Building a Concept Hierarchy Using Frequent Tag Sequences

ABSTRACT

Web sites that allow collaborative tagging of resources have become a commonplace development. As part of the second generation of applications available on the Web, these sites provide a tremendous amount of user-generated taxonomic information. However, information seekers are hindered by the lack of organization within these tags. To address this issue, several methods have been proposed for creating an organizational structure from the tags. Despite their benefits, the current methods do not directly represent an organization of concepts, as a concept is often composed of more than one tag. In this paper, we propose a new approach to generating a concept hierarchy from the user-generated tags. Exploiting the fact that users often express a concept over a set of sequential tags, we propose a two-step approach for generating a hierarchy of concepts. We first discover concepts through tag sequences with sufficient support. Using these concepts, we then calculate conditional probabilities to discover the existing hierarchical relationships. The key benefit of the hierarchy produced through our approach is that it is topic-based, as opposed to existing related work, which only produce hierarchies of tags. Our findings are illustrated on a domain-specific dataset of tags supplied by a popular collaborative tagging Web site.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database applications—*Data Mining*

General Terms

Algorithm, Experimentation

Keywords

Folksonomy, Tag Sequences, Sequence Mining, Subsumption, Concept Hierarchy

1. INTRODUCTION

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...$10.00.

Web sites that allow collaborative tagging of resources have become a commonplace development. Users within these sites are allowed to provide their own annotations (*tags*) to classify resources, where a resource can be any digital object. The process of tagging is similar to the process of keyword indexing that librarians and professional curators usually perform. However, the main distinction with tagging is that there is no use of a formal taxonomy to guide the process. The resulting collection of user-generated tags is called a *folksonomy*[1].

The word *folksonomy* is the combination of the words *folk* and *taxonomy*, emphasizing the fact that the taxonomic information generated within these Web sites is done by common users (*folk*). The main advantage of folksonomies, as compared to formal taxonomies, is the low cost in building and assigning resources. Unfortunately, this benefit also becomes a hindrance. Folksonomies allow users to enter tags containing misspellings, non-alphanumeric symbols, abbreviations, along with other items that introduce noise to the taxonomy.

Despite the challenges of dealing with the noisy data, several approaches have been proposed for organizing user-generated tags [6, 8, 10, 11, 12]. The structures generated in these approaches vary from a forest of trees such as in [8, 12] to a directed acyclic graph as in [6, 11] to clusters of directed graphs as in [10]. The current approaches, although promising, lack the ability to organize topics into a hierarchy of concepts, where a concept can span a sequence of more than one tag.

In this paper, we propose the following:

- A new approach for discovering domain-specific tags from a folksonomy. The proposed approach leverages domain sources such as reliable Web sites to seed the set of domain-specific topics for querying collaborative tagging sites.
- An algorithm for discovering important tag sequences. Utilizing these identified tag sequences, we construct a domain-specific concept hierarchy, visually representing topics and relationships.
- An implementation of our overall approach on the domain of information specific to patient and customer health.

This paper is organized as follows. Work related to this study is outlined in Section 2. Section 3 details our methodology and the steps performed in this research. Section 4 defines a frequent tag sequence and provides the algorithm for discovering one. Section 5 describes our concept hierarchy and details the algorithm for creating one. Section 6 provides results. Finally, we discuss our conclusions and future work in Section 7.

2. RELATED WORK

Several studies have investigated the organization of user-generated tags [6, 8, 10, 11, 12]. The main challenge common to all proposed techniques is dealing with noisy data present within the folksonomy tags. Each of these approaches produce the organizational structure among the tags in a different manner.

In [8], the tag grouping algorithm proposed uses graph centrality to sort tags. At each step, the algorithm adds the next most central node to the graph. Using the similarity measure, the new node is either added as a child to the most similar node in the graph, if the similarity is greater than a threshold value, or it is added as a child to the root signaling a new topic in the graph. The resulting structure is a forest of tag hierarchies, instead of a DAG. Similar structure is achieved in [12]. The clustering method is a variant of agglomerative clustering using parameters to control the groping of nodes into clusters and the hierarchy level.

When searching for communities within social and biological networks, an alternative approach is applied in [7]. Their algorithm recursively removes the least central edges within the graph until the desired number of communities is reached or until the complete hierarchy is built.

