# Construction of Disambiguated Folksonomy Ontologies Using Wikipedia

**Noriko Tomuro and Andriy Shepitsen**
DePaul University, College of Digital Media
243 S. Wabash, Chicago, IL USA
tomuro@cs.depaul.edu, ashepits@cdm.depaul.edu

## Abstract

One of the difficulties in using Folksonomies in computational systems is *tag ambiguity*: tags with multiple meanings. This paper presents a novel method for building Folksonomy tag ontologies in which the nodes are disambiguated. Our method utilizes a clustering algorithm called DSCBC, which was originally developed in Natural Language Processing (NLP), to derive *committees* of tags, each of which corresponds to one meaning or domain. In this work, we use Wikipedia as the external knowledge source for the domains of the tags. Using the committees, an ambiguous tag is identified as one which belongs to more than one committee. Then we apply a hierarchical agglomerative clustering algorithm to build an ontology of tags. The nodes in the derived ontology are disambiguated in that an ambiguous tag appears in several nodes in the ontology, each of which corresponds to one meaning of the tag. We evaluate the derived ontology for its *ontological density* (how close similar tags are placed), and its usefulness in applications, in particular for a personalized tag retrieval task. The results showed marked improvements over other approaches.

## 1 Introduction

In recent years, there has been a rapid growth in social tagging systems – so-called *Folksonomies* where users assign keywords or tags to categorize resources. Typically, the sources of folksonomies are web resources, and virtually any kind of information available on the Internet, ranging from web pages (e.g. Delicious (delicious.com)), scientific articles (e.g. Bibsonomy (www.bibsonomy.org)) to media resources (e.g. Flickr (www.flickr.com), Last.fm (www.last.fm)). Although tags in folksonomies are essentially semantic concepts, they have distinct characteristics as compared to conventional semantic resources which are often used in Natural Language Processing (NLP), such as WordNet (Miller, 1990). First, folksonomy tags are unrestricted – users are free to choose any words or set of characters to formulate tags. One significant problem arising from such free-formedness is *tag ambiguity*: tags that have several meanings (e.g. "Java" as coffee or a programming language or an island in Indonesia). Second, folksonomy tags are unstructured – tags assigned to a given resource are simply enumerated in a list (although oftentimes using a varying font size to indicate popularity), and no special organization or categorization of the tags is made (by the Folksonomy site). There have been several work recently which extracted structures from folksonomy tags and constructed ontologies (e.g. (Clough et al., 2005), (Schmitz, 2006)). However, most of them evaluate the effect of the extracted structures only in the context of specific applications, for instance generating user recommendations (e.g. (Shepitsen et al., 2008)).

In this work, we develop a novel method for constructing ontologies from folksonomy tags. In particular, we employ a clustering algorithm called *Domain Similarity Clustering By Committee (DSCBC)* (Tomuro et al., 2007). DSCBC is an extension of an algorithm called CBC (Pantel and Lin, 2002), and was originally developed for lexical semantics in NLP to automatically derive single/unambiguous word meanings (as *committees*) from ambiguous words. In this work, DSCBC is effectively adopted to derive disambiguated folksonomy tag committees, where a committee in this context is a cluster of tags in which the members share the same or very similar concept in one of their meanings. By using DSCBC, an ambiguous tag is identified as one which belongs to more than

one committee. One of the key ideas in DSCBC is the notion of *feature domain similarity*: the similarity between the features themselves, obtained a priori from sources external to the dataset used at hand. For example, if data instances x and y are represented by features f1 and f2, the feature domain similarity refers to the similarity between f1 and f2 (not between x and y). DSCBC utilizes this feature domain similarity to derive clusters whose domains are 'close', thereby producing unambiguous committees. In this work, we incorporate Wikipedia as the external knowledge resource, and use the similarity between Wikipedia articles to derive the committees of disambiguated tags. Finally using the tag committees derived by DSCBC, we build an ontology of tags by using a modified hierarchical agglomerative clustering algorithm. Ambiguous tags are mapped to several nodes in this ontology.

Note that in this paper, we refer to the structure derived by the hierarchical clustering algorithm as an 'ontology' instead of a 'taxonomy'. That is because, in the algorithm, the parent-child relation is determined by a similarity measure only, therefore sometimes does not correspond to the subsumption relation in the strict sense.

