CTSC: Core-Tag oriented Spectral Clustering Algorithm on Web2.0 Tags[1]

Yexi Jiang, Changjie Tang, Kaikuo Xu, Yu Chen, Liang Tang
Database and Knowledge Engineering lab,Department of Computer Science Sichuan University, China
{jiangyexi, tangchangjie}@cs.scu.edu.cn

Abstract—**With the rapid development of the Web2.0 communities, many researchers have been attracted by the concept of folksonomy from the field of data mining and information retrieval. Finding out semantic correlation of tags is avid requirement for web2.0 application. However, no proper algorithm can tackle this task very well. This paper proposes a core-tag oriented clustering method to handle the task. The main contributions include: (1) Proposing the concept of core-tag oriented space; (2) Proposing a method called Core-Tag oriented Spectral Clustering (CTSC) to cluster tags in the new space; (3) Designing experiments to evaluate the algorithm, and the results show that CTSC algorithm performs well on clustering tags.**

Keywords-Core-Tag; Spectral Clustering;

I. INTRODUCTION

With high speed development of Web information technology, the number of web pages grows explosively. **Taxonomy** is one of the traditional top-down methods to manage and category web pages. However, it cannot reflect the true opinions of users since the taxonomy hierarchy is created by field experts. According to the requirement of the users, literature [1-2] proposed a new concept named **folksonomy**. Its key idea is that users can add preference tags to the web resources. Folksonomy quickly gains ground by its ability to recruit the activity of web users into effectively organizing and sharing vast amount of information. Tag is the core of folksonomy. The formal definition of folksonomy can be seen in [3]. It's a kind of tripartite graph $\boldsymbol{G_F} = (\boldsymbol{V}, \boldsymbol{E})$, with $\boldsymbol{V} = \boldsymbol{U} \cup \boldsymbol{T} \cup \boldsymbol{R}$, $\boldsymbol{U}$ is the set of users, $\boldsymbol{T}$ is the set of tags and $\boldsymbol{R}$ is the set of resources. The users can annotate the resources with tags. Taking advantage of collaborative intelligence, the result of folksonomy represents the users' true opinions of the resources. The core of folksonomy is the tags defined by users. It would cause a large number of different tags for a similar resource, an efficient clustering mechanism to put tags together is necessary for users. They can get related tags by giving query tag. We call the query tag *core-tag*.

Traditionally, when user launches a query, there are two ways of returned results: (a) a mass of related tags without order. (b) a number of ordered tags. If the user only needs to know which tags are related tags of target tag, the first way is enough; if the user wants to know the detail similarity value between target tag and related tags, the second way is enough. However, in practice, users usually need the information more detail than case one but more general than case two. If two related tags are very close, there is no need to distinguish them. Only tags that are distant from each other should be distinguished. For case one, the detail relationships of tags cannot be provided. Case two provides too much detail, which user doesn't need. Obviously, both of the two methods cannot fulfill the requirement, and Example 1.1 illustrates these cases.

Example 1.1: John likes bobcat. He is interested in the relationship between animals. He is not an expert in Zoology, so he only needs to know a general correlation. There are three types of results. In Fig. 1, the left result is too general while the middle one is too trivial. For the right result John can know 'Jungle Cat' and 'Sand Cat' are the closest to '*Bobcat*', then '*Lynx*', '*Leopard Cat*', '*Ocelet*' etc.. More details in the middle one provide meaningless information to John. □

Target tag in final result is called core-tag. In this study, we present core-tag oriented clustering method that can meet the requirement mentioned above.

Our contributions includes: (1) Introduce the concept of core based space (2) Propose the Core-Tag oriented Spectral Clustering algorithm (CTSC) based on spectral learning to cluster tags in the new space. (3) Design experiments to evaluate the algorithm.

The rest of the paper is organized as follows. Section 2 describes the related work. Section 3 proposes the method of CTSC. Section 4 evaluates and gives an analysis to the new method. Section 5 concludes this paper and introduces some future work.

