|\{a = \langle u, r, t\rangle \in A : r \in R, t \in T\}|} \quad (8)$$

Step 2.2: *Calculate each resource's closest clusters*. The relation of a resource, r, to a cluster is calculated as the ratio of times the resource was annotated with a tag from the cluster over the total number of times the resource was annotated. We call this weight *rc_w(r,c)* and defined it as:

$$\textit{rc\_w(r,c)} = \frac{|\{a = \langle u, r, t\rangle \in A : u \in U, t \in c\}|}{|\{a = \langle u, r, t\rangle \in A : u \in U, t \in T\}|} \quad (9)$$

Both *uc_w(u,c)* and *rc_w(r,c)* will always be a number between zero and one. A higher value will represent a strong relation to the cluster.

Step 2.3: *Infer the users's interest in each resource*. The relevance of the resource to the user, *relevance(u,r)*, is calculated from the sum of the product of these weights over the set of all clusters, C. This measure is defined as:

$$I(u, r) = \sum_{c \in C} \textit{uc\_w(u,c)} * \textit{rc\_w(r,c)} \quad (10)$$

Intuitively, each cluster can be viewed as the representation of a topic area. If a user's interests parallels closely the subject matter of a resource, the value for $I(u, r)$ will be correspondingly high.

Step 3: *Calculate Personalized Rank Score*. In the final step we combine the cosine similarity measure found in step 1 with the relevance measure found in step 2. A personalized similarity is calculated for each resource by multiplying the cosine similarity by the relevance of the resource to the user. We denote this similarity as $S'(u, q, r)$ and define it as:

$$S'(u, q, r) = S(q, r) * I(u, r) \quad (11)$$

Once the $S'(u, q, r)$, has been calculated for each resource and the resources have been sorted, the top n resources are returned to the user as recommendations. While the weights from clusters to resources will be constant regardless of the user, the weights connecting the users to the clusters will differ based on the user profile. Consequently, the resulting S' will depend on the user and the results will be personalized. With this approach we would expect the biggest improvement with the tags that identify the user interest rather than subjective or organizational ones.

4. TAG CLUSTERING FOR RECOMMENDATION

Clustering is an off-line step that is performed independently from our personalized recommendation algorithm. The discovered tag clusters form the basis of the recommendation algorithm. In previous work we evaluated several clustering techniques: maximal complete link, K-means, and hierarchical agglomerative clustering [4, 5]. The complete link and hierarchical clustering algorithms shows comparable results in different systems. The maximal complete link algorithm is effective with ambiguous tags and hierarchical clustering allows to filter out the clusters which do not have strong relation to the query tag. In this paper, however, we base our personalized recommender algorithm on the hierarchical agglomerative clustering technique due to its computation efficiency and significant flexibility. Furthermore, we propose modifications to the algorithm to make context related recommendations by choosing clusters particularly relevant to the user's selected tag. In this section, we first present the standard hierarchical agglomerative clustering approach applied to tags. We, then present our context dependent version of the algorithm which takes a selected tag into account in order to focus on relevant clusters.

4.1 Hierarchical Agglomerative Clustering

The input to the clustering module is a set of tags, T, the *step* and *division coefficient*. Tags are represented as a vector of weights over the set of resources, R. In these experiments we used both term frequency and $tfidf$ (Equation 5). The parameter *step* controls the granularity of the derived agglomerative clusters. The similarity threshold, initially set to one, is reduced by the value *step* at each iteration until it reaches zero. Clusters of tags are aggregated together if their similarity measure meets the current threshold. The *division coefficient* also plays a crucial role in the agglomerative clustering routine. It defines the level where the hierarchy is dissected into individual clusters. The output of the system is a set of tag clusters.