For evaluation, we construct an ontology from the Delicious tags, and measure the quality (*ontological density*) of the derived ontology by comparing with the ontologies obtained without using Wikipedia. We also use the derived ontology in a personalized information retrieval task. The results show that our method achieved marked improvements over other approaches.

## 2 Related Work

Several efforts have been made recently which focused on extracting structures from folksonomies. Clough (Clough et al., 2005) and Schmitz (Schmitz, 2006) derived hierarchical structures from image folksonomies (St. Andrew collection (specialcollections.st-and.ac.uk/photcol.htm) and Flickr, respectively). In addition to the hierarchical relation, they also derived other relations such as "type of", "aspect of", "same-as", etc. Mika (Mika, 2007) and Heymann (Heymann and Garcia-Molina, 2006) proposed an automatic creation of tags in folksonomy networks based on the tag co-occurrences among resources and users. They then used a graph clustering algorithm to connect tags which were used by the same users and for the same resources to identify tag 'clouds' and communities of like-minded users. However, none of those work used NLP techniques, nor did they deal with the tag ambiguity problem; Oftentimes, highly ambiguous tags are even removed from the data.

In our previous work (Shepitsen et al., 2008), we used a standard hierarchical agglomerative clustering algorithm to build a tag hierarchy. We also considered only the most popular sense of an ambiguous tag and ignored all other senses.

Wikipedia has been attracting much attention in the recent NLP research. For example, Wikipedia as a lexical resource was exploited for thesauri construction (Milne et al., 2006) and for word sense disambiguation (Mihalcea and Csomai, 2007). Other NLP tasks in which Wikipedia was utilized to provide contextual and domain/encyclopedia knowledge include question-answering (Ahn et al., 2004) and information extraction (Culotta et al., 2006). In a similar vein, (Gabrilovich and Markovitch, 2006) also used Wikipedia to improve the accuracy for text categorization. An interesting text retrieval application was done by Gurevych (Gurevych et al., 2007), in which Wikipedia was utilized to improve the retrieval accuracy in matching the professional interests of job applicants with the descriptions of professions/careers.

The work presented in this paper applies an NLP technique (the DSCBC algorithm), which incorporates the domain knowledge (Wikipedia) as a critical component, to the task of extracting semantic structure, in particular an ontology, from folksonomies. Our method is novel, and the experimental results indicate that the derived ontology was of high semantic quality.

## 3 Deriving Unambiguous Tag Committees

The DSCBC algorithm, which we had developed in our previous work (Tomuro et al., 2007), is an extension of *CBC Clustering* (Pantel and Lin, 2002), modified to produce unambiguous clusters when the data contained ambiguous instances. Assuming the instances are represented by vectors of features/domains, consider the following data:

```
    a b c d
x: 1 1 0 0
y: 1 0 1 0
z: 1 0 0 1
```

where $x$, $y$, $z$ are data instances, and a, b, c, d are features. In most clustering algorithms, features are assumed to be independent to each other, or their dependencies are ignored. So in the example, $x$ is equally likely clustered with $y$ or $z$, because the similarity between $x$ and $y$, and $x$ and $z$ are the same (based on the Euclidean distance, for example). However if we have a priori, general knowledge about the features that b's domain is more similar to that of c than to d, it is better to cluster $x$ and $y$ instead of $x$ and $z$, because the $\{x, y\}$ cluster is "tighter" than the $\{x, z\}$ cluster with respect to the domains of the features.