In [6], tags are described using the PLSI (Probabilistic Latent Semantic Model) vector model. The similarity between edges is described with the JS divergence measure, to describe the distribution between probabilistic distributions. To link the nodes in the graphs, the entropy of the PLSI vector is used; and for each node, a node among the k nearest neighbors is added as child if its entropy is less that the entropy of a target node. The resulting structure is a DAG as opposed to a forest of hierarchies or groups of sub-graphs. The other feature of the approach is that is can be applied online, as it allows a target node to be expanded locally without having to re-construct the graph.

In [9], the subsumption hierarchy calculation was introduced as a means to derive conceptual topic/sub-topic relationships. Building off of this work, [11] utilized the co-occurrence of single tags within Flickr[1] to create a concept hierarchy. This work provided promising results; however, it did not take into account sequences of tags when constructing the concept hierarchy.

Discovering association rules among tags, users, and resources is described in [10]. A graph is constructed by creating a node for all premises and conclusions in the association rules and connecting the each pair of nodes where an association rule satisfies the support and confidence thresholds. In in this approach it is possible that within an association rule the premise or conclusion contains more than one tag; however, tag sequences are not specifically addressed.

Folksonomies represent an organization of representative topics shared by a large number of users. Topic identification can be mapped to the problem of discovering frequent sequences of tags. [5] proposed an algorithm for finding frequent maximal text sequences. This algorithm is a refinement of the algorithm for discovering sequential patterns introduced in [4]. Discovering sequences within itemsets is itself an adaptation of the problem of finding frequent itemsets in data [2].

[1]http://www.flickr.com

3. METHODOLOGY

3.1 Overview

This section outlines how to create a domain specific concept hierarchy from the user-generated tags found within a folksonomy. Figure 1 highlights the three-step process of: (1) Retrieving and cleaning a seed set of topics from reliable domain sources; (2) Querying, extracting, and cleaning domain-specific tags from one or more folksonomies; and (3) Creating the domain's concept hierarchy from the user-generated tag sequences. It is possible to perform any one of these steps more than once, especially to keep the concept hierarchy up-to-date. For example, Step (1) will need to be performed to keep the seed set of domain terms up-to-date. Moreover, performing steps (2) and (3) will keep the concept hierarchy in sync with the current topics. The framework introduced here can be applied to any domain as long as a quality set of seed keywords can be obtained. We specifically applied this methodology to domain of consumer and patient health topics and provide detail on each of the three steps performed in the rest of this section.

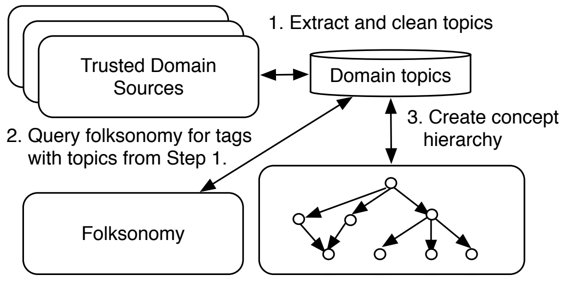
1. Extract and clean topics
Trusted Domain Sources
Domain topics
3. Create concept hierarchy
2. Query folksonomy for tags with topics from Step 1.
Folksonomy

Figure 1: Framework for the generation of the concept-based hierarchy using user tags.

3.2 Retrieval and Cleaning of Domain Topics

The first step in building a health concept hierarchy from folksonomy tags was to identify a seed set of consumer health topics. We started with a subset of 100 trusted health Web sites provided by the Consumer and Patient Health Information Section (CAPHIS) of the Medical Library Association Web site[2]. Our set consisted of the sites that allowed crawling, as indicated by each of the site's *robot.txt* file[3]. After identifying our allowable list of sites, we parsed each of the site's A-Z index glossary listing of terms and extracted the terms found within link's anchor text. For example, the glossary page for terms starting with "F" provided links to terms such as the following: *Flu*, *Fluoride*, *Food Allergies*, and *Fractures*, among others. We also performed the same term extraction process within the other main health categories available on these Web sites. For example, several

[2]http://caphis.mlanet.org/
[3]http://www.robotstxt.org/

of the sites contained links targeting patients of different ages: *Children's Health* and *Seniors' Health*; categories for both genders: *Men's Health* and *Women's Health*; as well as other health categorizations: *Diseases & Conditions* and *Treatments & Procedures.*

The list of downloaded topics was then processed for cleaning. We removed non-alphanumeric symbols and common stop words such as: *the*, *of*, *and*, etc. We also split the entities that included several related topics, such as *Acute Coronary Syndromes (Heart Attack; Myocardial Infarction; Unstable Angina)*, into separate topics. Our final step was to convert all words to lower case. After eliminating duplicate topics, the set of total unique health topics extracted from the seed Web sites totaled $11,679$.

