

Analyzing Tagging Behavior in Clustering Similar Web Resources through Interactive Visual Demonstration

Marjan Farsi

PhD scholar, Department of computer science, University of Delhi, New Delhi, India

ABSTRACT: *There are millions of web pages which are annotated by the means of freely chosen words called tags and saved in social tagging systems, daily. These Tagging-based systems like www.delicious.com provide internet users with the facilities to store their web resources in the web space in order to be accessible and retrievable from everywhere around the world. These sites by organizing data and providing the search facility based on tripartite elements (user, tag, bookmark url), give good information services to the internet users. In recent years, by developing and spreading the usage of social bookmarking sites, many web mining researchers and scientists have become motivated to study the data acquired from these sites to explore new information from these sites. Consequently the techniques for web crawling and data extraction, classifying and clustering algorithms and data visualization methods and tools have been applied for this aim. Usually these acquired and clustered data are analyzed in order to getting the hidden statistical or behavioral facts and concepts embedded in the relation between tripartite elements.*

In this paper, one aspect of these behavioral facts and concepts, the effect of tagging behavior to find web pages similar in content according to the common tags of the extracted urls, will be analyzed and discussed. All these required data comes from one of these social bookmarking sites, www.delicious.com.

This similarity will be explored through executing an implemented Java application in which the similar web pages will be clustered in similar groups by applying similarity measurement algorithms and k-mean clustering technique. This investigation has been done quantitatively and qualitatively. It means that the statistical facts about tagging behavior in finding similar web pages, which are generated by the produced application, will be reported in Excel sheet format, also the processed data will be visually represented in graph structure by applying 'Prefuse' visualizing tool. The relationships between visual objects in each graph will be discussed and analyzed from the tagging behavior point of view.

Key Words: *data visualization tool, k-mean Clustering, similarity measurement algorithms, social bookmarking sites, Tagging Behavior, Web mining,*

1. INTRODUCTION

By developing web 2.0, the need of keeping internet resources for further refers has emerged. Bookmarking and saving urls or web pages on the storage of personal computer carry some problems or have drawbacks. For instance, the saved data on PC web browser, could be diminished or lost by deleting history or by updating and changing operating system. To avoid these limitations, social bookmarking sites like delicious have been introduced to provide the facility for internet users allowing them to annotate web resources using freely chosen key words known as tags.

Storing and annotating these resources on existing web sites, not only keeps data and references safe and always accessible, but also categorized resources become easily findable when needed. However, tagging-based systems provide a rich storage of meta data organization, they do not give chance to researchers of different topics or subjects to extract and cluster the web pages similar in content and shared by other users were interested in the same area of study, based on their tag vectors.

In this paper the web pages similar in content of a bookmarking site, called delicious, are going to be clustered, visualized and analyzed based on their urls' tag vectors.

In the following some standard terms and concepts related to these fields of study will be introduced.

1.1 Basic Concepts and Terms

1.1.1 Tag

Bookmark sites like delicious, Flickr, Furl, Rojo, Connotea, Technorati, and Amazon allow users to bookmark and annotate the resources they are interested in using personal keywords called tag.[1]

1.1.2 Folksonomy or Tagging system or Social bookmarking sites

The term "Folksonomy" was first declared by *Thomas Vander Wal* in *AlfIA* mailing list. It is a neologism word which consists of word "taxonomy" and "folk". Folksonomy is considered as a space of tagged resources. In other words, different users by sharing their resources with specified keywords as tags, generating an aggregated tag space so-called folksonomy.

1.1.3 Delicious

Delicious is one of the first social sites which was constructed in 2003 by *Joshua Schachter*, and then because of its popularity was bought by the Yahoo company in 2005 [2].

This site allows users to organize, save and retrieve their bookmarks by tag. The ability of browsing, filtering and searching through tags are the facilities that this site provides for users. Also it is possible to look for a url to get its related tags and number of users by entering the url string in the search box or finding a specified user by his/her user name. Delicious is accessible from this address: <http://www.delicious.com>.

1.2 Background and related work

At first we are going to have an introduction to the basic algorithms, tools, methods and technologies based on which the different approaches and products have been emerged. Then we will have an overview of one related work, product or analysis tool, which is developed under these basic tools and technologies, to solve a problem in social bookmarking sites or give an analytical inference about the relationships between tags, urls or users.

1.2.1 Basic algorithms

1.2.1.1 Similarity Measurement algorithms

There are several similarity measurement algorithms such as Mutual Information, Simrank, Pythagorean Theorem, Pessimistic Similarity, Cosine similarity, adjusted Cosine similarity, Pearson coefficient and TF/IDF.[3]

In this paper, three of these algorithms are applied to measure the similarity between bookmarks.

In the following these three algorithms are explained.

“Let the tag set be T, the number of pages that tag A annotates be La, the number of pages tag B annotates be Lb, the number of common pages both tag A and tag B annotate be Lab and the number of web pages be N. “[3]

1.2.1.1.1 Definition: Similarity between Tags

Let A and B be two Tags and Lab be the number of shared web pages of A and B. Let the threshold be d, d>= 1. If Lab > d then A and B are called similar tags. Similarity is used to describe the degree how they are similar to each other. Similarity between A and B is denoted as s(A,B).[3]

1.2.1.1.2 SimRank

It is the simplest similarity algorithm, in which “two objects are similar if they are related to similar objects. ” This algorithm is based on simple and intuitive graph-theoretic model[4].

- Similarity between A & B denoted by:

$$\begin{cases} s(A,A) = 1/La \\ s(A,B) = C * Lab / (La * Lb) \end{cases} \quad \text{if } A \neq B$$

Equation 1: SimRank [3]

- If A = B, s(A,B) = 1, → s(A,A) = s(B,B) = 1 (An object is maximally similar to itself)
- C is called as “confidence level” or “decay factor” and shows less confidence on the similarity between A and B than the confidence between A and itself. It is a constant between 0 & 1
- If La or Lb is 0, s(A,B) = 0
- This method is symmetric: s(A,B) = s(B,A) [4].