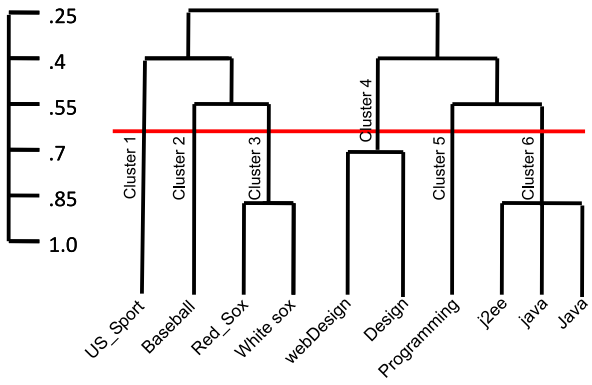

Figure 3: Hierarchical agglomerative clustering technique

Step 1. *Assign every tag to a singleton cluster*. To begin every tag in the system is placed in its own cluster, which serves as the seeds for the agglomerative clustering.
Step 2. *Combine all tags in one hierarchical cluster*. In this stage of the algorithm clusters of tags are aggregated together based on their similarity. Highly similar tags are aggregated together first, then less similar ones are combined. The result is a tree structure containing every tag in a hierarchy.

> ***Step 2.1*** *Combine clusters*. Clusters meeting the current similarity threshold are joined. There are many ways of determining the distance between clusters: single link, maximal link, average link, etc. In this paper, we determined the distance between clusters by using the centroids of the clusters.
> ***Step 2.2*** *Lower similarity*. The value for the *similarity threshold* is lower by *step*. The algorithm then repeated step 2.1 until all clusters have been aggregated.

Step 3. *Cut the tree to clusters*. The hierarchical tree is spit into clusters by cutting branches out of the tree at the *division coefficient* as shown in figure 3.

The tuning parameter in Step 2, *step* controls the granularity of agglomerative clustering and determines the number of levels for the hierarchical tree. An optimal value for *step* would aggregate tags slowly enough to capture the relationship between the concepts of the individual clusters. Aggregating the tags too quickly may over-specify the structure and lose important tag interdependencies.

In step 3, we used the second tuning parameter - *division coefficient*. It defines the level in the tag hierarchy where it is dissected into individual clusters. As we mentioned earlier the recommendation personalization algorithm relies on these clusters to serve as the intermediary between users and resources and presupposes the clusters represent distinct well defined topics. Hence, the selection of the *division coefficient* is integral to the success of the personalization algorithm. If the *division coefficient* is set too high, the result will be many small clusters. While the tags in these clusters may be very similar, they might not capture tag redundancies the way a larger cluster would. Likewise, if the coefficient is set too low the result is few clusters with low cohesiveness, providing insufficient disambiguation.

4.2 Context-Dependent Hierarchical Agglomerative Clustering

The algorithm described above uses the user-selected tag (query tag) only for the initial retrieval step. We propose here a modification to the recommendation algorithm that employs the tag to limit