[1] *Supported by the 11th Five Years Key Programs for Sci. & Tech. Development of China under grant No. 2006BAI05A01, the National Science Foundation under grant No. 60773169, the Software Innovation Project of Sichuan Youth under grant No. 2007AA0155.

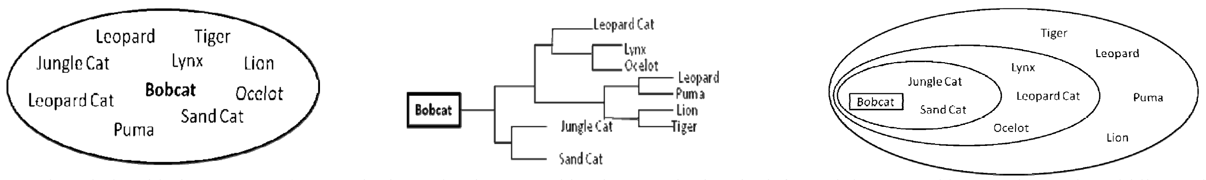


Fig 1. The relationship between '*Bobcat'* and other animals returned by three methods. The left result is returned by cased one; the middle one is returned by case two; the right one is John's needs.

II. RELATED WORKS

Spectral algorithm is one of the most successful algorithms for graph partitioning and matrices and it has been widely used in many fields of scientific applications. It was first applied in graph partitioning field in literature [7]. This method was created by Donath and Hoffman [4] who first suggested using the eigenvectors of adjacency matrices of graphs to find partitions. Fiedler proposed splitting the vertices of the corresponding eigenvector of the second-smallest eigenvalue of the Laplacian matrix to partition the graph (See [5]). It has been proved to work well on bounded-degree planar graphs and finite element meshes in [10]. [11] mentioned there exists different views on which eigenvectors to use, experiments observed that using more eigenvectors and directly computing the graph into k partition can get better performance. (See [12])

Recent years, spectral algorithm has been extended to use in wider fields. Literature [6] presented a random walks view of spectral clustering and segmentation method based on Markov Transition Process, and it provides a probabilistic foundation to the spectral methods for clustering and segmentation. Literature [8] provided a framework of spectral clustering algorithm, and it gives a very detail analysis on the situation of ideal case and general case of the algorithm. Furthermore, it gave the reason why this method can actually get reasonable clusters. The author further presents under which condition it can be expected to work well. Literature [9] presented a good spectral learning framework both for spectral clustering and spectral classification. Our algorithm built on the idea of spectral clustering for tag clustering.

III. CSTC ALGORITHM

We proposed core-tag oriented clustering method to fulfill the requirement mentioned in Example 1.1. For core-tag clustering, a formal definition of core-tag clustering is as follows:

Definition 1 (Core-tag Clustering): Given a core-tag t_c and tag set $\boldsymbol{S}$ that contains related tags of t_c. $\boldsymbol{S}$ is got by the result of similarity measurement between tags. Find clusters $\boldsymbol{C_1}$, $\boldsymbol{C_2}$ … $\boldsymbol{C_n}$ which contain the related tags, let $\boldsymbol{C}$ denotes the set of clusters. The result should satisfy three conditions:

- $\cup \boldsymbol{C} = \boldsymbol{T_{cond}}$, and $\boldsymbol{T_{cond}} \subseteq \boldsymbol{S}$. Variable $\boldsymbol{T_{cond}}$ denotes the selected related tags.
- For $\forall \boldsymbol{C}_j \in \boldsymbol{C}$, have $\boldsymbol{C_j} \neq \emptyset$ and $\boldsymbol{C} = \boldsymbol{C_1}$□$\boldsymbol{C_2}$…□$\boldsymbol{C_n}$. Variable $\boldsymbol{C_i}$ denotes a certain cluster.
- For $\forall \boldsymbol{C}_i, \boldsymbol{C}_j \in \boldsymbol{C}$, have either dist($t_c$, $\boldsymbol{C_i}$) < dist(t_c, $\boldsymbol{C_j}$) or dist(t_c, $\boldsymbol{C_i}$) > dist(t_c, $\boldsymbol{C_j}$). Function dist() denotes the distance from the core-tag to a cluster by considering the distance of all tags in the cluster.