### 3.1 Feature Domain Similarity

In DSCBC, the general knowledge about the features is incorporated as a measure called *Feature Domain Similarity*: the similarity between the features themselves, obtained a priori from sources external to the dataset used at hand. In this work, we used Wikipedia as the external knowledge source, and as the features to represent the folksonomy tags. To this end, we first obtained the most recent dump of Wikipedia and clustered the articles to reduce the size of the data. We call such a cluster of Wiki articles a *Wiki concept*. Clustering was based on the similarity of the terms which appeared in the articles. Detailed descriptions of the Wikipedia data and this clustering process are given in section 5.1. Then given a set of folksonomy tags $T$, a set of folksonomy resources $R$ and a set of Wiki concepts $W$, we defined a matrix $M$ of size $|T| \times |W|$, where the rows are tags and the columns/features are Wiki concepts. Each entry in this matrix, for a tag $t \in T$ and a Wiki concept $w \in W$, was computed as the cosine between two term vectors: one for $t$ where the features are terms used in (all of) the resources in $R$ to which $t$ was assigned (by the folksonomy users), and another for $w$ where the features are terms used in (all of) the Wiki articles in $w$. Thus, the matrix $M$ contains the similarity values for a given tag to all Wikipedia concepts, thereby identifying the (Wikipedia) domains of the tag.

Using the matrix $M$, we define the feature domain similarity between two tags $f$ and $g$, denoted $fdSim(f, g)$, as:

$$fdSim(f,g) = \frac{\sum_i \sum_j f_i \times g_j \times cos(w_i, w_j)}{\sqrt{\sum_i f_i^2 \times \sum_i g_i^2}}$$

where $f_i$ is the similarity of the tag $f$ to the $i^{th}$ Wiki concept (and likewise for $g$), and $cos(w_i, w_j)$ is the cosine (thus similarity) between the $i^{th}$ and $j^{th}$ Wiki concepts. In this formula, the domain knowledge is incorporated not only through the way a tag is represented (as a vector of Wiki concepts), but also directly by $cos(w_i, w_j)$, the similarity between Wiki concepts themselves.

In addition to Feature Domain Similarity, we also incorporated a measure of *reference tightness* for folksonomy tags and Wiki concepts. This metric measures and takes advantage of the *link structure* in the folksonomy system as well as Wikipedia. For example, when a tag was assigned to several web pages in the folksonomy system, some of those pages may be reachable from each other through hyperlinks – in which case, we can consider the tag's domains are tight. Likewise for Wiki concepts, if a folksonomy tag is 'similar' to several Wiki concepts (for which the similarity value is above some threshold), some of those Wiki concepts may be reachable in the Wikipedia structure – then we can consider the tag's domains are tight as well. Furthermore, based on the notion of reference tightness within a set of resources, we define the connectedness between two sets of resources as the fraction of the resources (web pages or Wiki concepts) in one set which are reachable to resources in another set. We define the reference tightness between two sets of resources $S$ and $U$, denoted $srt(S, U)$, as follows.

$$srt(S,U) = \frac{\sum_{s \in S, u \in U} reach(s,u) + reach(u,s)}{\sum_{s \in S} nRef(s) + \sum_{u \in U} nRef(u)}$$

where $nRef(k)$ is the number of outgoing reference links in the resource $k$, and $reach(a, b)$ is an indicator function which returns 1 if any reference link from the resource in $a$ is reachable from any resource in $b$ or 0 otherwise. There are two terms in the numerator because the reachability relation is directional.

### 3.2 The DSCBC Algorithm

Using the notions of feature domain similarity and reference tightness, we define the similarity between two tags $f$ and $g$ as follows.

$$\begin{aligned} dsSim(f,g) &= \alpha \times fdSim(f,g) \\ &\quad + (1-\alpha) \times srt(R_f, R_g) \end{aligned}$$

where $R_f$ is the set of references from all web pages to which the tag $f$ is assigned, $srt(R_f, R_g)$ is the reference tightness between $R_f$ and $R_g$, and

$\alpha$ is a weighting coefficient. In our experiments (discussed in section 5), we set $\alpha$ to be 0.8 based on the results of the preliminary runs.

The DSCBC algorithm is shown in Algorithm 1. DSCBC is an unsupervised clustering algorithm which automatically derives a set of *committees*. A committee is a group of folksonomy tags which are very similar to each other. In Phase I, a set of preliminary tag clusters are first created. In Phase II, some of those tag clusters are selected as committees – those which are dissimilar/orthogonal to all other committees selected so far. Then in Phase III, each tag is assigned to committees which are similar to the tag. The $dsSim$ function is used in Phase I and II to measure the similarity between clusters and committees respectively. In Phase III, an ambiguous tag is assigned to one of more committees, where each time the features of the assigned committee are removed from the tag. Thus, ambiguous tags are identified as those which belong to more than one committee.