1.2.1.1.3 Pythagorean

“*Pythagorean Theorem* is a classic theorem in Euclidean geometry. It describes the relationship among the three sides of a right triangle, as “The Square on the hypotenuse is equal to the sum of the squares on the other two sides”. Let Lab be the length of one leg, and (La – Lab) + (Lb – Lab) be the length of the other legs respectively; then the length of the hypotenuse is $\sqrt{Lab^2 + (La + Lb - 2 * Lab)^2}$. The formula to measure the similarity between two tags is shown in Equation (2). According to this, s(A,B) increases along the increasing of Lab and decreases along the increasing of La and/or Lb.”

$$s(A, B) = \frac{Lab}{\sqrt{Lab^2 + (La + Lb - 2 * Lab)^2}} = \frac{1}{\sqrt{1 + \left(\frac{La}{Lb} + \frac{Lb}{La} - 2\right)^2}}$$

Equation 2: Pythagorean [3]

1.2.1.1.4 Cosine

“In this section, the set of pages that tag i annotates is denoted as Ii, and the number of users who have annotated page j by tag i is denote as Ci,j.

In information retrieval, the similarity between two documents is often measured by treating each document as a vector of word frequencies and computing the cosine of the angle formed by the two frequency vectors. This formalism can also be adopted to calculate the similarity between tags, where tags take the role of documents, pages take the role of words, and user count take the role of word frequencies. Then the similarity is defined as”: (in Equation (3), “*” is a vector dot product)

$$s(A, B) = \frac{\sum_{j \in I_A} C_{A,j} * C_{B,j}}{\sqrt{\sum_{k \in I_A} C_{A,k}^2} \sqrt{\sum_{k \in I_B} C_{B,k}^2}}$$

Equation 3: Cosine [3]

1.2.1.2 K-Mean Clustering Algorithm

K-Mean algorithm is an iterative unsupervised learning algorithm for classifying and grouping objects based on one or more attributes into k (is a positive integer number) number of clusters. ”The grouping is done by minimizing the sum of squares of distances between data and the corresponding

cluster centroid.”[5]

1.2.1.2.1 Determining minimum distance

Assume that we have three objects A, B, E, with two attributes X and Y (x and y can be any measurable attribute, in the present project, each object (url) has just one attribute and it is the rate of similarity between that url and each centroid url; more details about the implementation of K-Mean algorithm in this project, is discussed in next part).

$$A(X_a, Y_a) = C0$$

$$B(X_b, Y_b) = C1$$

$$E(X_e, Y_e)$$

We get A and B as centroids, so the distance between object E with each centroid A and B is computed as follow:

$$\text{Distance}(E \text{ and } A) = \text{Sqr}((X_a - X_e)^2 + (Y_a - Y_e)^2)$$

$$\text{Distance}(E \text{ and } B) = \text{Sqr}((X_b - X_e)^2 + (Y_b - Y_e)^2)$$

If Distance(E and A) < Distance(E and B) Then

E will put in cluster with Centroid C0

Else

E will put in cluster with Centroid C1

1.2.1.2.2 Centroid determining method

In each iteration except in first iteration centroids will be determined as follows:

Assume we have three objects A, B and C in a cluster 0 in nth iteration, determining the centroid for next iteration is as follow:

$$\left. \begin{matrix} A(X_a, Y_a) \\ B(X_b, Y_b) \\ C(X_c, Y_c) \end{matrix} \right\} \rightarrow C0 = ((X_a + X_b + X_c) / 3, (Y_a + Y_b + Y_c) / 3)$$

In first iteration centroids are selected randomly. K-Mean algorithm, is an iterative process which through completing the three steps below, put all objects in a best-fit cluster.

1. Determine the centroid.
2. Determine the distance of each object to the centroids.
3. Group the object based on minimum distance (find the closest centroid)

The process cycle of K-Mean algorithm is illustrated in Fig.1:

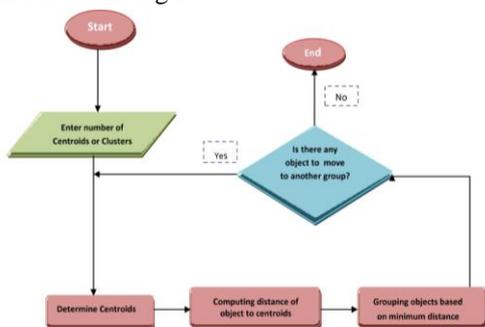


Fig. 1: K-Mean algorithm

The K-Mean clustering process will be repeated until no more data is moving to another cluster [5].

1.2.1.2.3 K-Mean algorithm as used in this paper

I have applied K-Mean clustering algorithm by the following characteristics in my work.

In the following, these characteristics and changes will be illustrated.

1. This algorithm in my work is used for clustering similar bookmarks or resources based on the number of common tags between each pair of urls, so the rate of closeness or similarity is the factor of placing objects in one cluster (Similarity = 1/Distance, it is normalized number).
2. The only attribute of any object (url) is its similarity rate to each centroid which is a constant value instead of each url.
3. As this attribute will not change, the k-mean process in my work will be accomplished in the first iteration, because there is just one constant attribute value (rate of similarity to each centroid) instead of each url, which will not change. So in the first iteration all urls will be put in the best cluster which have maximum similarity to its centroid.

1.2.2 Standard and tools

1.2.2.1 Prefuse

“Prefuse is a set of software tools for creating rich interactive data visualizations. The original prefuse toolkit provides a visualization framework for the Java programming language. Prefuse supports a rich set of features for data modeling, visualization, and interaction.” [6].

1.2.2.2 RDF

RDF stands for “Resource Description Framework” which is a standard of W3C for describing resources on the web. From the resources we mean anything that can be identified by Uniform Resource Identifier (URI). [7]

1.2.2.3 Jena

Brian McBride introduces Jena in his ‘An introduction to RDF and Jena RDF API’ paper as follow:

“Jena is a Java API for semantic web applications. This package contains interfaces for representing models, resources, properties, literals, statements and all the other key concepts of RDF, and a ModelFactory for creating models.”[8]

“Jena is an open source semantic web framework based on W3C recommendation for RDF and OWL.”[9]

It is used for reading and writing and processing OWL, RDF, RDFS, OWL, SPARQL

data models in an integrated rule-based inference mechanism. [9]

The present project use this tool for reading RDF statements in order to extract data from delicious site.

