

# Collective Collaborative Tagging System

Jong Youl Choi, Marlon Pierce  
Department of Computer Science  
Community Grid Lab  
Indiana University at Bloomington  
Email: jychoi@cs.indiana.edu

*Abstract—*

## I. INTRODUCTION

Motivation is two fold.

- Collaborative tagging, also known as social tagging, is a system to collect knowledge from the people and the quality of knowledge users can get will increase as the quantity of data people provided grows. Currently in the Internet lots of collaborative tagging sites exist but there is no way to integrate the data from the multiple sites to form a large and unified set of collaborative data from which users can have more accurate and richer information than from a single site.
- During the recent development of information retrieval (IR) and machine learning technology, lots of IR algorithms have been well studied and open to public. Although most of the collaborative tagging sites provide various searching services, their algorithms are closed to public and somewhat secret to the users. Furthermore, most of them provide only one type of searching algorithm and the users have no choice to apply various other IR algorithms to find the best information available from the data. Using the same data set with various different searching algorithms, users can have more possibilities to discover hidden information varied in the data set.

The purpose of this paper is i) introducing a new collaborative tagging system which can collect tag data from other repositories and merge them in order to provide better quality of knowledge and comparing commonly used algorithms for the folksonomy analysis.

## II. A NEW SYSTEM

Motivated from the above observations, we propose a *collective collaborative tagging* (CCT for short) system which can provide various collaborative tagging services in a uniform way to users. Our CCT system is designed to provide the following key functions.

- Importing data from multiple sources to build a large and unified tag repository
- Query services with options to run various IR algorithms
- Query services with options to run with different data sources and parameter settings

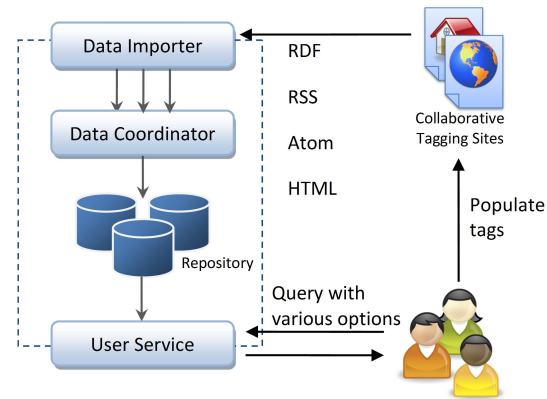


Fig. 1. Overview of Collective Collaborative Tagging (CCT) System

### A. Architecture

The system consists of three main components; *data importer*, *data coordinator*, and *user service* (Figure 1).

Details of main three components are as follow.

- **Data Importer:** Importing tagging data with machine readable format such as RDF, RSS, Atom or Web APIs from number of different collaborative tagging sites. Importing can be done asynchronously or synchronously.
- **Data coordinator:** Merging data from different sources and storing them into a uniform repository. The coordinator will resolve possible format conflicts and duplication problem which may exist in multiple sites.
- **User service:** Providing various machine learning based searching algorithms and options users can choose to run as a form of Web service API. The queries will be performed with the unified repository which stores tagging data collected from different collaborative tagging systems

### B. Service Type

Various kinds of user requests to extract information from the annotated data can exist in collaborative tagging system; for example, searching items by using tags, getting personalized recommendations based on user's profiles or past activities, discovering group of users or communities sharing similar interests, just to name a few. Those demands can be generally categorized into 4 classes and our CCT system will provide services to support those requests. The following

TABLE I

GENERAL TYPES OF SERVICE IN COLLABORATIVE TAGGING SYSTEMS

Type	Name	Description
I	Searching	For a set of given tags as an input, find the most relevant objects (documents, items, users, or tags)
II	Recommendation	Create a recommendation list of objects which a user hasn't observed yet. User's profile or past activity information can be used.
III	Clustering	Find communities or groups of users or objects based on the similarity.
IV	Trend detection	Detect interesting or abnormal tagging behavior in time series analysis manner.

classification is not exclusive but rather overlapping in some sense.