3.3 Extracting Domain-Specific Tags

In order to target the subset of tags relating to health subject matters available from a folksonomy, we used the seed set topics discovered in the previous steps as the keyword tags. We chose to use Delicious as our source folksononmy for this research. Utilizing its data feed API[4], we generated a query for each of the $11,679$ health topics.

The advantages of using the API include the availability of different options to query the site and the availability of structured data in the query results that is more manageable for the next processing steps. The results provided from the API are presented as a set of entries. Each entry corresponds to an annotation record created within the Delicious folksonomy, where a resource (a Web page URL) has been annotated by a user, at a specific time, and with a set of tags. The annotation record is uniquely identified by the combination of the URL of the resource and the user who annotated it.

For topics consisting of only one word, we combined the topic term with the term *health* to retrieve results with both tags from Delicious. For all other health topics consisting of more than one term we used the terms as tags to query Delicious. We retrieved a total of $163,128$ annotation records from Delicious for use within this study.

The downloaded tags were cleaned in the same manner as performed on the seed topics described above. During this cleaning process, any single tag consisting of more than one word concatenated with a non-alphanumeric symbol was split into more than one tag. For example, the tag *Heart_Disease* split into two tags: *heart* and *disease.* During the analysis of the tags generated by users, we discovered that users also concatenated more than one term with the empty space ''. For example, the tag *HeartDisease* consists of more than one word; however, we did not attempt to split tags of this form.

3.4 Creating the Concept Hierarchy

After the tags have been retrieved and cleaned, the next step is to discover any frequent tag sequences. Once the frequent tag sequences are discovered then a directed graph is generated to expose the concepts and relationships that exist within these tag sequences. The nodes of the graph are created from the sequence of tags and edges are created between any two nodes, if there exist a topic/sub-topic relationship. We utilize the subsumption relationship described in [9] to establish the association between the nodes. The next two sections describe in detail the algorithms developed to create frequent tag sequences and construct the concept hierarchy.

[4]http://delicious.com/help/json/

4. FREQUENT TAG SEQUENCES

Discovering sequential patterns within the annotation records of the folksonomy allow concepts of length greater than one to be identified. For example, the health topics of *heart disease* and *acute coronary syndrome* span more than one term and the sequential order of these terms is semantically important. It is our observation that when annotating resources associated to topics such as these that users will place the tags in the same order as the words in the topic. Therefore, discovering sequences of varying lengths allow us to investigate the relationships of topics, and not just single tag-to-tag relationships. In this section, we first provide formal definitions to clarify the problem of discovering sequences of tags that are frequently found in folksonomy annotation records. We then detail the process on how we discover these sequences.

4.1 Definitions

We define the data within a folksonomy in much of the same manner as it is defined in [8, 10, 12], with a couple of distinct differences.

Definition 1. A *folksonomy*, $\mathbb{F}$, is a collection of the finite sets U, R, T, D, A where

- U, R, T, D represent *users*, *resources*, *tags*, and *datetimes*, respectively.
- A is a set of annotation records. An item $a \in A$ has the following format: $(u, r, \{t_1, ..., t_n\}, d)$, represending the user, $u \in U$, who annotated a resource, $r \in R$, with a set of tags, $\{t_1, ..., t_n\}$, where $t_i \in T$, on a date and time, $d \in D$.

A folksonomy allows users to signup to participate, thus creating a unique user id. Users provide tags at their own discretion to annotate a resource. A tag can be any type textual string whether the string consists of a word, a term, an acroynm, or any combination thereof. The type of resource that is tagged is dependent on the folksonomy itself. Many of the popular folksonomies existing today allow users to either tag a URL to a specific Web page (i.e., Delicious), tag an image (i.e., Flickr), or tag a purchasable item (i.e., Amazon[5]).