## 4 Building Folksonomy Tag Ontology

After obtaining the committees by DSCBC, we organize the tags into a ontology by using a modified hierarchical agglomerative clustering algorithm.[1] We first compute the pair-wise similarity between any two tags and sort those pairs according to the similarity values. Then we take the most similar pair and create the first cluster. Afterwards, we iterate through the whole tag/cluster pairs and substitute all instances in which either tag is a member, if the tag is not ambiguous, by the obtained cluster, and repeat the process until the list of pairs is empty. The committees derived by DSCBC are utilized to identify ambiguous tags – when a tag belonged to more than one committee. When we process an ambiguous tag, we first find its "core meaning" by finding the committee to which the tag is most similar, then remove all (non-zero) features that are encoded in committee from all instances left in the dataset. With this scheme, we can cover all senses of an ambiguous tag, for all such tags, during ontology generation. The similarity is computed using the $dsSim$ function described in the previous section; the only difference that, if one member of a pair is a cluster, it is rep-

[1]Our algorithm is essentially a modification of the Average-Link Clustering by (OConnor and Herlocker, 2001).

**Input**: Set of tags T. Tuning coefficients:
$n$ - number of the most similar tags chosen for the target tag
$q$ - number of features for finding the centroid
$\beta$ - similarity threshold for adding tags to committees
$\gamma$ - similarity threshold for assigning tags to committees
**Output**: Set of committees C. Set of tags T where each t $\in$ T is assigned to committees in C.
**Phase I.** Finding set of clusters $L$
**foreach** $t_i \in T$ **do**
  Select a set $k$ of $n$ most similar $t_j$ : i $\neq$ j
  add $k$ to L if it is not already in L.
**end**
**Phase II.** Find Communities $C$
**foreach** $c \in L$ **do**
  Find the centroid of $c$ using only $q$ features shared by most of tags in the cluster
  Add $c$ to $C$ if its similarity to every other cluster is lower than $\beta$
**end**
**Phase III.** Assign tags to committees
**foreach** $t \in T$ **do**
  Assign $t$ to committee $c$ in $C$ if the similarity is higher than $\gamma$
**end**
**Algorithm 1**: Clustering tags using DSCBC

resented by its centroid. Figure 1 shows an example folksonomy ontology. The modified hierarchical agglomerative clustering algorithm is shown in Algorithm 2.

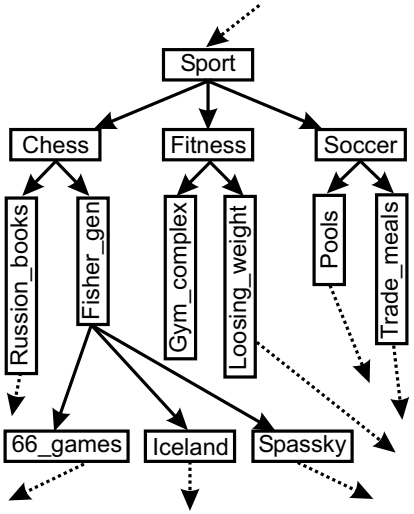


Figure 1: Example Folksonomy Ontology

**Input**: Set of tags T. Set of Committees C.
**Output**: An ontology of folksonomy tags.
L is a list containing pairs of tag/clusters with associated similarity, initially empty.
**foreach** $t_i \in T$ **do**
  Compute the similarity to all other tags $t_j$ (i $\neq$ j), and add a pair $\langle t_i, t_j \rangle$ in L.
**end**
**while** *L is not empty* **do**
  1. Sort $L$ by the similarity of the pairs.
  2. Pop the pair with the highest similarity from $L$. Let it $\langle t_i, \alpha \rangle$. $\alpha$ can be a single tag or a cluster of tags.
  3. Make $t_i$ the parent of $\alpha$.
  4. Join $t_i$ with $\alpha$, and create a new cluster $\beta$.
  **if** $t_i$ *belongs to more than one committee in C* **then**
    1. Find the committee $c$ which is the most similar to $t_i$.
    2. Remove all features intersecting with $c$ from $t_i$.
  **end**
  **else**
    1. Substitute all instances of $t_i$ in the pairs in L by $\beta$.
  **end**
**end**

**Algorithm 2**: Ontology Construction Algorithm

## 5 Experimental Evaluations

We applied our proposed algorithm to data from a real-world social tagging system Delicious and derived a tag ontology. Then we evaluated the derived ontology on two aspects: the *density* of the ontology, and the usefulness of the ontology in a personalized Information Retrieval (IR) task. Note that in the experiments, we determined the values for all tuning coefficients in the algorithms during the preliminary test runs.