1.2.3 Review of Application SOM

According to [10], in the SOM tool, Marco and Edwin proposed an algorithm in which, each tag with frequency more than the threshold, become the core of a cluster, and bookmarks tagged with this core tag are connected to it. By changing the threshold, the number of clusters changes. Finally in an iterative process the optimal number of clusters with reasonable size and distribution are acquired.

SOM then provides the facility to cluster any new bookmark into one of these clusters based on the number of shared tags. The new bookmark will be put in the cluster that has the most common tags with cluster's most frequent tags.

Tagging behavior in this paper could be analyzed from two aspects: tag clustering and classification of new tagged bookmark.

Tag behavior here is explained by changing the number of cluster, three kind of clusters could be emerged by changing the number of clusters:

1. Some clusters appear (blue)
2. Some clusters disappear (Purple) (over-populated clusters disappeared)
3. Some clusters persisted (Green)

By decreasing the threshold, the number of clusters increased. So tags with lower frequency also get a chance to be the core of a cluster.

By increasing the threshold some clusters disappeared; these clusters are those who consist of unpopular tags with low frequency. Clusters that do not disappear by increasing the threshold, consist of tags with high frequency and assumed as popular tags.

In classification of new tagged bookmark, bookmark will fall into the cluster whose tags are the same as the most frequent tags of cluster.

Based on the experience I have acquired during my project, most frequent tags are usually 'general' tags like 'design', 'google' or 'web' which have general meaning that could be included as concepts or topics, resulting that many pages with different content may be tagged with these kind of words.

Consequently, if these general meaning words with most frequency are supposed to become the core of a cluster, like what happens in SOM system, and clustering similar web pages formed based on these general tags, it seems that it could not be a proper way for clustering, as the connected web pages are not necessarily similar in content; because many pages with different content may have been already tagged with these kind of words. So it seems that if SOM changes its criterion in selecting cluster core, better results could be

achieved.

1.2.3.1 Comparison and discussion:

I have had almost the same experiment in my work in clustering web pages similar in content. In my work for clustering similar bookmarks, at first the urls with maximum connection to other urls became selected as cluster centroid, but the experiences showed that this approach for selecting centroid could not lead to good clustering because these kind of urls are the ones that their tag vectors have two characteristics. Firstly they have high tag vector size. Secondly most of their tags are general words and web pages connected to these centroids are not necessarily similar in content. So the approach I applied then, has been centroid random selecting.

2. A Framework For Extracting Semantic Correlations And Effective Clustering

2.1 Overview

In line with organizing these information from Tagging system sites like delicious and represent them in graphical form to get meaningful analysis and interpretation, this project is emerged.

The project objective is to introduce a software agent that is implemented in Java:

1. To extract, classify, organize and manage the web resources belonging to clients kept on tagging sites such as delicious.
2. To develop a visualization tool based on a software package like Prefuse to provide an interactive visualization of bookmarks correlation in a graph structure for giving better understanding of visual objects (tags and urls) correlations.
3. To analyze the semantic relationships among tags or analyzing tagging behavior in relation to clustering similar web pages in order to achieve some quantitative and qualitative inferences from the visual objects correlations.

For this aim, clustering methods and algorithms to find semantic relationships between the tags and urls are applied.

The main work here is to generate a system which automatically extracts data then visualizes and analyses the bookmarks correlations. The first stage involves downloading web content from a social bookmarking system (delicious web site) and processing them to get similar web pages. Then to create an interactive visualization system which uses intuitive visualization techniques and allows the users to quickly get an insight and understanding on the data correlations and grouping, then the last stage is to explore the tagging behaviors between visual objects. For this, it should be understood that what kind of information can be extracted explicitly from the visualization regarding the tagging behavior.

In the following the summary of implementation

steps of this project are explained.

2.2 From Tag an url Extraction and Clustering to Interactive Visualization

Step 1: Extracting popular tags from the delicious site:

The “PopTagExtraction” class in this project implements this task.

The goal in this stage is to access to amount of initial tags in order to use them for extending a domain of dataset consist of the collection of <Bookmark, tag> in next stage.

Html parser library, Jericho, is used to extract the most popular tags from the Tag cloud at <http://delicious.com/tag/>, based on their popularity.

Besides the tag name, there is an attribute class determined by “size” which indicates the popularity of the tag in relation to the other tags of the tag cloud. For instance tag with size 5 is more popular than tag with size 3. Here is an example of its html tag:

```
<a href="/tag/design;_ylt=A0wNBqAG5sJL3xEA20RjRh54;_ylv=3" class="size5">design</a>
```

Tags with size below 1 will not be contributed in process. Finally tags are saved in “txtTagItems.txt”.

Step 2: Using the most popular tags in URLs to obtain more tags.

This functionality is implemented in “RDF_TagUrlExtractor” class.

For getting more tags and urls to extend the dataset domain, program reads in the RDF file format at the following web address:

```
http://feeds.delicious.com/rss/tag/{Tagname}
```

{Tagname} is come from the extracted tags from previous step. In an iterative process each tag from “txtTagItems.txt” should be read and added at the end of above URL in order to get more related tags and urls embedded in each feed/rss page. For example: <http://feeds.delicious.com/rss/tag/design>.

The feed formats are in xml, so use an open source RSS Feed Reader java libraries (here is “Jena”) to read the web feed which contain the bookmarks and related tags.

In RDF file structure, there is a “Bag” tag in each iteration which is consists of BookMark URI and its related tags. By Jena tool, system can read the particular statement of RDF file and getting required data. Data is saved in text file, “txtUrlTags.txt”.

Step 3: Generate xml file of extracted data

The “XmlGenerator_TagUrl” class is implemented to forming an xml structure from dataset (dataset here is consisted of extracted tags and ulrs from previous steps which are saved in ”txtUrlTags.txt” text file). Data obtained from previous step and saved in text file, is converted to xml file format with<ITEM>, <URL> and <tag> tags and is saved in “txtUrlTags.xml” file for further usage.

Step 4: Clustering

Clustering means putting the similar data in the same categories.