Definition 2. A *tagset*, within a folksonomy, is the set of m tags $\{t_1, t_2, ..., t_m\}$ created by a user to annotate a resource at a specific date and time.

The *tagset* is analogous to the set of items purchased (an itemset) within a transactional database system [2]. A distinction exists, however, between an itemset and an item sequence. Within an item sequence, the order in which the items appear must be considered. For example, it is semantically important to know that tags t_i, t_{i+1}, t_{i+2} (i.e., $t_i =$ *attention*, $t_{i+1} =$ *deficit* $t_{i+2} =$ *disorder*) appear in this order. Therefore, it is essential to consider the notion of a *text sequence* explained in [5] as well as the concept of an *item sequence* discussed [4].

[5]http://www.amazon.com

Definition 3. A *tag sequence* is a serial arrangement of n tags, $\langle t_1, t_2, ..., t_n\rangle$ existing within a tagset. We refer to a tag sequence of length n as a n-sequence.

A sequence $\langle s_1, s_2, ..., s_m\rangle$ is a sub-sequence of another tag sequence $\langle t_1, t_2, ..., t_n\rangle$, if $m \leq n$ and $\exists j \in \{0, ..., n-m\}, \forall i \in \{1, 2, ..., m\} : s_i = t_{i+j}$.

Definition 4. The *support for a tag sequence* is defined as the fraction of the total tagsets containing the tag sequence. Therefore, a tag sequence is *frequent* if its support is greater than a minimum support threshold σ. Similar to above, a frequent n-sequence, is a sequence of length n that is frequent.

4.2 Finding Frequent Tag Sequences

We split the task of discovering frequent tag sequences into two phases: (1) the *Transformation Phase* and (2) the *Discovery and Expansion Phase*. The *Transformation Phase* converts the information retrieved from Delicious into a format for finding frequent tag sequences. The *Discovery and Expansion Phase* iterates through n-sequences discovering frequent n-sequences and then expanding them to create candidate $(n+1)$-sequences for the next iteration. Each of the phases is described below.

Transformation Phase. The first phase of the algorithm transforms an annotation record into a set of 1-sequence records. The annotation record has the form: (tid, u, r, t, d), where *tid* is the databbase primary key for this record and u, r, t, and d are defined by $\mathcal{A}$ above. The 1-sequence records have the form: $(tid, tag_i, index_i)$, where tag_i represents the ith tag in the sequence and $index_i = i$. For the following annotation record:

Field	Value
tid	1
u	‘johndoe’
r	‘http://www.cdc.gov/’
t	{‘centers’, ‘disease’, ‘control’, ‘prevention’ }
d	‘2009-JAN-01 12:00:00 AM’

the resulting 1-sequence records are generated:

tid	tag1	index1
1	centers	0
1	disease	1
1	control	2
1	prevention	3

In general, to represent an n-sequence record, we create a $2n+1$ tuple of the following form:

$$(tid, tag1, .., tagn, index1, ..., indexn).$$

It is also possible to represent the n-sequence record with an alternative format of

$$(tid, tag1, .., tagn, index1, length);$$

however, we chose to implement the record using the first approach.

Discovery and Expansion Phase. After the 1-sequence records are created in the *Transformation Phase*, the *Discovery Phase and Expansion Phase* will begin. At each level $(n = 1, 2, ...)$, the *Discovery and Expansion Phase* will discover frequent n-sequences from the current set of n-sequences and then expand these frequent sequences into candidate $(n+1)$-sequences to be used in the next iteration of the loop. This approach to building frequent tag sequences of length n has the same foundation as the *apriori*[3] algorithm for finding frequent item sets, the *AprioriAll*[4] algorithm, and the algorithm for finding maximal frequent sequences in text presented in [5].