**Type I – Searching by tags :** For a given set of tags as an input, searching the most relevant objects with the input tags is an essential function in the collaborative tagging system. Generally the objects can be either documents, items, users, or anything annotated by tags in the system. Results will be returned to users in an ordered fashion based on some computed scores.

**Type II – Recommendation :** With no explicit input of tags, the system will return a recommendation list of objects. While the input tags used in searching by tags should be explicitly defined by a user, in recommendation those are generated implicitly by the system, based on user's previous activities, preferences, or profiles. For an example, the system can give to a user a recommendation list of documents which haven't been discovered by the user, based on the user's past tagging activities. Also, recommendation of tags is possible when a user wants to annotate a document for the first time, the system can recommend other co-used tags with his initial input.

**Type III – Clustering :** This is so called community discovery. Not only searching for the most relevant objects, it is also useful finding a group or a community which shares more common interests expressed by tags within the group members than with others.

**Type IV – Trend Detection :** The system analyzes the tagging activities in time-series manner and detect interesting patterns of tagging or abnormality among the tag data set.

More specific examples of service types or information users can get for each category are summarized in Table I.

In the following section we discuss how those services can be implemented by using various machine learning algorithms.

### III. MODELS FOR TAG ANALYSIS

A collaborative tagging system is designed to utilize the power of peoples knowledge and provide an efficient way of searching information from the collaboratively annotated data set. In this way, the system can help users to find the information with more efficiency and discover unexposed or hidden information buried under piles of information. Thus, developing efficient models and algorithms for searching is

$$A = \begin{matrix} & d_1 & d_2 & d_3 \\ \begin{matrix} t_1 \\ t_2 \\ t_3 \\ t_4 \end{matrix} & \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{pmatrix} \end{matrix}$$

Fig. 2. An example

the key step for building a successful collaborative tagging system. In this section, we discuss the models for developing folksonomy searching engines and various algorithms for searching and tag analysis.

#### A. Models

For building an efficient searching engine for folksonomies, the way to represent folksonomy data is an important issue. In the field of Information Retrieval (hereafter IR for short), two models – *the vector space model* and *the graph model* – have been widely used and they are both well applicable in folksonomy indexing.

Although both models are sharing many similar aspects, they are distinct in many practical points of views. As examples, the Latent Semantic Indexing (LSI) (we will discuss details of this algorithm later) is using the vector space model for indexing and measuring pairwise similarities between objects, and the famous ranking algorithm PageRank used by Google and its variant TagRank for folksonomy searching are based on the graph model. While the vector space model has been widely used in many areas due to its simplicity, not many researches have been conducted for the use of the graph model so far.

*1) Vector space model:* In the vector space model, also known as bag-of-words model, each object can be represented as an unordered collection of tags and by using mathematical notation a vector can be used. I.e., an object  $d_j$  can be represented as a  $q$ -dimension column vector  $(w_{1j}, \dots, w_{qj})$ , where  $q$  equals the total number of distinct tags in the system and  $w_{ij}$  is a weight of the occurrence of the tag  $t_i$  (We will discuss various weight schemes shortly). Thus, the whole collection of  $n$  objects can be represented as a matrix  $A \in \mathbb{R}^{q \times n}$  where each column corresponds to  $d_j$ . An example is shown in Figure 2.

*2) Graph-based model:* Although the vector space model is simple and easy-to-use, sometimes it lacks the ability to describe object-object relationships, which is more easier in the graph model. In the graph model, folksonomies can be represented as a network of connections, also known as *tag graph*, which consists of objects as nodes and connections between objects as edges. An example is shown in Figure 3.

More specifically, a tag graph is a undirected tripartite graph  $G = (V, E)$  where nodes in  $V$  are one of objects in disjoint subsets of three entities – objects, tags, and users – and edges exist only between three entities. Each edge will be added for a single transaction, i.e., annotating an object with a set of tags by a user.