The basic required activity for clustering is to generate a matrix in which the similar objects and their similarity values are declared and determined. This matrix is known as a **Similarity Matrix**.

This project aims to analyze and visualize correlation between urls. Similarity matrices here are supposed to convey the web pages similarities. As it is introduced in previous part, there are several methods for classifying data into similar groups.

The definition of two kinds of similarity matrixes as the main output of step 4 of the project are explained here:

Definition 1: Tag -Url Similarity Matrix

Let $M_{UT} = (U, T)$ be a graph of visual objects , where U is the set of urls and T, the set of related tags, with $|U| = M$ and $|T| = N$ and N and M are integer. $\{\forall u_m \in U, \exists T_i = \{t_1, t_2, \dots, t_n\}, n \in N | T_i = \text{tag vector for } u_m\}$

	U₁	U₂	...	U_m
T _i {	t ₁	t ₁		t ₁
	t ₂	t ₂		t ₂
	.	.		.
	.	.		.
	t _n	t _n		t _n

Fig. 2: Tag-Url similarity matrix

Definition 2: Url – Url Similarity matrix

Let $M_{UW} = (U, w)$ be a Matrix of visual objects , where U is the set of Url pairs and w is the set of weight Edges of a pair of visual objects u_i ,and u_j and $|w| = 2$ and N is an integer, with $|U| = N$. $\{\forall U_n \in U, \exists u_i, \text{and } u_j \in \text{URL collection and } \exists w_i \in \{10,200\}, n \in N, \text{ if } u_i, u_j \text{ are Similar}\}$

	U1	U2	...	U_n
u₁₀		u₉₈		u_i
u₇₃		u₄₅		u_j
10		200		10

Fig. 3: Url-Url similarity matrix

‘URL collection’ consists of all urls in the reference dataset.

The edge weight, W_i , is representative of the level of similarity between each url pair. The url similarity value is a normalized number between 0 and 1, but for displaying this rate of correlation, the visualize tool need an integer number, which determines the thickness of the edge between each pair of tags, which in its turn shows how much two

urls are similar [11].

Step 5: Visualizing clustered data

By generating the similarity matrix the clustered data is available, it should be just displayed in proper graphical format by visualize tools like ‘Prefuse’.

For generating a graphical display, it is required to create an xml file consists of nodes and edges of the graph.

2.3 Generating Similarity Matrix for extracting similar web pages

As explained in step 4, the main part of project is dedicated to generating similarity matrix. In the following the functionality of main function which implements this matrix is introduced.

The clustering type, similarity accuracy, the type of selecting centroid (Random or Urls with most connection to other urls), type of similarity algorithm and the size of dataset as a percentage of total dataset are given as input parameters to related class for executing the function which cluster similar web pages.

This is the main function and these steps are followed in this function:

Step 1: Get reference dataset. At first this function get the reference dataset for further usage in this function.

Step 2: Get pairs of urls collection with similarity rate greater than specified threshold (threshold determined experimentally) by applying similarity measurement algorithms.

Step 3: Applying clustering algorithm to cluster similar urls, also grouping similar urls without applying any clustering algorithm.

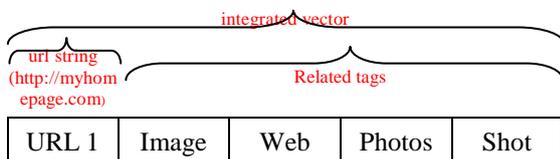
There are two approaches to put the similar urls in the same group:

- Simple grouping
- K-Mean clustering

Simrank, Cosine and Pythagorean similarity measurement functions which implement similarity measurement algorithms, are applied in this work to compute the rate of matching between each pair of bookmarks.

Each one of these functions receives two vectors consisting of bookmark url and its related tags. Co-occurrence between two tag collections (the number of similar tags that occur in both vectors) is computed as subscription between those two urls, and put in the similarity formula to get the match rate of these two urls.

For example assume there are two elements of two vectors BM₁ and BM₂ integrated in one vector :



URL 2	Tools	Search	Image	Web
-------	-------	--------	-------	-----

Fig. 4: sample of two elements of two vectors BM₁ and BM₂ integrated in one vector

According to the Fig. 4, the subscription between these two urls is two, as they have two common tags, “Image” and “web”,

$$L(url_1 \cap url_2) = 2$$

in other word

Put this in Simrank formula:

$$match\ rate = 2 / L(url_1) * L(url_2)$$

(Let L(url) be the length of tag collection of each url) the match rate between these two urls achieves.

The match rate variable is a normalized number between 0 and 1 in double format, which represents the rate of similarity between those two urls.

After measuring similarity, urls can be clustered into groups, based on their similarity, in order not only to give more useful and precise information to users about the similar web pages, but also provide more clear and understandable view of relation between data.

Considering this fact that each url may be in connection with, or similar to, one or more urls with different rates of similarity, here, by giving weight to these similarity values, the rate of similarity between content of each pair of similar web pages is determined. The effectiveness of weight difference will be illustrated in the visualization demo.

2.4 The Clustering Process

Based on three important steps in K-Mean clustering algorithm, as it was explained before, the “K-Mean_Clustering” class covers the following steps:

1. Determining proper centroid as the core of each cluster.
2. Computing the distance of each url to each centroid url (here instead of distance, similarity will be considered, similarity is 1/Distance).
3. Put each url into the group which has lowest distance to its centroid or it is most similar to its centroid.

Note: Here k-mean clustering is performed in one iteration because the rate of similarity to each centroid as an attribute, is a constant and unique number (each object distance from each centroid will not be changed), which is sufficient to determining which url should be located in which group. So changing the centroid in next iteration will not be helpful, and one iteration is sufficient.

The k-mean clustering class performs two tasks:

Task 1: This function returns all related centroids for each url. In other words among the determined centroids the ones which have similarity rate more than or equal to the specified threshold, with token

url, put together with it, in one collection, as vector. After this task has been performed, there will be a vector of vectors in which each array i consists of each url and its possible centroids.

i = 0			i = 1			i = 2		
Token Url 1	Token Url 1	Token Url 1	Token Url 2	Token Url 2	Token Url 2	Token Url 3	Token Url 3	Token Url 3
Cent 2	Cent 3	Cent 5	Cent 1	Cent 5	Cent 1	Cent 1	Cent 2	Cent 3
MR 2	MR 3	MR 5	MR 1	MR 5	MR 1	MR 2	MR 3	

Fig. 5: vector consists of possible centroids for each url

MR here is stand for Match Rate, which is a number between 0 and 1 and represents the rate of similarity between each token url and related centroid.