Based on the concept of core-tag clustering, we present CTSC algorithm and the key steps are illustrated as follows:

Step 1: Generate $\boldsymbol{S}$ and then $\boldsymbol{T_{cond}}$ for core-tag. The detail of $\boldsymbol{S}$ and $\boldsymbol{T_{cond}}$ will be explained in later part.

Step 2: Generate similarity matrix by $\boldsymbol{T_{cond}}$ and convert the matrix into distance matrix. This step maps related tags into the space with core-tag located at the origin.

Step 3: Utilize spectral learning method to transform the distance matrix. Through this step, the tags with different groups are separated from each other.

Step 4: Employs any suitable clustering algorithm to cluster the transformed matrix.

3.1 Generate Tag Sets

In core-tag clustering, eight measurements are used to evaluate similarity between tags, they are: Augmented Expected Mutual Information, Simrank, Pythagorean Theorem, Pessimistic Similarity, Cosine similarity, TF/IDF and adjusted Cosine similarity, Pearson coefficient. We only compute tags pair which they tag same resource exceeding certain times. If a tag is deemed as related tag of t_c by any of the measurements, it is put into $\boldsymbol{S}$. $\boldsymbol{S}$ are used to generate $\boldsymbol{T_{cand}}$. Definition 2 is the formal definition of $\boldsymbol{T_{cand}}$.

Definition 2 ($\boldsymbol{T_{cand}}$): Let t_c be a giving core-tag, The candidate tag set $\boldsymbol{T_{cand}}$ is defined as follows:

(1) According to t_c, a group of measurements $\mu = \{m_1, m_2 \ldots m_n\}$ are employed to find $|\mu|$ groups of correlated tags $\boldsymbol{T} = \{\boldsymbol{T_1}, \boldsymbol{T_2} \ldots \boldsymbol{T_n}\}$.

(2) The candidate tag set $\boldsymbol{T_{cand}}$ satisfies: $\boldsymbol{T_{cand}} = \{ t \mid t$□ $\cap \boldsymbol{T_i} \wedge T_i$□$\boldsymbol{T} \}$.

Note that, tags contained in all measurements are put into $\boldsymbol{T_{cand}}$. These common tags are related to t_c with high confidence because all similarity measurements deem them as correlated tags to the core-tag. Through extracting, we get a tag set $\boldsymbol{T_{cand}}$ of t_c.

For any two tags in $\boldsymbol{T_{cand}}$, their similarity values to core-tag can be used to measure the distance from each other. But right now only their relative correlation is available by the mediator t_c.

3.2 Generate Similarity and Distance Matrices

In order to get similarity between tags in $\boldsymbol{T_{cand}}$, similarity matrix $\boldsymbol{M_s}$ should be generated. Rows of $\boldsymbol{M_s}$ represent the tags in $\boldsymbol{T_{cand}}$; columns store the results computed by measurements mentioned above. All the values are then normalized into domain [0, 1] and $\boldsymbol{M_s}$ is turned into distance matrix $\boldsymbol{M_d}$ by a conversion function $\boldsymbol{f}$. The conversion can be described as follows: For each element $e_{ij} \in \boldsymbol{M_d}(i,j)$, $e_{ij} = \boldsymbol{f}(\boldsymbol{sim_j}(t_t, t_t))$, t_i and t_t are two tags. t_t is core-tag t_c and t_i is the *i-th* correlated tag of t_c. Two points should be aware: (a) all the correlated tags are unordered, i don't mean the tag is the *i-th* most similar to t_c. (b) $\boldsymbol{sim_j}(t_{j,} t_t)$ represents the *j-th* similarity measurement value between t_j and t_t, the similarity are also unordered.