4.2.1 Algorithm

generate-frequent. The `generate-frequent` function is the main function for generating the frequent tag sequences. It takes as parameters C_1 and σ and returns the set of all frequent sequences discovered, F. C_1 is the set of 1-sequence records generated in the *Transformation Phase*. σ is the minimum support threshold to use when discovering frequent sequences. We chose $\sigma = .04\%$. In our implementation, σ is then converted from the support threshold value (.04%) into a frequency count value by multiplying the number of annotations records with σ.

```
function generate-frequent($C_1, \sigma$)
$i = 1$
$F = \{\}$
while $C_i \neq \emptyset$
  $F_i \leftarrow$ find-frequent($C_i, \sigma$)
  $C_{i+1} \leftarrow$ generate-candidates($F_i$)
  $F \leftarrow F + F_i$
  $i$++
return $F$

function find-frequent($C_i, \sigma$)
$F_i = \{\}$
foreach $t \leftarrow$unique $\langle tag1, ..., tagi\rangle \in C_i$
  if $t.count() > \sigma$
    $F_i \leftarrow F_i +$ all sequences in $C_i$ containing $t$.
return $F_i$

function generate-candidates($F_n$)
$C_{n+1} = \{\}$
foreach $f \in F_n$
  foreach $S \leftarrow$ set of $f$s with the same $tid$
    $Sort(S)$ by $index1$
    for consecutive records $s_i, s_j \in S$
      if $s_i.index1 + 1 == s_j.index1$
        $C_{n+1} \leftarrow C_{n+1} +$
          $(s_i.tid, s_i.tag1..n, s_j.tagn, s_i.index1..n, s_j.indexn)$
return $C_{n+1}$
```

find-frequent. The `find-frequent` function takes as parameters C_i and σ. C_i is the set of candidate frequent i-sequences and σ is the support threshold. For each unique tag sequence, if the count of records containing the i-sequence $\langle tag1, ..., tagi\rangle$ is greater than σ then all of the i-sequence records containing the sequence are added to the frequent sequence set, F_i, that is returned.

generate-candidates. The `generate-candidates` function is similar to `apriori-generate` described in [4]; however, `generate-candidates` does not join the set of frequent n-sequences with itself to create all of the candidate $(n+1)$-sequences. Instead, one pass is made over the set of frequent n-sequences and the tag indices of each n-sequence record are utilized to create the candidate $(n+1)$-sequences. The other benefit of this approach is that there is no need for a step to prune any $(n+1)$-sequences with a non-frequent n-sequence because all of our candidate $(n+1)$-sequences contain a frequent n-sequence.

The generate-candidates function takes as a parameter F_n, the collection of frequent n-sequence records sorted by *tid*. The algorithm will iterate through each frequent n-sequence record, and for each grouping of records with the same *tid*, the algorithm will sort the set of records by the first index ($index1$). If the $index1$ value for two consecutive records, within this group, is offset by one then the function will create the candidate $(n+1)$-sequence record using the n tags $(tag1, ..., tagn)$ and n indices $(index1, ..., indexn)$ from the first record and the nth tag $(tagn)$ and nth index $(indexn)$ from the second record. This is possible because the last $n-1$ indices of the first record will match the first $n-1$ indices of the second record, based on how the n-sequence records have been defined. C_{n+1}, the set of candidate $(n+1)$-sequence records, is returned.

5. HEALTH CONCEPT HIERARCHY

A concept hierarchy is a directed graph that places more general concepts closer to the root and pushes more specific topics closer to the terminal points of the graph. Directed edges create a parent-to-child relationship between two nodes within a graph. The parent node *subsumes* the child node, if the parent's concept is more general than the child's[9]. Using conditional probabilities, the concept hierarchy algorithm creates a directed edge between nodes d_1 and d_2, noted $d_1 \longrightarrow d_2$, if $P(d_1|d_2) > \theta$ and $P(d_2|d_1) < P(d_1|d_2)$, where θ is the subsumption threshold. From our experiments, we chose $\theta = 0.7$. The conditional probabilities are calculated using the tagsets containing d_1 and the tagsets containing d_2.

A candidate node for the hierarchy is created for each frequent n-sequence. The value displayed for the node consists of concatenating the n tags with a space (' '). For example, if the frequent tag sequence discovered is ⟨'post', 'traumatic', 'stress', 'disorder' ⟩ then the label for this node is 'post traumatic stress disorder.'