The output of this task is a vector, which it's structure is shown in Fig. 5.

Task 2 (Bubble sorting): As it is understood from the previous step, each token url may have a percentage of similarity rate or correlation with one or more centroids.

In the K-Mean clustering algorithm the aim is to put each token url into the group with the most similar centroid. So in this task, each token url's related centroids should be taken and then sorted in descending order. The output unsorted vectors from this task are illustrated in Fig. 6:

i = 0				i = 1			i = 2			
Token Url 1	MR2	MR3	MR5	Token Url 2	MR1	MR5	Token Url 3	MR1	MR2	MR3
Url 1				Url 2			Url 3			

Fig. 6: sample elements of unsorted vector consists of centroids similarity rate to a url

Assuming that the first token url's similarity to the fifth centroid is greater than its similarity to the second and third centroids, and its similarity to second centroid is greater than its similarity to third centroid, the bubble sort algorithm will return the following setting for token url1's related centroid, as illustrated in Fig. 7.

MR5>MR2>MR3.

i = 0				i = 1			i = 2			
Token Url 1	MR5	MR2	MR3	Token Url 2	MR1	MR5	Token url 3	MR2	MR1	MR3

Fig. 7: sample elements of sorted vector consists of centroids similarity rate to a url

After sorting the related centroids in descending order, the first centroid Match Rate would be assumed as the closest centroid in similarity to the specified token url.

In this step all token urls (Turl) put into their appropriate cluster, which has closest similarity to its centroid or cluster core.

The output vector from this step is illustrated in

Fig. 8:

Cent1	Cent1	...	Cent1	Cent2	Cent2	...	Cent2	Cent3	Cent3	...	Cent3
Turl5	Turl135	...	Turl i	Turl34	Turl1002	...	Turl j	Turl3	Turl904	...	Turl k
MR5	MR135	...	MR i	MR34	MR1002	...	MR j	MR3	MR904	...	MR k

Fig. 8: vector consists of closest centroid to each url with their rate of similarity value (Turl = Token Url)

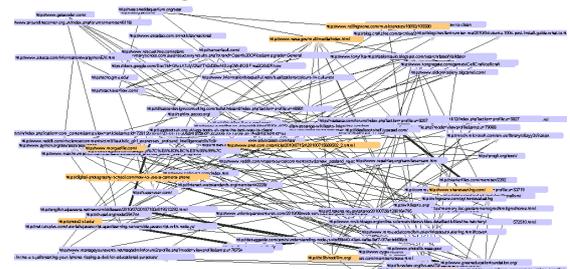
This is the final output vector from the K_Mean_Clustering class, and this vector will be given to the graph generator class for generating the xml file, which is given as a feed for displaying the graph.

2.5 Visualization

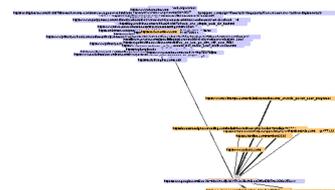
The visualization tool, here prefuse, needs an xml file as a feed to provide the information to generate the graph and visualization which can depict the outcome of the clustering process as described in the previous section. Thus the program also needs to generate an xml structure of visual objects (urls) and their relations.

The function generates the xml file for displaying the clustered urls. The produced xml file is comprised of weight property values for the edges.

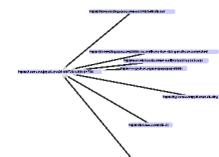
Fig. 9(a) illustrates how the rate of similarity between two urls could be recognized by the difference in thickness of the connection edge; Fig 9(b) shows the clustered urls when clustering algorithm is applied. Fig. 9(c) and (d) demonstrate a close view of weighted graph and clustered graph (Fig. 9(c) and (d) are not the closer view of Fig. 9(a) and (b))

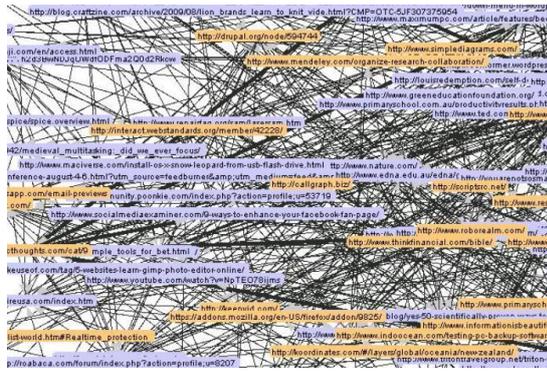


(a)

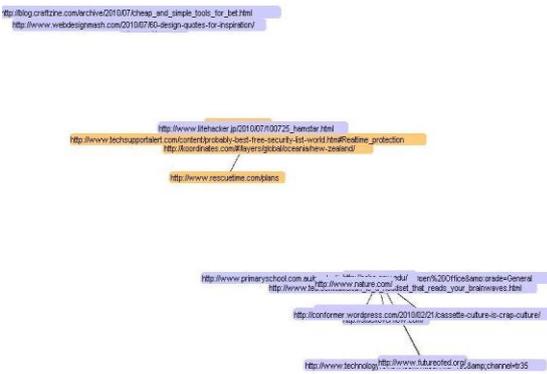


(b)





(c)



(d)

Fig. (9): Output weighted graph from 'Prefuse' tool, similar Urls (nodes with orange color are similar urls) (a) Without applying clustering algorithm, (b) Applying clustering algorithm, (c) Close view of weighted graph, (d) Close view of clustered urls. (Output from running the program of the present project)

In the following, the modality of user/system interaction for setting the independent parameters will be described.

Fig. 10: form for setting parameter for displaying similar urls

In the following each part which is illustrated in the forms is explained.