### 5.1 Datasets

We first crawled the Delicious site and obtained data consisting of 29,918 users, 6,403,442 resources and 1,035,177 tags. In this data, 47,184,492 annotations were made by just one user, or for one resource, or by one tag. This distribution followed the Zipf's law – small numbers of tags were in frequent use and large numbers of tags were rarely used. Our intuitions were that the effect of using the semantic/encyclopedia knowledge from Wikipedia would probably be better reflected in the low frequency "long tail" part of the Zipf's distribution rather than the high frequency part. Likewise for users, we have discovered in our previous research that search personalization algorithms often produce different results for users with rich profiles and for users who have sparse profiles. This problem is known as the "Cold Start" problem in search personalization: a new user has very little information/history in the profile, therefore the system cannot reliably infer his/her interests. Since our experiments included a personalized IR task, we decided to extract two subsets from the data: one set containing high frequency tags assigned by users with rich profiles (randomly selected 1,000 most frequent tags entered by 100 high profile users), and another containing low frequency tags assigned by users with sparse profiles (randomly selected 1,000 least frequent tags entered by 100 sparse profile users). We refer to the former set as the "Frequent Set" and the latter set as the "Long Tail Set". The total number of resources in each dataset was 16,635 and 3,356 respectively.

Then for both datasets, we applied a part-of speech tagger to all resources and extracted all nouns (and discarded all other parts of speech). We also applied the Porter Stemmer (tartarus.org/∼martin/PorterStemmer) to eliminate terms with inflectional variations. Finally, we repre-

sented each resource page as a vector of stemmed terms, and the values were term frequencies.

As for Wikipedia, we used its English version available from BitTorrent Network (www.bittorrent.com). The original data (the most recent dump, as of 24 July, 2008) contained 13,916,311 pages. In order to reduce the size to make the computation feasible, we randomly chose 75,000 pages (which contained at least 50 words) and applied the Maximal Complete Link clustering algorithm to further reduce the size. After clustering, we obtained a total of 43,876 clusters, most of which contained one or two Wiki articles, but some of which had several articles. We call such a Wiki article cluster *Wiki concept*.

As with the tag datasets, for each Wiki article we applied the Porter Stemmer to reduce the number of the terms. Then we represented each Wiki concept page as a vector of stemmed terms, and the values were term frequencies.

### 5.2 Evaluation 1: Ontological Density

For the first evaluation, we evaluated the derived Delicious tag ontology directly by measuring the topological closeness of similar semantic concepts in the ontology. To that end, we developed a notion of *ontological density*: all tags assigned to a specific resource should be located close to each other in the ontology. For instance, a web resource *java.sun.com* in Delicious is assigned with various tags such as 'Java', 'Programming' and 'Technology'. Those tags should be concentrated in one place rather than scattered over various sections in the ontology. By measuring the distance as the number of edges in the ontology between tags assigned to a specific resource, we can obtain an estimate of the ontology density for the resource. Then finding the average density of all resources can give us an approximation of the overall density of the ontology's quality.

But here a difficulty arises for ambiguous tags – when a tag is ambiguous and located in several places in the ontology. In those cases, we chose the sense (an ontology node) which is the closest to the unambiguous tags assigned to the same resource. For example, Figure 2 shows a part of the ontology where an ambiguous tag 'NLP' (with two senses) is mapped: 1) Natural Language Processing (the left one in the figure), and 2) Neurolinguistic programming (the right one in the figure). The target web resource is tagged with three tags: two unambiguous tags 'POS' and 'Porter', and an ambiguous tag 'NLP'. To identify the sense of 'NLP' for this resource, we count the number of edges from the two unambiguous tags ('POS', 'Porter') to both 'NLP' tag nodes, and select the one which has the shortest distance. In the figure, the first sense has the total distance of 4 (= 2 edges from 'Pos' + 2 edges from 'Porter'), while the second sense has the distance 10 (= 5 edges from 'Pos' + 5 edges from 'Porter'). Therefore, we select the first sense ('Natural Language Processing') as the meaning of 'NLP' for this resource.