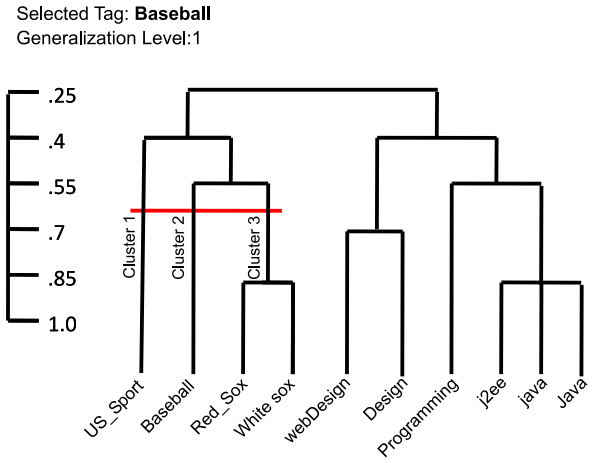

Figure 4: Context-dependent hierarchical agglomerative clustering technique

the set of clusters considered when calculating the user-resource interest score $I(u, r)$. The query tag is therefore used like a filter to choose clusters particularly interesting to the user context and increase the effectiveness of the personalized recommendation. This process is analogous to feature selection in data analysis when the most important feature of the object is selected. Specifically, we modify the clustering algorithm to return a subset of the available clusters, as a function of the input tag. See Figure 4. If the query tag is "baseball", the clusters on the right containing tags like "java" and "webDesign" are clearly not applicable. To select relevant clusters, we start with the cluster containing the target tag. We then generalize by moving up in the hierarchy to a pre-specified generalization level. The clusters below this node are then divided as before. The generalization level therefore determine the breadth of clusters to which we apply the user's query. Note that the generalization level is in some sense dependent on the value of the *step* parameter. The coarser the clusters, the more tags are subsumed by each generalization level. This version of the algorithm departs from the original at step 3.
Step 3. *Identify the user context*. After the user selects a tag, the algorithm finds the position of that tag in the hierarchical tree. According to *generalization level*, the algorithm widens the swath of clusters by first traveling up the hierarchy by the predefined number of levels and cutting off a larger branch. For example if the coefficient is two, the algorithm will travel two levels up the hierarchy from the selected tag.
Step 4. *Divide a branch of the tree into separate clusters*. This step is similar to Step 3 in the standard hierarchical algorithm with the difference that only a subtree, related to the user context, is dissected into clusters. We can see in the figure that a smaller number of clusters are generated by the context-dependent technique, because only the left-half of the tree is near the "baseball" node at one level of generalization. When the division coefficient is applied, three clusters are generated.

5. EXPERIMENTAL EVALUATION

5.1 Datasets and Experimental Methodology

We evaluated the recommendation algorithm based on two collaborative tagging systems. One is del.icio.us where the resources are Web-pages, and the second is Last.Fm where the resources are musical entries.

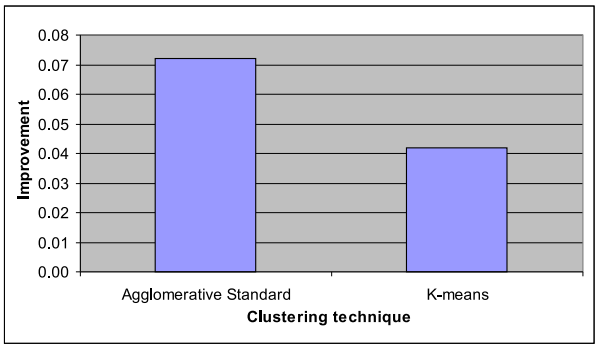

Figure 5: Comparison of clustering techniques for del.icio.us dataset

A Web crawler was used to extract data from del.icio.us from 5/26/2007 to 06/15/2007. The dataset contains 29,918 users, 6,403,442 resources and 1,035,177 tags. There are 47,184,492 tuples with one user, resource and tag per tuple. Last.Fm was crawled from 03/08/2008 to 03/26/2008. That dataset contains 18,364 users, 46,641 tags and 227,341 resources. There are 1,668,757 of users, tags and resources.

	Del.icio.us		Last.fm	
	Set 1	Set 2	Set 1	Set 2
Users	5,000	5,000	5,000	5,000
Tags	302,825	341,345	5,326	5,562
Resources	1,586,958	1,427,825	48,569	45,793

Table 1: Description of two Folksonomy datasets

In both cases we iterate over users and extract their entire profiles. In the del.icio.us dataset we built a queue with the users that had tagged the most popular resources. As we extracted the user profile we added users to the queue that had tagged the resources found in each profile.

Last.FM provides a service called "Audioscrobbler" through which we were able to extract user IDs. Every user has a list of neighbors and friends. Again we used a queue of users adding new user's IDs from the neighbor and friends list. Having users IDs we then extracted the users' full histories from the Last.FM web-site.

Two samples of 5,000 users were taken from each dataset. On each sample, five-fold cross validation was performed to measure the improvement in recommendation generation. For each fold, 20% of the users were partitioned from the rest as test users. Clustering was completed on the remaining 80%. From the test users, 10% of the instances were selected as test cases. The complete statistical description of two datasets is provided in the table 1.

Each test case consists of a user, tag and resource. We consider this resource as the target resource since we know that the user is interested in it. After performing a basic recommendation using only the tag, the rank of the target resource in the recommendation set was recorded. Likewise, after a personalized recommendation using both the tag and the user profile, the rank was also recorded.