1. In the first step, the user can create a new random dataset by providing the percentage of the total amount of desired data, or the user can just select the percentage of data from present dataset he/she would like to have taken under test, and do not need to push the create button.
2. Set clustering parameters, including clustering algorithm (K-Mean or none), number of clusters user wants to have, the method of selecting centroid (random or based on maximum correlation). Check the 'create random index' checkbox if the user wants to generate a new random centroid instead of the present centroids (three recent parameters should be set if user select the k-mean clustering algorithm).
3. Set the rate of similarity accuracy as threshold for filtering data.
4. Set similarity measurement algorithm type.

3. Tagging behavior analysis

Similar web pages' clustering, is going to be analyzed from the perspective of tagging behavior. The output will be analyzed from two aspects: Quantitative and qualitative.

3.1 Bookmark similarity

Most users are interested in getting similar web pages to the ones they are interested in.

In this study, this facility is provided for the users to have a view of clustered similar web pages.

In the following, the effectiveness of the clustering algorithm and similarity accuracy are discussed.

The effectiveness of the rate of similarity accuracy in grouping similar data by applying clustering k-mean algorithm and not applying any clustering algorithm is going to be discussed here through a sample or test:

The test condition or initial condition is as follows:

Initial dataset:

As the entire dataset is too large, and analyzing the whole data set not only may take too much, but also cause the output graph become too messy and unreadable, so in this test just a random sample of the data set is analyzed. The examination is done on 30% of the reference dataset, acquired from the delicious site. Around 170 urls and their corresponding tags are randomly utilized in this test, which is a favorable amount for this analysis.

Initial parameters' values:

- Data amount percentage: 30%
- Number of cluster: 3
- Centroid selecting method: Random
- Similarity Accuracy: 0.3 and 0.09
- Total Bookmark amount: 564

- Bookmark amount under test: 169
- Clustering algorithm: K-Mean
- Similarity measurement algorithm: Cosine

3.1.1 Test Results and tagging behavior analysis

Steps for getting results:

In the following the steps performed for evaluating the results are described:

1. Running the program by applying K-Mean clustering algorithm and save each centroid and its related Urls in a text file as “K-Mean_ClusteringData”, instead of specified similarity accuracy.
2. Running the program without applying any clustering algorithm and save each pair of related Urls in another text file as “Simple_Grouping Data.txt” instead of specified similarity accuracy.
3. Reviewing the content of web pages in each cluster and compare it to centroid web page content. (Similar review was done for the result acquired when no clustering algorithm was applied).
4. Reviewing the urls’ tag vectors and comparing them to the centroid tag vector. (The same review was done for the result acquired when no clustering algorithm was applied).
5. Making decision about which urls are really similar in content by comparing the content of urls and count the real similar urls.
6. Computing the proportion of similarity in each cluster.

The above steps can be summarized in these statements:

The centroid urls are considered and the related urls to these centroid urls are studied and investigated within two cases, by applying clustering algorithm and not applying clustering algorithm, to compute the proportion of similarity.

Under the above condition the test was performed and the following results obtained. Before discussing the quality of clustering under the above condition, studying and comparing the following statistical data about this sample gives good general insights about the clustering method, the effectiveness of similarity accuracy and etc.

In this sample, the K-Mean algorithm was applied with two values of similarity accuracy, 0.3 which determines rather high degree of similarity, and 0.09 which shows rather low degree of similarity.

Once again without using K-Mean algorithm and just by changing similarity accuracy from 0.3 to 0.09, the program was run.

Note: These two values, 0.09 and 0.3, are sample thresholds and they were achieved experimentally,

and they work fine for only this quantitative analysis under the above condition (by working fine, means that they provide acceptable amount of connected urls for studying tagging behavior in both methods, by applying clustering algorithm and not applying any clustering algorithm). If the data set or the amount of data under examination changes, it may be better these two values become changed too; for example if the percentage amount of data increases from 30% to 60%, the threshold value of 0.4 or greater may lead to good result too. As these two threshold values (0.3 and 0.09) determine the rate of similarity accuracy, through trial and error method, it was investigated that in this data set and under the above condition the value much less than 0.09, causes that the web page urls with very few common tags to get involved as similar web pages, while they are not actually similar in content. Value much greater than 0.3 causes some similar web pages to become ignored because, in this case, the number of common tags should be more than what is required to satisfy the condition of 0.3 similarity accuracy.

The quantitative results are shown in Table 1 and 2 and Fig 11.

Table 1: Comparing clustering quality when similarity accuracy is 0.3

Similarity Accuracy = 0.3				
	Centroids	Number of Correct clustered urls	Number of Total urls in cluster	Percentage of clustering quality
K-Mean	C0	3	5	66%
	C1	5	5	
	C2	2	5	
Non K-Mean	C0	3	5	66%
	C1	4	4	
	C2	2	5	

Table 2: Comparing clustering quality when similarity accuracy is 0.09

Similarity Accuracy = 0.09				
	Centroids	Number of Correct clustered urls	Number of Total urls in cluster	Percentage of clustering quality
K-Mean	C0	6	17	50%
	C1	5	5	
	C2	5	17	
Non K-Mean	C0	3	16	46%
	C1	5	5	
	C2	5	17	



Fig. 11: comparing similar urls clustering quality in similarity accuracy of 0.3 and 0.09

The above values are obtained from reviewing the url web pages and their related tag collections in each cluster.

NB: The difference between the results obtained from applying k-mean clustering algorithm and not applying it at all, resides in the number of urls which are common between centroids; for example a url may be similar with two or more centroid urls. In the k-mean clustering algorithm, this url is placed in one cluster which have the most rate of similarity with it, while if k-mean algorithm is not applied, this url will be in connection with all centroid urls that it has similarity with them and will be counted and computed instead of each one.

As it is understood from the above charts and statistics, in the test case performed, there is no noticeable difference in clustering quality when k-mean clustering was applied (50%) or when it was not applied (46%), because there are few common urls between centroids.

In Table 1 and 2, the number of urls which are clustered correctly and the total number of urls in each group is defined.

Note: Correctly clustered urls means the urls whose contents are similar to the content of the centroid url.