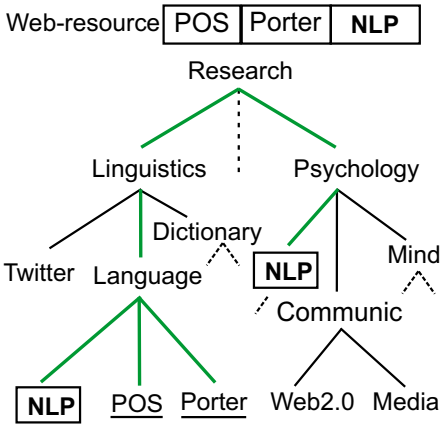


Figure 2: Example of Ambiguous Tags in the Ontology

Formally we define the density of the ontology $T$ for the set of resources $R$, denoted $Dens(T, R)$, as the average density over all resources in $R$, as follows.

$$Dens(T, R) = \frac{1}{|R|} \sum_{r \in R} density(r, T)$$

where $density(r, T)$ denotes the density for the given resource $r$ for the ontology $T$, defined as:

$$density(r, T) = \frac{nTags(r) - 1}{\text{argmin}_{i,j}\, dist(node(i, T), node(j, T))}$$

and $nTags(r)$ is the number of tags assigned to $r$, $node(k, T)$ is the node in $T$ for the $k^{th}$ tag (assigned to $r$), and $dist(n1, n2)$ is the number of edges between nodes $n1$ and $n2$ in $T$. So the density for the given resource is essentially the inverse of the minimum distance among the tags assigned to it. We computed the density value for the ontology derived by our approach ('Ontology Enhanced with Wiki Concepts') and compared with the ontologies obtained by using only the resources (where a tag vector is presented by

the stemmed terms in the resources to which the tag is assigned), and only the tags (where a tag vector is presented by the resource to which they were assigned). Figures 3 and 4 show the results, for the two datasets. For both datasets, the differences between the three ontologies were statistically significant (at p=0.05), indicating that the encyclopedia knowledge obtained from Wikipedia was indeed effective in deriving a semantically dense ontology.

Here, one observation is that the relative improvement was more significant for the "Frequent Set" than the "Long Tail Set". The reason is because frequent tags are generally more ambiguous than less frequent tags (as with words in general), therefore the effect of tag disambiguation by DSCBC was more salient, relatively, for the frequent tags.

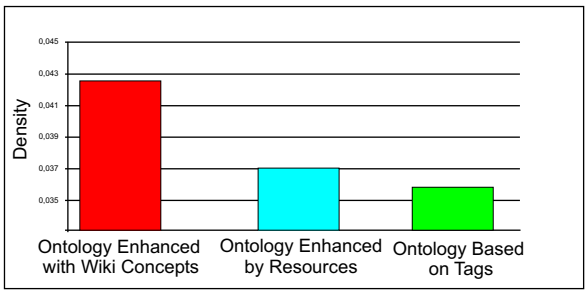


Figure 3: Ontological Density for "Frequent Set"

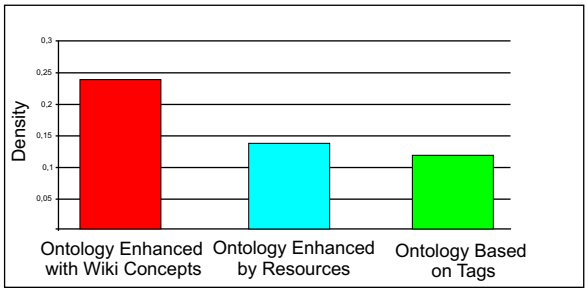


Figure 4: Ontological Density for "Long Tail Set"

### 5.3 Evaluation 2: Personalized Information Retrieval

For the second evaluation, we used the derived Delicious ontology in an IR task and measured its utility. In particular, we personalized the search results for a given user by utilizing the tag ontology as a way to present the user profile and infer his/her information needs.