By comparing the rank of the resource in the basic and personalized recommendations, we measured the effectiveness of the personalized recommendation algorithm. This section describes in more detail the process for judging the proposed algorithm, and supplies an evaluation of different clustering techniques and their modifications.

5.2 Comparison of Cluster Based Recommendations

Using the selected tag in the basic recommendation algorithm the rank of the target resource, r_b, was recorded. The rank of the resource is based on cosine similarity between the selected tag and list of resources. Likewise, the personalized recommendation algorithm was completed with the user profile and the tag. The rank of the resource in the personalized recommendations, r_p, was also recorded. Since the user had annotated this resource with the selected tag, we can assume that it is in fact relevant to the user, and a personalization should improve the position of the resource in the recommendation set. We used the "leave one out" approach, removing the target tag/resource pair from the user profile and took the rest of the user profile as input for the generation of the personalized recommendations.

The difference in the inverse of the two ranks can be used to judge the improvement provided by the personalized recommendations, imp [17]. It is defined as:

$$imp = \frac{1}{r_p} - \frac{1}{r_b} \tag{12}$$

If the personalized approach moves the resource higher in the recommendation ranking, the improvement will be positive. Similarly, if the personalized approach moves the resource further down the ranking in the recommendation set, the improvement will be negative. Note that r_b is the same for every combination of query and resource, so the *imp* measure shows how good a given personalization technique is at moving "good" resources to the front of the list.

We found that the choice of tf or $tf * idf$ played an important role. In our evaluation, *tf* and *tfidf* have identical trends, but *tfidf* always provides superior results, so we have reported only results found based on those weights.

For comparison purposes we implemented the well-known k-means clustering algorithm. The k-means algorithm allows the partitioning of tag objects based on their attributes into a predefined number of groups. A predetermined number of clusters are randomly populated with tags. Centroids are calculated for each cluster, and tags are reassigned to a cluster based on a similarity measure. This process continues until tags are no longer reassigned.

For each algorithm, we experimented to determine the best tuning parameters. *Step* was found to be 0.004 for both datasets, the *generalization level* was found to be 8 and 4 for the del.icio.us dataset and Last.FM dataset respectively. The best value for division coefficient is 0.1 for del.icio.us data and 0.4 for Last.FM data. The best k (number of clusters) was found to be 350 for k-means.

For both k-means and hierarchical agglomerative clustering algorithms and their respective parameters, the average improvement across all folds and samples was averaged for both datasets and is reported in Figures 5 and 6. What we see is that the recommendation algorithm based on hierarchical agglomerative clustering provides a much more substantial improvement than the k-means-based algorithm. However, we also see that these differences are much more substantial in the Last.FM dataset. (Note the difference in the y-axis scales.)

We believe that one important factor distinguishing these datasets is their relative sparsity. The del.icio.us data set is approximately 400 times more sparse (99.9993% vs 99.7203%) than Last.FM. Data sparsity is known to reduce the effectiveness of clustering algorithms because denser datasets allow more precise measurements of the connections between tags, when generating tag clusters.