5.1 Algorithm

The entire concept hierarchy can be constructed from the candidate node tuples generated in generate-candidate-nodes and the set of edges generated in generate-edges.

```
function generate-candidate-nodes($F$)
$N = \{\}$
$i = 0$
for $f \in F$
  $N \leftarrow N + (i, \text{concat}(\langle tag1, ..., tagn \rangle, ` '))$
  $i{+}{+}$
return $N$

function generate-edges $(F, N, \theta)$
$E = \{\}$
for $n_i, n_j \in N$ where $n_i \neq n_j$
  if $P(n_i|n_j) > \theta$ and $P(n_j|n_i) < P(n_i|n_j)$
    $E \leftarrow E + (n_i \longrightarrow n_j)$
  else if $P(n_j|n_i) > \theta$ and $P(n_i|n_j) < P(n_j|n_i)$
    $E \leftarrow E + (n_j \longrightarrow n_i)$
return $E$
```

generate-candidate-nodes. The generate-candidate-nodes function takes as a parameter, F, the set of frequent tag sequences discovered in the generate-frequent function described above. The set of node tuples, N, is returned. A node tuple, $(nid, value)$, contains a unique node id, nid, and the value for the node, *value*. The value is created by concatenating the tags with a space, as described above. The nodes generated in this function are considered *candidates* because it is possible one or more may not appear in the hierarchy, if it is not connected to an edge.

Table 1: Health Sequences (Seq.) Discovered

Seq. Lgth.	Seed Seq. Ct.	Discovered Tag Seq. Ct.
1	2,202	295,656
2	4,176	70,058
3	2,835	6,022
4	1,346	546
>4	1,120	54
Totals	11,679	372,336

generate-edges. The generate-edges function takes three parameters: F, N, and θ and returns E. F is the set of frequent sequences discovered in the generate-frequent function. N is the set of candidate nodes generated in generate-candidate-nodes. θ is the conditional probability used to create an edge. For our implementation, we chose $\theta = 0.70$. The set of edges, E, is returned from generate-edges. Each directed edge from n_i to n_j is denoted $n_i \longrightarrow n_j$, where n_i is known as the *source* of the edge and n_j is known as the *target* of the edge.

The two conditional probability calculations of nodes n_i and n_j are denoted $P(n_i|n_j)$ and $P(n_j|n_i)$. These values are calculated using the tagsets associated with each node, which can be easily discovered because a node is generated from a frequent tag sequence $\langle tag1, tag2, ..., tagn \rangle$ and the n-sequence record $(tid, tag1, ..., tagn, index1, ..., indexn)$ contains the tagset id, *tid*, along with the sequence of tags. Therefore, a collection of tuples containing the node id and tagset id, (nid, tid), can be used to calculate the conditional probabilities.

When generating the concept hierarchy, our graph construction algorithm creates a virtual node labeled $\{root\}$ to connect all nodes that are not subsumed by another node. These nodes are not a sub-topic of another node and thus placed at the top level of the concept hierarchy. The figures provided in the Section 6 display this virtual node.

6. RESULTS AND DISCUSSION

The $11,679$ health topics originally extracted from the health Web sites where discovered $372,336$ times within the downloaded and cleaned tag sequences. Table 1 provides analysis on the health topic sequences. To further analyze these numbers, we split the health topics by term length. The sequence length column, *Seq. Lgth.*, has separated the topics into their appropriate lengths bucket. The *Seed Seq. Ct.* column details how many of the seed health topics have the respective length. The *Discovered Tag Seq. Ct.* column shows how many times a health topic of the associated length was discovered within the tag sequences from Delicious. It can be seen that of the $372,336$ times a health topic was discovered that 79% were of length *one* $(295,656$ times). Although, this was the greatest majority of items discovered, the analysis indicates 20% of the topics discovered where of length *two* or *three* $(76,080$ times).

Our resulting concept hierarchy has a total of $3,204$ nodes and $20,785$ edges. Of the $3,204$ nodes within the graph, 404 were not subsumed by another and thus considered the top concepts. Additionally, 1,805 of the 3,204 nodes were created from 1-sequences and the other 1,399 were created from n-sequences, where $n > 1$.

Figures 2 and 3 display the change in frequent tag sequences associated with the tag *flu* over the course of our study. As the topics of *h1n1* and *swine flu* gathered public attention during the end of the month of April 2009, so too did the frequent tags sequences associated with these topics. Figure 2 displays the following sub-topics associated with the flu: *colds*, *cold* and the *flu*; *pandemics*; *shots* and *flu shots*; and *bird*, *bird flu*, *avian*, *avian flu*, and *h5n1*. Figure 3 displays all of the data queried from Delicious during this research. As it can be seen additional topics revolving around *pandemics*, *swine flu*, and *h1n1* became frequent.