Using the derived ontology, we search in the ontology for the query tag entered by a specific user. We first match the ontology with the user's profile and derive a score distribution for the nodes in the tree which reflects the user's general interest. To do so, we take each tag in the user's profile as the initial activation point, then spread the activation up and down the ontology tree, for all tags.

To spread activation from a given node, we use two parameters: *decay factor*, which determines the amount of the interest to be transfered to the parent/child of the current node; and *damping threshold* - if the interest score becomes less than this value we stop further iteration. Thus the resulting score distribution of the tree is effectively personalized to the user's general interest.

Using the obtained score distribution of a given user, we search the tree for a query tag (of this user). In the same way as the tags in the profile, we spread activation over the ontology from the node to which the tag belongs, but this time we add a weight to emphasize the relative importance of the query tag compared to the tags from the profile, because the query reflects the user's current information needs. Finally we feed the preference vector to the modified FolkRank algorithm (Hotho et al., 2006) to retrieve and rank the relevant web resources which reflect the user-specific preferences. Figure 5 shows the overall scheme of the personalized ranked retrieval using an ontological user profile.

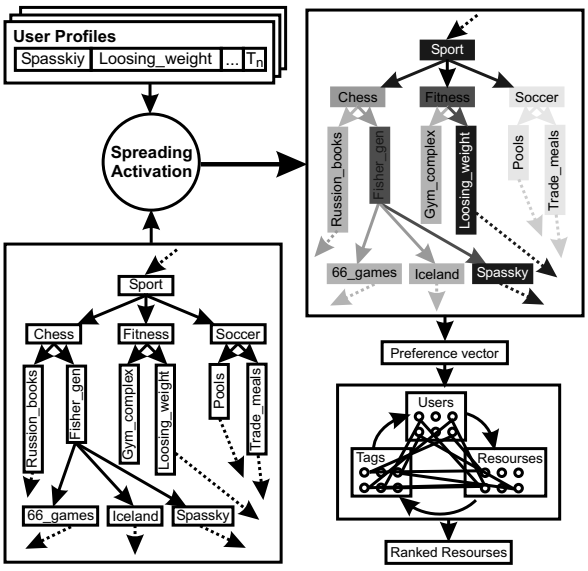


Figure 5: Ranked Retrieval in Folksonomies using Ontological User Profile

We evaluated the retrieval results by 5-fold cross validation. Given a test user profile, we used

the leave-one-out method for tags – we removed a target tag from the user profile and treated it as a query. All resources which the user assigned with that tag was the relevant set. For the final results, we computed the F-score, which is defined as standard:

$$F = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

Figure 6 and 7 show the F-scores for the two datasets. Note that 'TopN' indicates the top N retrieved resources. As you can see, the ontology enhanced with the Wiki concepts was able to better reflect the users' interest and produced significant improvements compared to the ontologies built only with the Delicious resources. Moreover, the improvements were much more significant for the "Long Tail Set" than the "Frequent Set", as consistent with our intuitions – Wikipedia's encyclopedia knowledge helped enhance the information about the less-frequent tags (assigned by the users with sparse profiles), thereby overcoming the "Cold Start" problem in search personalization.

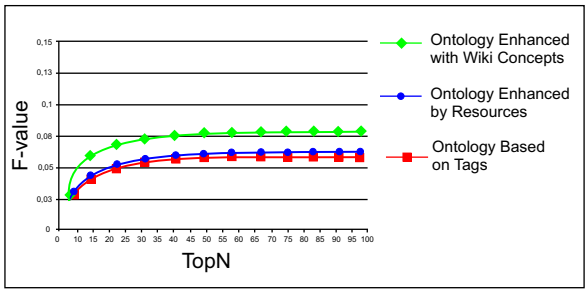


Figure 6: F-score of the Ontology for "Frequent Set"