One reason for the relative density of the Last.FM dataset is its

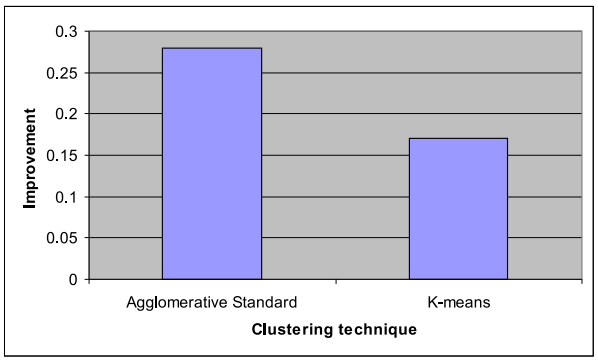

Figure 6: Comparison of clustering techniques for Last.FM dataset

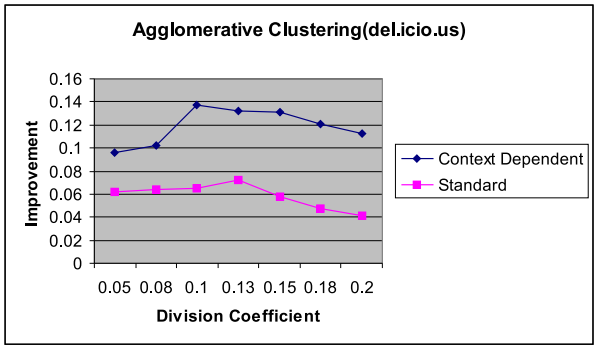

Figure 7: Comparison of the Recommendation Improvement in Standard and Context Dependent Algorithms for del.icio.us

focus in one specific topic domain, namely music. In contrast, the del.icio.us data set is much more diverse as it covers much wider range of topic areas.

One consequence of the narrow focus of Last.FM is seen in the distribution of tags. Only 10% of the users in the del.icio.us dataset use the same tag more than once to describe different resources. Approximately 50% of users in Last.FM do this. Such repeated tagging behavior creates a denser data set and more opportunities to find similarities between users, resources, and tags. k-means does not appear to be nearly as effective as the agglomerative algorithm. Bottom-up agglomeration appears to capture the semantics of tag associations more effectively than random selection of cluster seeds found in k-means.

5.3 Context-Dependent Cluster Selection

Figures 7 and 8 shows the agglomerative algorithm with the addition of the context-dependent cluster selection based on the query tag. It has been our hypothesis that selecting clusters on the basis of the user's query will improve accuracy by eliminating ambiguity: targeting the retrieval precisely to the meaning the user intended. Figure 7 shows that this effect holds true in the del.icio.us dataset. The context-dependent version of the algorithm is better across all choices of division coefficient. On the other hand, the differences between the two algorithms do not appear to be statistically significant for the Last.FM data.

Therefore, we see that the Last.FM data, due to its greater density and smaller degree of tag ambiguity, does not benefit from the context-dependent cluster selection technique, while the del.icio.us

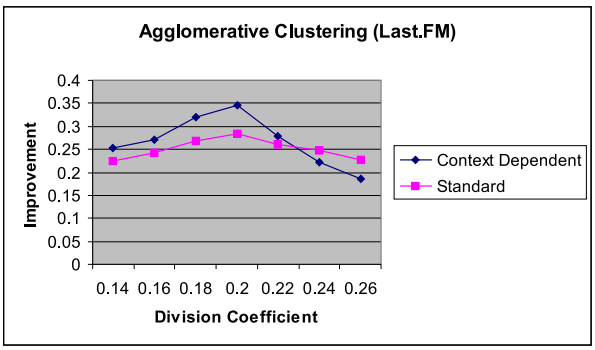

Figure 8: Comparison of the Recommendation Improvement in Standard and Context Dependent Algorithms for Last.FM

application, which suffers from greater sparsity and tags that can take on a variety of meaning across different topic areas, gets a boost from selecting clusters specific to the user's query. Thus, a major difference between the datasets that may explain this difference is the degree of generality in the topics covered by each folksonomy. Users of del.icio.us are tagging pages from the entire World-Wide Web, which spans nearly any topic imaginable. Users of Last.FM are tagging only music and related resources. We suspect that in a more focused folksonomy, the clusters are relatively tightly focused and selecting a subset of them does not yield better disambiguation.

6. CONCLUSION AND FUTURE WORK

In this article we showed that hierarchical agglomerative clustering of tags is an effective means to personalize navigational recommendations in collaborative tagging systems. Clusters of tags can be effectively used as a means to ascertain the user's interest as well as determine the topic of a resource. Clusters therefore provide an effective means to bridge the gap between users and resources.

The sparsity of the data significantly influences the quality of the clusters and therefore has a direct effect on the accuracy of recommendations. Both of our clustering techniques did much better on the denser Last.FM dataset. For sparse folksonomies like del.icio.us, we found that context-dependent cluster selection can further improve the recommendations by removing clusters which are not directly related to the user's query.