The interconnectedness between several topics can be seen within the graph. Our visualization provides a very easy way to see the progression of frequent topics among the tag sequences. The time dimension associated with each annotation record provides the ability to study the evolution of health topics important to consumers and patients, such as disease outbreaks, current diet and nutrition practices, and treatments for diseases such as cancer, just to name a few.

Figure 4 displays the concepts related to *eating* and *disorder*. There are several concept communities related to *disorder* that have emerged from the frequent tag sequences: *borderline personality disorder*, *obsessive compulsive disorder*, *attention deficit/hyperactivity disorder*, *seasonal affective disorder*, *post traumatic stress disorder*, and *eating disorder*. Although these disorders are not interrelated, they are all specific types of disorders. The concept of *eating* introduced health concepts such as *healthy eating* and *eating well*, *binge eating*, as well as *eating disorder*. This figure provides a quality example of two distinct concepts (*eating* and *disorder*) and how they are connected together by the sub-topic of *eating disorder*.

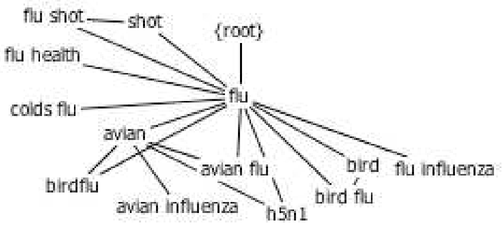

Figure 2: Graph created for tag sequences associated with the tag *flu* - tags before April, 23 2009

Figures 2 - 4 all display concept hierarchies related to specific tags. Several of the topic/sub-topic relationships that exist within these graphs are simply displaying a single term topics that subsumes a two-term topic, where the single term is contained within the two-term topic. For example, the directed edge $flu \longrightarrow flu\ shot$. Although this relationship may seem obvious, the fact that an edge appears between these two concepts underscores the fact that the the two concepts are frequent and a conditional probability was satisfied between them.

7. CONCLUSIONS AND FUTURE WORK

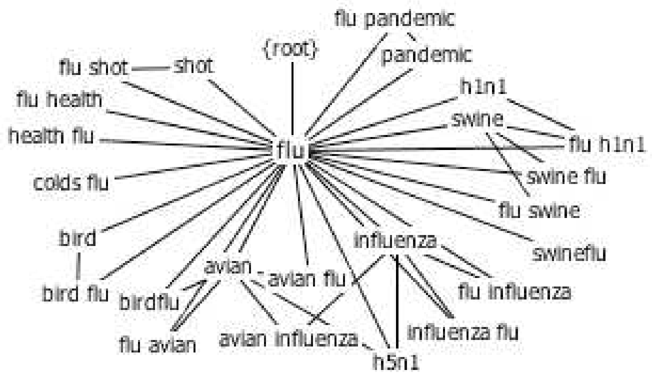

Figure 3: Graph created for tag sequences associated with the tag *flu* - all tags

In this paper we introduced the concepts of tag sequences and frequent tag sequences as applied to folksonomies. We used a seed set of patient and consumer health topics mined from reliable health Web sites to query the social tagging Web site Delicious. Using the collection of user-generated tags retrieved from Delicious, we were able to discover frequent tag sequences of length $1..n$. Utilizing these tag sequences, we then created a directed graph to establish the concept hierarchies that existed within the user-generated content.

Our findings provide a way to visualize the relationships that exist within the folksonomy by only concentrating on the most frequent tag sequences. Additionally, frequent concepts spanning more than one tag are displayed, something of which is not possible when using hierarchical clustering or graph creation methods utilizing only tag-to-tag similarities.

In the future we plan to extend this work to:

- Incorporate the associations between the tagsets generated by different users for the same URL. If we utilize a two-step approach of (1) finding URLs based on the seed topics and then (2) retrieving all tagsets associated with this resource, then it is possible to establish additional semantic relationships between topics.

- Detect and remove tagsets highly regarded as spam. It has been discovered through our current research effort that users may generate identical or nearly identical frequent n-sequences (where $n > 10$). Therefore, the approach of finding highly similar frequent, long tag sequences that do not produce relevant topical information can be detected as spam and removed from the concept hierarchy.

- Incorporate resources such as WordNet[6] to aid in discovering synonyms and hypernyms to assist in the construction of the concept hierarchy.

- Provide a way to easily visualize the concept hierarchy changes as the minimum support threshold for frequent tag sequences is manipulated.