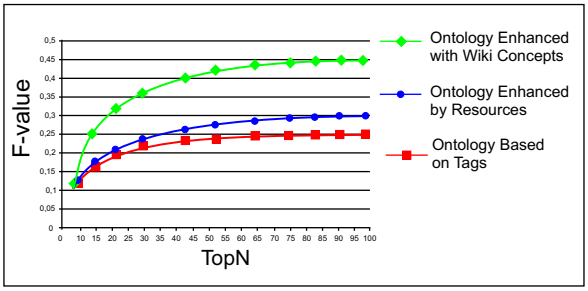


Figure 7: F-score of the Ontology for "Long Tail Set"

## 6 Conclusions and Future Work

In this paper, we presented a novel method for disambiguating tags and incorporating encyclopedia knowledge from Wikipedia in building folksonomy ontologies for social tagging systems. We applied our method to the data from Delicious and showed that, not only was the derived ontology semantically more dense (i.e., similar tags/concepts are clustered in close proximity), it also proved to be very effective in a search personalization task as well.

For future work, we are planning on investigating different ways of incorporating the link structures of Wikipedia and web pages in the tag similarity function (in DSCBC). Possible ideas include adding different weights on various types of links (or links appearing in various sections of a page/article), and using distance in the reachability relation, for example using the work done in Wikipedia Mining (Nakayama et al., 2008).

Finally, we are planning on applying information extraction or summarization techniques on Wikipedia articles to focus on sentences which provide relevant and important information about the subject.

## References

D. Ahn, V. Jijkoun, G. Mishene, K. Muller, M. DeRijke, and S. Schlobach. 2004. Using Wikipedia at the TREC QA Track. In *Proceedings of the 13th Text Retrieval Conference (TREC 2004)*.

P. Clough, H. Joho, and M. Sanderson. 2005. Automatically Organizing Images Using Concept Hierarchies. In *Proceedings of the SIGIR Workshop on Multimedia Information Retrieval*.

A. Culotta, A. Mccallum, and J.Betz. 2006. Integrating Probabilistic Extraction Models and Data Mining to Discover Relations and Patterns in Text. In *Proceedings of the Human Language Technology Conference*.

E. Gabrilovich and S. Markovitch. 2006. Overcoming the Brittleness Bottleneck Using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. In *Proceedings of the National Conference on Artificial Intelligence*.

I. Gurevych, C. Muler, and T. Zesch. 2007. What to be? - Electronic Career Guidance Based on Semantic Relatedness. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.

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

A. Hotho, R. Jaschke, C. Schmitz, and G. Stumme. 2006. Folkrank: A Ranking Algorithm for Folksonomies. In *Proceedings of the FGIR*.

R. Mihalcea and A. Csomai. 2007. Wikify!: Linking Documents to Encyclopedic Knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*.

P. Mika. 2007. Ontologies Are Us: A Unified Model of Social Networks and Semantics. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(1).

G. Miller. 1990. WordNet: An Online Lexical Database. *International Journal of Lexicography*, 3(4).

D. Milne, O. Medelyan, and I. Witten. 2006. Mining Domain-Specific Thesauri from Wikipedia: A Case Study. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*.

K. Nakayama, T. Hara, and S. Nishio. 2008. Wikipedia Mining - Wikipedia as a Corpus for Knowledge Extraction. In *Proceedings of Annual Wikipedia Conference (Wikimania)*.

M. OConnor and J. Herlocker. 2001. Clustering Items for Collaborative Filtering. In *Proceedings of SIGIR-2001 Workshop on Recommender Systems*.

P. Pantel and D. Lin. 2002. Discovering Word Senses from Text. In *Proceedings of the 8th ACM Conference on Knowledge Discovery and Data Mining (KDD-02)*.

P. Schmitz. 2006. Inducing Ontology From Flickr Tags. In *Proceedings of the Collaborative Web Tagging Workshop (WWW 06)*.

A. Shepitsen, J. Gemmell, B. Mobasher, and R. Burke. 2008. Personalized Recommendation in Social Tagging Systems Using Hierarchical Clustering. In *Proceedings of the 2008 ACM conference on Recommender Systems*.

N. Tomuro, S. Lytinen, K. Kanzaki, and H. Isahara. 2007. Clustering Using Feature Domain Similarity to Discover Word Senses for Adjectives. In *Proceedings of the 1st IEEE International Conference on Semantic Computing (ICSC-2007)*.