Our future work will analyze the other methods for modeling folksonomies such as probabilistic latent semantic analysis, principal component analysis, Apriori algorithm, decision trees, artificial neural networks, Bayesian learning, genetic algorithms and analytical learning for finding hidden factors influencing the user behavior.

In addition, we are interested in incorporating methods of natural language processing and semantic analysis for overcoming the problems of tag ambiguity and redundancy.

Finally, we plan to investigate ways of clustering users and resources in addition to tags and use those clusters for further improvement the recommendations in Folksonomies.

7. ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation Cyber Trust program under Grant IIS-0430303 and a grant from the Department of Education, Graduate Assistance in the Area of National Need, P200A070536.

8. REFERENCES

[1] S. Bao, G. Xue, X. Wu, Y. Yu, B. Fei, and Z. Su. Optimizing web search using social annotations. *Proceedings of the 16th international conference on World Wide Web*, pages 501–510, 2007.

[2] G. Begelman, P. Keller, and F. Smadja. Automated Tag Clustering: Improving search and exploration in the tag space. *Proc. of the Collaborative Web Tagging Workshop at WWW*, 6, 2006.

[3] H. Chen and S. Dumais. Bringing order to the Web: automatically categorizing search results. *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 145–152, 2000.

[4] J. Gemmell, A. Shepitsen, B. Mobasher, and R. Burke. Personalization in Folksonomies Based on Tag Clustering. *Intelligent Techniques for Web Personalization & Recommender Systems*, 2008.

[5] J. Gemmell, A. Shepitsen, B. Mobasher, and R. Burke. Personalizing navigation in folksonomies using hierarchical tag clustering. *Proceedings of the 10th International Conference on Data Warehousing and Knowledge Discovery*, 2008.

[6] T. Hammond, T. Hannay, B. Lund, and J. Scott. Social Bookmarking Tools (I). *D-Lib Magazine*, 11(4):1082–9873, 2005.

[7] C. Hayes and P. Avesani. Using tags and clustering to identify topic-relevant blogs. *International Conference on Weblogs and Social Media*, March 2007.

[8] P. Heymann and H. Garcia-Molina. Collaborative Creation of Communal Hierarchical Taxonomies in Social Tagging Systems. Technical report, Technical Report 2006-10, Computer Science Department, April 2006.

[9] A. Hotho, R. Jaschke, C. Schmitz, and G. Stumme. Information retrieval in folksonomies: Search and ranking. *The Semantic Web: Research and Applications*, 4011:411–426, 2006.

[10] A. Mathes. Folksonomies-Cooperative Classification and Communication Through Shared Metadata. *Computer Mediated Communication, LIS590CMC (Doctoral Seminar), Graduate School of Library and Information Science, University of Illinois Urbana-Champaign, December*, 2004.

[11] P. Mika. Ontologies are us: A unified model of social networks and semantics. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(1):5–15, 2007.

[12] D. Millen, J. Feinberg, and B. Kerr. Dogear: Social bookmarking in the enterprise. *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 111–120, 2006.

[13] S. Niwa, T. Doi, and S. Honiden. Web Page Recommender System based on Folksonomy Mining for ITNGŠ06 Submissions. *Proceedings of the Third International Conference on Information Technology: New Generations (ITNG'06)-Volume 00*, pages 388–393, 2006.

[14] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management: an International Journal*, 24(5):513–523, 1988.

[15] G. Salton, A. Wong, and C. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[16] C. Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann Newton, MA, USA, 1979.

[17] E. Voorhees. The TREC-8 Question Answering Track Report. *Proceedings of TREC*, 8:77–82, 1999.

[18] X. Wu, L. Zhang, and Y. Yu. Exploring social annotations for the semantic web. *Proceedings of the 15th international conference on World Wide Web*, pages 417–426, 2006.

[19] Z. Xu, Y. Fu, J. Mao, and D. Su. Towards the semantic web: Collaborative tag suggestions. *Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland, May*, 2006.

[20] R. Yan, A. Natsev, and M. Campbell. An efficient manual image annotation approach based on tagging and browsing. 2007.