8. REFERENCES

[6]http://wordnet.princeton.edu/

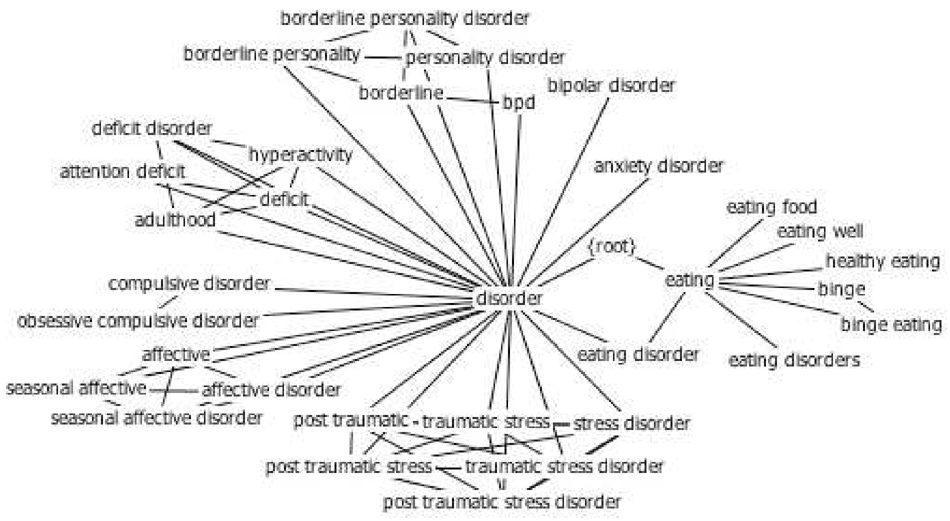

Figure 4: Graph created for tag sequences associated with the tags *eating* and *disorder*

[1] Folksonomy, 2009. http://en.wikipedia.org/wiki/Folksonomy.

[2] Agrawal, R., Imielinski, T., and Swami, A. N. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data* (Washington, D.C., 1993), P. Buneman and S. Jajodia, Eds., pp. 207–216.

[3] Agrawal, R., and Srikant, R. Fast algorithms for mining association rules in large databases. In *Proceedings of 20th International Conference on Very Large Data Bases* (1994), J. B. Bocca, M. Jarke, and C. Zaniolo, Eds., Morgan Kaufmann, pp. 487–499.

[4] Agrawal, R., and Srikant, R. Mining sequential patterns. In *Proceedings of the 1995 IEEE International Conference on Data Engineering* (1995), pp. 3–14.

[5] Ahonen-Myka, H. Finding all frequent maximal sequences in text. In *16th International Conference on Machine Learning ICML-99 Workshop on Machine Learning in Text Data Analysis* (Ljubljana, Slovenia, 1999), D. Mladenic and M. Grobelnik, Eds., pp. 11–17.

[6] Eda, T., Yoshikawa, M., and Yamamuro, M. Locally expandable allocation of folksonomy tags in a directed acyclic graph. In *WISE '08: Proceedings of the 9th international conference on Web Information Systems Engineering* (Berlin, Heidelberg, 2008), Springer-Verlag, pp. 151–162.

[7] Girvan, M., and Newman, M. E. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America 99*, 12 (June 2002), 7821–7826.

[8] Heymann, P., and Garcia-Molina, H. Collaborative creation of communal hierarchical taxonomies in social tagging systems. Tech. Rep. InfoLab Technical Report 2006-10, Stanford University, 2006.

[9] Sanderson, M., and Croft, B. Deriving concept hierarchies from text. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 1999), ACM, pp. 206–213.

[10] Schmitz, C., Hotho, A., Jäschke, R., and Stumme, G. Mining association rules in folksonomies. In *Data Science and Classification*. Springer Berlin Heidelberg, 2006, pp. 261–270.

[11] Schmitz, P. Inducing ontology from flickr tags. In *Proceedings of the Collaborative Web Tagging Workshop (WWW '06)* (2006).

[12] Shepitsen, A., Gemmell, J., Mobasher, B., and Burke, R. Personalized recommendation in social tagging systems using hierarchical clustering. In *Proceedings of the 2008 ACM conference on Recommender systems* (New York, NY, USA, 2008), ACM, pp. 259–266.