Since there is no comparison of values across dimensions, the conversion of values under different measurements into same domain is safe. It eases matrix transformation in later step. After conversion, entries in matrix $\boldsymbol{M_d}$ represents the location of correlated tags in new space, with t_c at the origin. There is one constraint for the convert function.

Constraint: For a similarity value s. The result of convert function is $r = \boldsymbol{f}(s)$ must satisfy that for $\forall\, s_i, s_j$, without loss of generation, set $s_i < s_j$, if $r_i = \boldsymbol{f}(s_i)$ and $r_j = \boldsymbol{f}(s_j)$, we can get $r_i > r_j$.

The definition is flexible since the constraint is loose. There are three strategies of convert function:

1) Convert the similarity values into reciprocal value. That is $r = 1/s$. This is a straightforward way to handle conversion.
2) Convert the similarity values as v_{max}-v.
3) Convert similarity values according to features of specified measurement. This method considers the differences of measurements, but its computation complexity is the highest.

In order to explain the conversion process clearly, example 3.1 is given to illustrate the step.

Example 3.1: Consider an m-measurements scenario. Assume $\boldsymbol{T_{cond}}$ contains n correlated tags of a core-tag. Thus $\boldsymbol{M_s}$ (**Fig.** 2a) is an $n \times m$ matrix. Then $\boldsymbol{M_s}$ is converted to $\boldsymbol{M_d}$ (**Fig.** 2b) by strategy 2. **Fig.** 2c shows the locations of tags in distance space with located at the origin according to the first 2 dimensions.□

Through the steps foregoing mentioned, the problem of core-tag oriented clustering can be converted to a spectral clustering problem. $\boldsymbol{M_d}$ resembles to the *term-document matrix* in document clustering, where tags are the 'document' and different similarity values are the 'term' that appear in the 'document'.

3.3 Spectral Transformation

In Section 3.2, $\boldsymbol{M_d}$ is generated. $\boldsymbol{M_d}$ can be deemed as an adjacency matrix of weighted graph $\boldsymbol{G}$ ($\boldsymbol{V}$, $\boldsymbol{E}$). In $\boldsymbol{G}$, vertices correspond to the correlated tags and edges are similarity between them.

By utilizing spectral transformation method on $\boldsymbol{G}$, tags can be well separated. We build upon the work of [12] and [6] by using multiple eigenvectors and directly get multiple partition. The following steps illustrate spectral transformation process:

1. Given matrix $\boldsymbol{M_d}$, generate the affinity matrix $\boldsymbol{A}$. $\boldsymbol{A} = \varphi(\boldsymbol{M_d})$. A□ ^{nxn}.
2. Define $\boldsymbol{D}$ to be diagonal matrix whose diagonal elements is the sum of $\boldsymbol{A}$'s corresponding row.
3. Map $\boldsymbol{A}$ into transition probabilities by constructing Laplacian matrix $\boldsymbol{L}$ by normalization.
4. Find $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots \boldsymbol{x_k}$ the k largest eigenvectors of $\boldsymbol{L}$ and form matrix $\boldsymbol{X} = [x_1, x_2 \ldots x_k]$ □ ^{nxk} by stacking eigenvectors in column.
5. Form matrix $\boldsymbol{Y}$ by renormalizing the matrix $\boldsymbol{X}$.

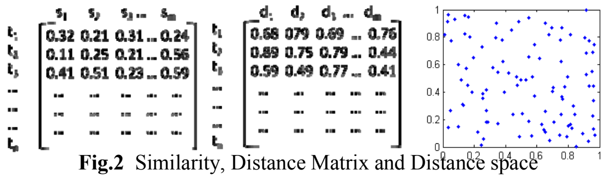
In Step 1, φ can be any function that measures similarity of two elements. In our work, we use Euclidean distance. In our work we use (1), where σ is a free scale parameter.

$$A_{ij} = e^{-\|x_i - x_j\|^2 / 2\sigma^2} \tag{1}$$

In Step 2, the matrix $\boldsymbol{D}$ is a diagonal matrix with the same dimension of $\boldsymbol{A}$. Each element located at (i, i) is the sum of $\boldsymbol{A}$'s i-th row. See formula (4).