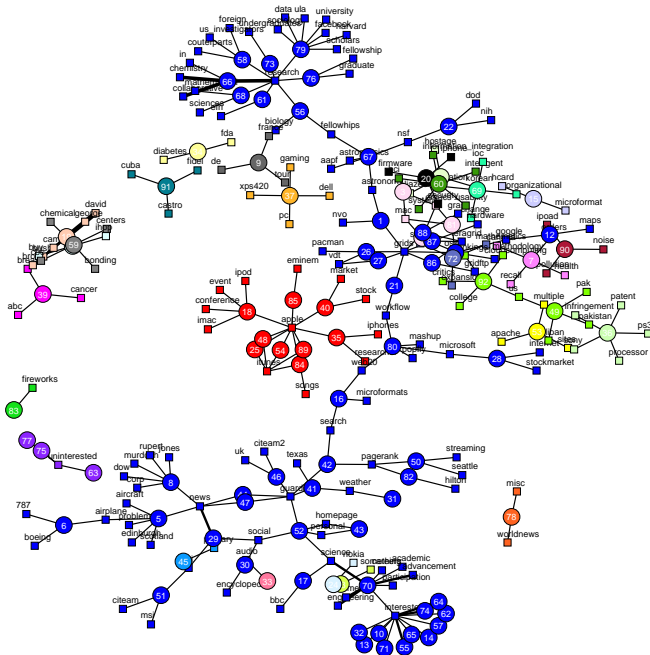


Fig. 3. An example of tag graph. The data used in this figure obtained from our in-house collaborative tagging system, MSI-CIEC portal (See Table III)

## B. Similarity Measurement

Measuring similarity between two objects is a key step in folksonomy analysis and it is directly related to the performance of the system. Although it is possible in folksonomy analysis to measure various types of similarities such as object-object, object-tag, object-user, user-tag, and user-user, in this paper we only consider object-object similarity for simplicity. The other measurements can be easily estimated by using the same manner.

1) *Weight Measurement*: Weight measurement is a scheme to quantify the weight element  $w_{ij}$  of the tag-object matrix  $A$  for each  $1 \leq i \leq q$  and  $1 \leq j \leq n$ . A simple minded approach is to count the occurrence of the tag  $t_i$  for the object  $d_j$ , which is known as Term Frequency (TF for short). As observed in many IR researches, however, this approach has an disadvantage to utilize the low frequency terms or tags. Tag distributions in folksonomies usually follows the Zipf's power law where a few majority tags govern the most of distributions and thus minor tags can lost their importance in many searching algorithms. Thus, some normalization scheme should be used to avoid this problem and to collect more variety information by exploiting minor tags in folksonomies.

Various schemes have been suggested in many IR literatures but the most popular scheme is Term Frequency-Inverse Document Frequency (TF-IDF for short) which is the multiplication of TF and IDF. In a nutshell, term frequency  $tf_{ij}$  is the number of tagged term  $t_i$  for document  $d_j$  and the document frequency  $df_i$  is the number of documents having the same tag  $t_i$ . IDF is computed by  $\log \frac{n}{df_i}$  for the total number of document  $n$  and

thus TF-IDF equals  $tf_{ij} \times \log \frac{n}{df_i}$ . Formulas are summarized in Table II.

2) *Similarity Measurement*: Similarity measurement is to measure a degree of likeness between two tagged objects in folksonomies. In the vector space model, various similarity measurement schemes have been developed in the field of IR and in practice three similarity measurement schemes are the most popular among them: Cosine, Jaccard, and Pearson [1] (summarized in Table II).

While in the vector space model such similarities are measured by geometric characteristics (such as cosine angles) or statistical ways (such as Jaccard and Pearson), similarities in the graph model can be measured by graph theoretic properties, such as hop distances, shortest paths, maximum flows, and so on.