The examination shows that under the condition in which the similarity accuracy is rather high (around 0.3), most of the web pages in relation to one centroid url point to one special concept. For example all web pages about ‘photo’ and ‘photography’ have just two out of five urls which are different from the others. By checking the tag vector of this centroid url, we understand that the ‘photography’ and ‘photo’ tags are common in all of these urls’ tag vectors. Consequently, by considering tags and urls’ contents, a specific topic could be assigned to a cluster (like the ‘photography’ topic in this example). However, by paying more attention, it is seen that each web page may include special kinds of photos. For instance, ‘Nasa’ web page and ‘Morguefile’ are placed in one cluster and both of them are describing a photos’ topic, while Nasa consists of space photos and Morguefile is a download site of any kind of

photos. Consequently, however, general tags like ‘photo’ cause many similar urls to be put together in one cluster, and all of these web pages also provide users with photos; but as such tags are too general, the web pages in one cluster are not necessarily consisting of special groups of photos.

The test was repeated using different datasets with bigger size and with more urls under test; but as quantitative analysis and investigating each url’s content and its tag vector, like what is described in the previous test, is a complicated process, it sufficed to do qualitative and intuitive analysis in the following section. Regarding all tests which have been conducted, in this analysis, the effectiveness of three main factors in the quality of similar web pages’ clustering will be discussed.

3.1.2 Study on the effectiveness of three factors in clustering quality (Presence of general tags in tag vector, similarity accuracy rate and centroid tag vector size)

By repeating the test with similarity accuracy of 0.09 (the reason for selecting this value was explained above), or by decreasing the rate of accuracy, we have clusters in which it seems that most of urls represent the same content.

With a closer attention to originating clusters, it is understood that each cluster is divided to some implicit sub clusters that each sub cluster may have different content in comparison to the others.

The main reason for this phenomenon is that, as the similarity accuracy is decreased, more urls from a dataset get a chance to be assigned to a cluster with a specific centroid, while their common tags with centroid url may be less than when the similarity accuracy was higher. So if the tags are assumed as an important factor that represents the content of the web pages, it is possible that a web page which is about *downloading free video* and tagged with a ‘free’ tag, and there is another web page about *downloading programming software codes* tagged with ‘free’ again, fall into the same cluster. Here, as we decrease the similarity accuracy, the presence of the ‘free’ tag among the other tags in the tag vector causes these two urls to be placed in the same cluster while they are not similar in content.

In the experienced case, when the similarity accuracy is 0.09, the number of urls connected to C0 (the first cluster) reached to 15. By reviewing them, three sub groups were discovered in one cluster. The content of urls in one of these three sub groups is about *video*, the other one is about *photography* and the third one is about *education*. Also two or three irrelevant urls could be seen in the collection. This multi grouping of web page urls in one cluster is the result of decreasing similarity accuracy and it cause urls with lower common tags with centroid tag vector and with different content to get a chance to be assigned to the same cluster.

For example most of the urls related to ‘video’ entered in the cluster because of the presence of tag *free*, while ‘free’ is a very general word for tagging and does not reflect the special content.

Note:(Multi group in one cluster) By decreasing the similarity accuracy rate, the number of web page urls that were put together as similar web pages in one cluster were increased, as the similar pages were determined based on their common tags; and when the centroid tag vector size is large the probability that urls with different contents and lower common tags, get a chance to be assigned to the same cluster is increased. This will cause some sub groups of urls with different content in one cluster.

The cluster about ‘education’ was formed because of the presence of tag ‘education’ among the tags collection of C0 (is a url of *Nasa* site). Again because ‘education’ is a relatively general word, it could not represent the content, so that the content of web pages in this cluster vary. There is a web page about military education, the two other ones introduce two games for teaching different concepts. One of them consists of games and puzzles as learning aids for teaching school books concepts to school children. The other one introduces thinking games about educating, training and improving thinking skill.

These facts are improved, even when the k-mean algorithm is not applied. By increasing the threshold of similarity accuracy, the number of urls in one cluster may be decreased, but the urls in one cluster are more similar, while by decreasing the rate of accuracy the number of urls in each cluster increases, but the number of real similar web pages decreases. Also some sub clusters with different topics originate. In other words, it seems that by increasing the rate of similarity accuracy threshold, the rate of similarity between selected urls increases, or we have the groups of related urls with higher percentage of similarity; however the number of related urls in each cluster decreases. By decreasing the rate of similarity accuracy threshold, the number of related urls in each cluster increases. But these web pages may not be really similar as their rate of similarity accuracy is fairly low.

As shown in Table 1 and 2, all four urls around centroid C1, are relevant to it. All of their contents are about ‘food’. By paying attention to the centroid tag vector, we see that the only tag in its vector is ‘food’, so this causes just web pages with food content to be put in this cluster, so all web pages are similar in this case. Since food is a specific word that is not too general, and directly points to a special group of data, we do not have multi grouping in one cluster. Also we can determine a topic ‘food’ for the cluster.

In contrast, in the third cluster with C2 centroid, not only the size of the tag vector is large, but also there are a number of general tags in the

vector. These two, cause too many irrelevant urls to be gathered in one cluster. Also we have a multi group of web page urls with specified and different content in one cluster (this matter is explained more in the above note.)

For example one of the tags in its vector is ‘google’, which is too broad word and included a range of web pages under different subjects, resulting in too many irrelevant pages in one collection.

3.1.3 Study on the effectiveness of centroid selecting method in clustering quality

The test was repeated, but the selection of centroid was not random this time. The urls which have maximum number of relation with other urls were selected. In other words, from the matrix which contains the pair of urls that have a degree of similarity with each other, the ones which have maximum number of relations, were selected. These urls were usually the ones that have the maximum number of tags in their tag vector. These kinds of urls can attract more urls in their clusters because the probability of having more urls that have similar tags with this centroid url is increased, and they can form populous clusters. So populous clusters are the ones that have a centroid with a large size of tag vector, and consist of general tags and could cover web pages with different contents.

The following numerical results were obtained from the test when the centroid selection type changed from ‘random’ to based on ‘maximum number of correlation’.

- Similarity accuracy = 0.3
- Centroid selecting method = urls with maximum number of correlation

Table 3: Comparing Similar urls clustering quality when similarity accuracy is 0.3 and centroid selecting method is not random.