$$D_{ii} = \sum_j A_{ij} \tag{2}$$

In Step 3, $\boldsymbol{A}$ can be normalized into transition probabilities matrix in several ways. The simplest way is let $\boldsymbol{L} = \boldsymbol{A}$. In our work we use $L = D^{-1}A$ since it performs best in our work.



$$\begin{array}{c|ccccc} & s_1 & s_2 & s_3 \ldots & & s_m \\ \hline t_1 & 0.32 & 0.21 & 0.31 & \ldots & 0.24 \\ t_2 & 0.11 & 0.25 & 0.21 & \ldots & 0.56 \\ t_3 & 0.41 & 0.51 & 0.23 & \ldots & 0.59 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ t_n & & & & & \end{array} \qquad \begin{array}{c|ccccc} & d_1 & d_2 & d_3 \ldots & & d_m \\ \hline t_1 & 0.68 & 079 & 0.69 & \ldots & 0.76 \\ t_2 & 0.89 & 0.75 & 0.79 & \ldots & 0.44 \\ t_3 & 0.59 & 0.49 & 0.77 & \ldots & 0.41 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ t_n & & & & & \end{array}$$

Fig.2 Similarity, Distance Matrix and Distance space

In Step 4 an $n \times k$ matrix $\boldsymbol{X}$ is generated by using the first k largest eigenvectors. For each row, there are k dimensions. Fiedler vector [5] is also a good choice since it is the best vector to distinguish elements. We adopt the first way because experiment shows that use more eigenvectors have better performance (See [12]).

In Step 5 converts $\boldsymbol{X}$ into $\boldsymbol{Y}$ by. The matrix can be normalized by formula (3).

$$Y_{ij} = X_{ij} / (\sum_j X_{ij}^2)^{\frac{1}{2}} \quad (3)$$

After the data has been processed by above steps, we can use the output $\boldsymbol{Y}$ as input of any clustering algorithm. Assign the original element x_i to cluster j if row i of the matrix $\boldsymbol{Y}$ is assigned to cluster j.

Algorithm 1 gives the details of CTSC algorithm that process a batch of core-tags in a run:

```
Algorithm 1: CTSC Algorithm
Input: Similarity values in different similarity measurement of a set of target core-tag $t_c$.
Output: Core-tag oriented clusters let $\boldsymbol{Results} = \varnothing$    // $\boldsymbol{Results}$ is the set of results
1.  for each core tag $t_c$ do
2.    Generate $\boldsymbol{M_s}$ and $\boldsymbol{M_d}$.
3.    Generate $\boldsymbol{A}$ by $\varphi$. Let $\boldsymbol{A} = \varphi(\boldsymbol{M_d})$.
4.    Generate $\boldsymbol{L}$ using either one of formula (6)-(9)
5.    Using first k largest eigenvector of $\boldsymbol{L}$ generate $\boldsymbol{X}$.
6.    if additive normalization is used do
7.      Renormalize $\boldsymbol{X}$ into $\boldsymbol{Y}$.
8.    $\boldsymbol{C}$ = ClusterAlgorithm($\boldsymbol{Y}$).
9.    Put $\boldsymbol{C}$ into $\boldsymbol{Results}$   // $\boldsymbol{C}$ is the result for current $t_c$.
10. Return $\boldsymbol{Results}$
```

Note that, Algorithm 1 can process clustering tasks for a batch of core-tags. The detail of Step 4 – 9 is mentioned in Section 3.3. In Step 10 we can call any clustering algorithm that is suitable for core-tag oriented clustering. Matrix $\boldsymbol{Y}$ should be added to the parameters list of these algorithms. The return value is $\boldsymbol{C}$, it contains all the clusters.