Pairwise similarity is also an important measurement for using in finding groups or communities. Note, however, measuring pairwise similarity is also different in both models. In vector space mode, all object-object similarities can be directly computed from the tag-object matrix  $A$ ; I.e., in the vector space model, we can compute a pairwise similarity matrix  $D = [\delta_{ij}] \in \mathbb{R}^{n \times n}$  and its entries  $\delta_{ij}$  by computing the similarity between any two objects  $d_j$  and  $d_k$  among total  $n$  documents. Thus, the computation cost to build  $n \times n$  pairwise similarity matrix  $D$  is  $\mathcal{O}(n^2)$

However, in the graph model we cannot compute pairwise similarities directly from the matrix  $A$  but, instead, we should do this iteratively; Firstly, compute only similarities of directly connected objects, i.e., objects sharing at least one common tag between them, and then, measure similarities of the others, which have no direct connections, by means of discovering paths between them. Path discoveries can be done by using the algorithms for finding the shortest path. Floyd-Warshall algorithm [2] is well known for this problem and this requires generally  $\mathcal{O}(n^3)$  computations.

## IV. ALGORITHMS

Currently numerous algorithm have been studied for supporting various types of services in collaborative tagging systems and this is also very active research area. In this section, we focus on core algorithms which can successfully support our service classification as shown in Table I.

### A. Latent Semantic Indexing

The Latent Semantic Indexing (hereafter LSI for short) has been widely used for indexing the Web pages or documents in libraries and served as one of the most popular searching algorithms based on the vector space model. The LSI algorithm can be also used in folksonomies as a searching engine to support the Type-I service in the vector space model. Using the tag-object matrix collected in the system as an input, the LSI algorithm can help to recover underlying or latent structures of folksonomies, often obscured by noisy data, and enable to find the true relationship between tags and objects without noises based on the statistical information.

TABLE II  
EQUATIONS USED FOR MEASURE WEIGHTS AND DISSIMILARITIES. SLIGHTLY MODIFIED FROM ORIGINAL EQUATIONS.

Abbr	Name	Definition
TF $tf_{ij}$	Term Frequency	The number of tagged term $t_j$ for document $d_i$
DF $df_j$	Document Frequency	The number of documents having the same tag $t_j$
TF-IDF $tfidf_{ij}$	TF-Inverse DF	$tf_{ij} \times \log \frac{n}{df_j}$ where $n$ is the total number of $d_i$
$COS(d_i, d_j)$	Cosine	$\frac{\sum_k w_{ik} w_{jk}}{\sqrt{\sum_k w_{ik}^2} \sqrt{\sum_k w_{jk}^2}}$
$JAC(d_i, d_j)$	Jaccard	$\frac{\sum_k w_{ik} w_{jk}}{(\sum_k w_{ik}^2 + \sum_k w_{jk}^2 - \sum_k w_{ik} w_{jk})}$
$PEA(d_i, d_j)$	Pearson	$\frac{(\sum_k w_{ik} w_{jk} - \frac{1}{q} \sum_k w_{ik} \sum_k w_{jk})}{\sqrt{(\sum_k w_{ik}^2 - \frac{1}{q} (\sum_k w_{ik})^2) (\sum_k w_{jk}^2 - \frac{1}{q} (\sum_k w_{jk})^2)}}$

The core idea of LSI algorithm is that since the dimension of the raw or untreated tag-object matrix is usually too high to find the concise relationships between tags and objects, the dimension should be reduced to recover latent structures of the input matrix. Thus, the algorithm projects the tag-object matrix  $A = [a_{ij}] \in \mathbb{R}^{q \times n}$  in the  $n$ -dimension space onto a lower dimension space  $d$  such that  $d \ll n$  in order to remove “noisy” information and recover the true relationships. In this sense, the LSI algorithm can be considered as a dimension reduction algorithm from  $n$ -dimension to  $d$ -dimension.