K-Mean	Centroids	Number of Correct clustered urls	Number of Total urls in cluster	Percentage of clustering quality
	C0	6	14	61%
	C1	10	14	
	C2	7	10	
Non K-Mean	C0	6	14	57%
	C1	10	14	
	C2	7	12	

The following sentences explain the effectiveness of centroid selecting method when the similarity accuracy decreases from 0.3 to 0.09:

Consider two different methods of determining centroid, ‘random’ selecting centroid and selecting centroid based on ‘maximum number of correlation’, Table 1 and Table 3, it is inferred from Table 3 that in the case in which the similarity accuracy is high (0.3), there is no perceptible

difference between the result acquired from each one of these two centroid selecting methods. But when the similarity accuracy decreased to 0.09 and the centroid selecting method was based on the 'maximum number of correlation', the total number of urls in each cluster increased so much that it was not possible to open all urls, review their contents, and generate numerical statistics. But by investigating the number of urls and comparing their tag vectors, it could be inferred that the total number of urls in each cluster suddenly increased, while the number of urls that were really similar in content in each cluster almost did not change, and this caused the percentage of clustering quality to decrease.

So this experience shows that selecting the centroid based on the maximum number of correlation is not appropriate.

4. Conclusion and future work

4.1 Summary of Contribution

Tagging behavior analysis of data acquired from a bookmarking site like delicious and demonstrating the visual objects, urls' semantic relations in the form of graph structure were the aims of this paper. To achieve these aims, the following process have been implemented in java code:

1. Clustering web pages that are similar in content, based on the similarity between their tag vectors.

For this process the following methods and algorithms were implemented:

- Similarity measurement methods: Simrank, Cosine and Pythagorean
- Clustering algorithm: K-Mean

2. Then the result of the process has been represented in the form of graph structure. A powerful java library called 'Prefuse' has been applied for implementing this visualization phase of project.

3. For studying and analyzing tagging behavior, some statistical demonstrations have been required, so according to the data the program generates, the statistical results are reflected in tables and Excel spreadsheet.

Regarding that the basic element in this process is 'Tag', in a separate part, tagging behavior in the process was discussed and the statistical evaluation and analysis have been done.

Through the provided work and analysis performed, some new findings and interpretation from the tag relations has been acquired.

In similar web pages' clustering analysis, the effectiveness of different kinds of tags (general or specified) in url clustering quality, and the impact of important factors like rate of similarity accuracy in the severity of similarity, presence of general tags in the centroid tag vector, the size of centroid tag vector, and the centroid selecting method in similar web pages' clustering, were discussed.

The framework and steps of the project were explained and more details about the application steps implementation, visualization process and program user interface are also shortly explained.

In this project, the clustering algorithm works in a way that just similar web pages to centroids are displayed, and other urls, which may be similar and related to each other will not be displayed; it is not good as users want to see all related urls beside clustered urls.

4.2 Future work

Many users are interested in finding web pages that are similar in content to the one they are studying. In this work, clusters of similar web pages were determined and represented. This work could be simply extended to suggest similar web pages to the web page url given by a user, based on the tag vectors.

One of the problems in folksonomy sites like delicious is that users can choose any word as tag, whether meaningful or not, relevant or irrelevant to the content for annotating web pages, without any limitation, so that the tag space in these site are usually messy and ambiguous, and categorizing tags in this space is very hard work. The presence of a recommendation system for tag choosing could somehow limit users from selecting any miscellaneous and irrelevant words, and finally lead to many advantages like better similar url clustering. The basic activity for achieving this improvement is to extract co-occurrence tags from datasets.

In this work the proposed system could be extended to generate a tag recommendation system providing the user appropriate tags and resources related to a topic of interest as specified by the user. In this case the desired web page would be automatically labeled using more relevant tags, and as a consequence the users could then share their resources more efficiently.

References

Journal Papers:

- [1] Valentin Robu, Harry Halpin and Hana Shepherd, Emergence of Consensus and Shared Vocabularies in Collaborative Tagging Systems, *ACM Transactions on the Web*, Vol. 3, No. 4, September 2009.
- [10] Marco Luca Sbodio and Edwin Simpson, Tag Clustering with Self Organizing Maps, *HP Laboratories*, October 5, 2009, 338.

Books:

- [5] Kardi Teknomo, *K-Means Clustering Tutorial* (Stullu, Jun 03, 2014)

Chapters in Books:

- [3] Kaikuo Xu; Yu Chen; Yexi Jiang; Rong Tang; Yintian Liu and Jie Gong1, A Comparative Study of Correlation Measurements for Searching Similar Tags, *Advanced Data*

Mining and Applications Lecture notes in Computer Science,
(New York: Springer, Volume 5139, 2008) pp 709-716

Thesis:

[2] Mitja Koren , *Combining ontologies with social tagging systems How current tagging systems can be improved using Semantic Web techniques*, Master thesis, Feb 11, 2009.

Proceeding Papers:

[4] Glen Jeh and Jennifer Widom, SimRank: a measure of structural-context similarity, International Conference on Knowledge Discovery and Data Mining, Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, New York, 2002, 538-543.

Thesis:

[7] Ian Davis and Talis Systems Limited, *Supporting Change Propagation in RDF*, Paper submitted, 2005.

<http://www.w3.org/2009/12/rdf-ws/papers/ws0>

[8] Brian McBride, Updated by: Daniel Boothby and Chris Dollin, *An Introduction to RDF and the Jena RDF API*, http://jena.sourceforge.net/tutorial/RDF_API/, Id: index.html,v 1.24 2009/02/09 11:53:07 andy_seaborne Exp.

[9] Konstantinos Kanaris, *Jena Framework*, www.icsd.aegean.gr/kotis/OE&SW'07/myPresentations/Jena.ppt

[11] Hakan Duman, Alex Healing and Robert Ghanea-Hercock, *Adaptive Visual Clustering for Mixed-Initiative Information Structuring()*

portal.acm.org/citation.cfm?id=1601544.1601591

Websites:

[6] <http://prefuse.org/> , under the terms of the [GNU Free Documentation License](#), visited on 12/10/2014.

Appendix A: Java code of application