IV. EXPERIMENTS

We designed experiments to test it on the aspects of efficiency and accuracy. Since no state-of-the-art algorithm can solve the core-tag clustering task, existing algorithm had to be modified to fulfill the requirement. G-means [15] and X-means [16] are considered, they are extension version of K-means. However, the test dataset cannot meet the assumption of these two algorithms that the dataset should be Gaussian distribution. We only had to use original K-means as the counterpart of CTSC.

All experiments are implemented in Java with open source library JAMA [2] to process matrix operation. They are conducted on a PC with CPU Intel core2 Duo, 2G main memory, and Windows server 2003.

We ran our algorithms directly on real data downloaded from the recent user post of *http://del.icio.us* from Nov. 20 to Dec. 15, 2007, the span is 25 days. The statistics of the dataset can be seen in following table. All of the data are pre-processed so the processed data will be directly used. Table 1 is a brief statistic of the real data.

Total posts	1738225
Available posts	1085030
Distinct tags	234023
Distinct users	196442
Distinct web pages	749971

Table 1: Statistics of real data downloaded from

[2] http://math.nist.gov/javanumerics/jama

The available data are the number of data we used in the experiments. Some data clean tasks had been done to remove the unavailable data.

4.1 Performance test

Since no other algorithm considering problem like core-tag oriented clustering, so there is no counterpart for comparison. We test the scalability of this algorithm to see whether it can process large amount of data.

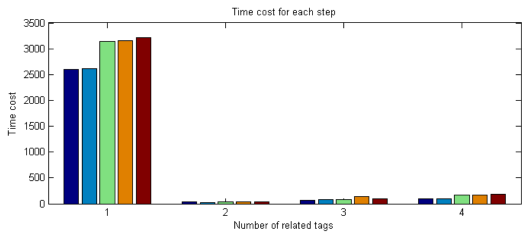


Fig. 3: Running time of different steps

Fig. 3 shows for different size of $\boldsymbol{T_{cond}}$ (60, 70, 80, 90, 100 respectively) the running time of the steps: 1) Load and generate similarity set $\boldsymbol{M_s}$. 2) Convert $\boldsymbol{M_s}$ into $\boldsymbol{M_d}$ and generate Affinity Matrix $\boldsymbol{A}$. 3) Generate Matrix $\boldsymbol{N}$. 4) Generate Matrix $\boldsymbol{X}$ and renormalize to $\boldsymbol{Y}$.

From Fig.3 we can see that the first step cost the majority of time, the I/O operation is the main reason. Considering the scalability, the time cost of CTSC didn't increase linearly with the increase of size of $\boldsymbol{T_{cond}}$. The result shows that the algorithm has good scalability. Since for different core-tag, the process step is the same, there is no need to evaluate the running time with different size of core-tag batch. The time cost is linear increase with the size of core-tag batch.

4.2 Accuracy test

For accuracy test, we need the counterpart algorithm that can meet the requirement of core-tag oriented clustering tasks mentioned in Section 1. Here are some candidate algorithms can be used:

1) Modify hierarchical clustering algorithms to do core-tag clustering algorithm. These clustering algorithms include BIRCH, CURE etc.
2) Modify K-means. Deem the cluster with the center most close to core-tag is the closest cluster.

In this experiment, there is no good candidate measurement to further group the result into different parts, if we simple divided the result into several parts with equal number of tags, the result is not guaranteed to be accuracy. Thus, it is better for us to modifying K-means as the counterpart.

We pick up 5 core tags and for each core tag we only check the first 3 closest clusters. The 5 tags are: 'css', 'Blogs', 'politics, 'Health', 'Javascript'. We consider the case sensitive situation since there may have different meaning for the same word in uppercase and lowercase. The result can be seen in table 3.

From the above table we can have a general view of the result of CTSC. The result is not very ideal because we only download the data for less than a month so the data may be skewed. If we collected the data for longer time they may be more close to the core-tag.