For dimension reduction processing, the LSI uses the Singular Value Decomposition (SVD) method to find the best lower dimension matrix  $\hat{A}$  of the raw matrix  $A$  as an input in a way to make the 2-norm difference  $\|A - \hat{A}\|_2$  minimized.

1) *Preprocedure*: The LSI algorithm finds the best projections of the input tag-object matrix  $A$  onto a lower dimension or latent space by using SVD. Compute the decomposition of the input matrix  $A$  by using the SVD,

$$A = U \Sigma V^T \quad (1)$$

where  $U$  and  $V$  are orthogonal matrices (i.e.,  $UU^T = V^T V = I$ ) and  $\Sigma$  is a diagonal matrix having  $n$  eigen values such as  $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_n)$  and  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$ .

Choose a target lower dimension  $d$  such as  $d \ll n$  and define a new reduced diagonal matrix  $\hat{\Sigma}$  from  $\Sigma$  by removing  $\sigma_{d+1}, \dots, \sigma_n$ , such that  $\hat{\Sigma} = \text{diag}(\sigma_1, \dots, \sigma_d)$  where  $d < n$ . Similarly, compute  $\hat{U}$  and  $\hat{V}$  by removing  $(d+1, \dots, n)$ th columns. Then,  $\hat{V}$  represents the new object coordinates in the reduced space.

Note that the new matrix in the lower dimension is

$$\hat{A} = \hat{U} \hat{\Sigma} \hat{V}^T \quad (2)$$

and  $\hat{A}$  is the best approximation of the matrix  $A$  in a sense that 2-norm difference  $\delta = \|A - \hat{A}\|_2$  is minimized.

2) *Queries*: A query  $q$  is given by a vector of tags such that  $q = (q_1, \dots, q_t)$ . By using the reduced matrices above, Compute  $\hat{q}$ ,

$$\hat{q} = q^T \hat{U} (\hat{\Sigma})^{-1}, \quad (3)$$

and compare  $\hat{q}$  with each document (i.e, each row of  $\hat{V}$ ) in the reduced space by measuring similarity. Objects having the highest similarities are the answers of the query.

TABLE III  
DATA SETS USED IN OUR EXPERIMENTS

Data Sets	Documents	Tags	Remarks
MSI-CIEC portal	92	178	In-house system
Connotea	1131	6071	Harvested from Connotea

### B. FolkRank

Inspired from the PageRank algorithm which exploits the network structures of Web pages, the FolkRank algorithm has been developed as a folksonomy search engine by using the graph model. The FolkRank algorithm can be used to provide Type-1 service by using the graph model.

The FolkRank algorithm uses the weight spread approaches, which is the same strategy used by the PageRank algorithm. The intuition is that the popularly tagged objects will receive more and more weights from the neighbor objects. The difference from PageRank, however, is that weights are spreading out through the undirected edges of the tag graph which is the characteristic in the folksonomy graph model, while the weight spreads have a direction in the PageRank.

The FolkRank algorithm as follows: First, build a tripartite graph from the folksonomy data. Second, spread weights iteratively by using the following equation.

$$w = dAw + (1 - d)p \quad (4)$$

### C. Clustering

To be added... (k-means and Deterministic Annealing)

## V. EXPERIMENTS

For the experiments in this paper, we used two sets of folksonomy data: one from our in-house collaborative tagging system called MSI-CIEC portal, which is currently under development, and the other harvested from Connotea, one of the well-known folksonomy systems. The Connotea data was obtained in January 2008 and only collected approximately 1000 documents in the most popular document list and their related tags. The data used in this experiment is summarized in Table III

### A. Latent Semantic Indexing

To be added...

## B. Clustering

To be added...

## VI. CONCLUSION

### REFERENCES

- [1] K. Boyack, R. Klavans, and K. Börner, "Mapping the backbone of science," *Scientometrics*, vol. 64, no. 3, pp. 351–374, 2005.
- [2] R. Floyd, "Algorithm 97: Shortest path," *Communications of the ACM*, vol. 5, no. 6, p. 345, 1962.