As there are no authoritative benchmark for research of web2.0, we asked three human to evaluate the quality of results. We mark the result of each algorithm for 10 random core-tags we picked. The evaluator should give a score from 1 - 5 according to the clustering result. The better result the higher score.
The method to evaluate the algorithm is very simple.

$$S_a = (\sum_{i=1}^{U} \sum_{j \in a} score_{aj}) / N_a \qquad (10)$$

Here, S_a means score of algorithm a, U is the number of evaluators, $score_{aj}$ is the score for each core tag get by algorithm a, N_a is the total number of scores of a.

Algorithm	Average Scores	Standard Deviation
K-means	2.71	0.51
CTSC	3.72	0.39

Table 2: The scores of algorithms

Table 2 shows the evaluate result of the two algorithms. CTSC has better quality for it does some pre-work to similarity set. Additionally, spectral method can find cliques far from each other by finding out global optimum. We use approximation algorithm to at least find out a near global optimum. Moreover, divisive normalization can bear the noisy data, which is lack for the preprocess step of modified K-means.

V. CONCLUSION AND FUTURE WORKS

We presented the concept of core-tag clustering and introduce the model of core-tag oriented space. We utilized spectral learning method to cluster the tags and showed CTSC works well to cluster the correlated tags.

The future work includes: (a) Extend CTSC to an online algorithm. (b) Find better in order to evaluate the tags in different aspect. (c) Find better normalization method for spectral clustering algorithm.

VI. REFERENCES

[1] A.Mates, *Folksonomies* – Cooperative Classification and Communication through Shared Metadata, *Computer Mediated Communication*, LIS590CMC. (2004)

[2] T.Hammond, et al. Social Bookmarking Tools:A General Review,*D-Lib Magazine* (2005)

[3] A. Hotho, R. Jaschke, et al. Information Retrieval in Folksonomies: Search and Ranking. *The Semantic Web: Research and Applications*, 2006

[4] W.E.Donath and A.J.Hoffman. Algorithms for partitioning of graphs and computer logic based on eigenvectors of connection matrices. IBM Technical Disclosure Bulletin, 15:938-944, 1972.

[5] M.Fiedler. A property of eigenvectors of nonnegative symmetric matrices and its applications to graph theory. Czechoslovak Mathematical, 1975.

[6] Marina Melia, Jianbo Shi. A Random Walks View of Spectral Segmentation. AIS-TATS, 2001

[7] Jianbo Shi, Marina Melia. Normalized cuts and image segmentation, PAMI, 2000.

[8] Andrew Y.Ng, Michael I.Jordan, and Yair Weiss. On Spectral clustering: Analysis and an algorithm. NIPS2002, 2002.

[9] Sepandar D.Kamvar, Dan Klein, Christopher D.Manning. Spectral Learning. IJCAI, 2003.

[10] Daniel A.Spielman and Shang-Hua Teng. Spectral Partitioning Works: Planar graphs and finite element meshes. FOCS, 1996.

[11] Y.Weiss. Segmentation using eigenvectors: A unifying view. ICCV, 1999.

[12] C.Alpert, A.Kahn and S. Yao. Spectral partitioning: The more eigenvectors, the better. Discrete Applied Math, 90:3-26, 1999.

Core tag	1st cluster	2 nd cluster	3rd cluster
css	Web, javascript,showcase,tabs,standards	Design, reference,ajax, development,dhtml,	tools,xhtml,grid, dropdown, webdev, webdevelopment,
Blog	blog, web2.0,blogging, edublogs,googleandbeyond,	video,bloglines,tech	culture,technology, politics
politics	Corruption,Environment,Economics,history,democracy,Magazine	Law,war,president,progressive, libertarian	blog, reference,government, election, Iraq, bush, news
Health	medical, medicine, Weight, bodybuilding,calories, diseases	obesity,eca,weightloss,remedies	doctor,hospital,depression
Javascript	xmlhttprequest,tools,compressor, modal,scriptaculous,drag,tutorial,PHP	Design,web2.0,extjs,css, google,securitycouch	tooltips,browser,reference,script.aculo.us,Flash,dojo,slider,compression

Table 3: First 3 clusters of 5 sample core-tag