2009 IEEE International Conference on Data Mining Workshops

Discovering Domain Specific Concepts within User-Generated Taxonomies

Jonathan Klinginsmith*, Malika Mahoui†, Yuqing Wu* and Josette Jones†
*School of Informatics and Computing
Indiana University, Bloomington, IN, USA
Email: jklingin, yuqwu @indiana.edu
†School of Informatics
IUPUI, Indianapolis, IN, USA
Email: mmahoui, jofjones @iupui.edu

***Abstract*—Collaborative tagging of resources on the Web has become a commonplace occurrence. Web sites allowing resources to be tagged provide a tremendous amount of user-generated taxonomic information. However, information seekers are hindered by the lack of organization within these tags as well as the multitude of domains encompassed within these sites. To address these issues, we propose a multi-step approach for creating domain specific concept hierarchies from collaborative tags. Each concept hierarchy is based on domain specific subject matters, which may span more than one tag, as opposed to related work which are only concerned with the relationships between single tags.**

***Keywords*-folksonomy; sequence mining; suffix tree; subsumption; concept hierarchy**

I. Introduction

Collaborative tagging of resources on the Web has become a commonplace occurrence. As part of the second generation of the Web, users within these sites are allowed to provide their own annotations (*tags*) to classify digital resources. The resulting collection of user-generated tags is called a *folksonomy*.

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, compared to formal taxonomies, is the low cost in building and assigning resources, thus allowing communities of users to contribute to the classification process.

Because of this collaborative annotation effort, there is a great deal of user-generated taxonomic information to discover within folksonomies. However, the resources classified within these sites span a multitude of domains. As a researcher interested in a particular domain, whether for analyzing domain specific trends or using the data for marketing or advertising purposes, one would need to discover the taxonomic information for the particular domain of interest. As as result, a multi-step framework is needed to first extract the subset of appropriate domain specific tags and then to discover the concepts and relationships among the information extracted.

Several approaches have been proposed for organizing user-generated tags [1]–[5]. The structures generated in these approaches vary from a forest of trees such as in [2], [5] to a directed acyclic graph as in [1], [4] to clusters of directed graphs as in [3]. The current approaches, although promising, do not provide a methodology for discovering domain specific concepts within a collaborative tagging Web site. Additionally, the ability to organize topics into a conceptual hierarchy is absent from the current approaches, where a concept may span more than one sequence of tags.

In [6], the subsumption hierarchy calculation was introduced as a means to derive conceptual topic/sub-topic relationships. Building off of this work, [4] 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 multiple tags in a sequence.

Folksonomies represent an organization of topics shared by a large number of users. Topic identification can be mapped to the problem of discovering frequent sequences of tags. [7] proposed an algorithm for finding frequent maximal text sequences. This algorithm is a refinement of the algorithm for discovering sequential patterns introduced in [8]. The problem of sequential pattern mining and incremental sequence mining has been discussed in a collection of studies [9]–[15]. Expanding on this research, [16] introduced the concept of a decreasing support constraint on length-increasing sequences.

Towards discovering domain specific concepts within user-generated tags we propose the following:

- An approach for discovering domain specific tags from within a folksonomy. The proposed approach leverages curated domain sources such as reliable Web sites to seed the set of domain specific topics and terms for querying collaborative tagging sites.
- The use of a tag sequence suffix tree for discovering important sequences of user annotated tags.
- The generation of a concept hierarchy from the important tag sequences discovered.
- An implementation of our overall approach on the domain of information specific to patient and customer health.

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

978-0-7695-3902-7/09 $26.00 © 2009 IEEE
DOI 10.1109/ICDMW.2009.57

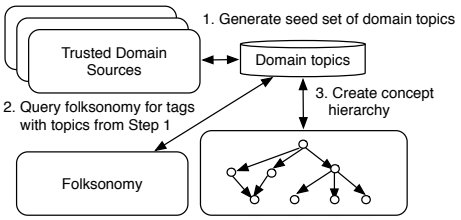

Figure 1: Framework for the generation of the concept-based hierarchy using user tags.

II. Methodology

A. 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: 1. Generate a seed set of domain specific topics from reliable sources associated with the domain; 2. Use this seed set of topics to query, extract, and clean the tags from one or more folksonomies; and 3. Create the domain's concept hierarchy from the user-generated tag sequences.

It is possible to perform any one of these three 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 or as other reliable domains sources are introduced. Moreover, performing Steps 2. and 3. will keep the concept hierarchy in sync with the current domain topics tagged by users. This framework is neither implementation nor domain specific. It can be applied to any domain as long as a quality set of seed keywords can be obtained. We specifically applied this methodology to the domain of consumer and patient health topics. The rest of this section provides details of each of the steps performed.

B. Generating a Seed Set of Topics from Reliable Domain Sources

The first step in building a domain specific concept hierarchy from a collaborative tagging Web site is to identify a seed set of domain specific topics and terms from one or more reliable sources. The seed set of topics and terms can be queried from reliable Web sites, ontologies, or controlled vocabularies.

The consumer and patient health domain was selected for this study. The initial list of reliable Web sites 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]. This list was then reduced to those that allowed crawling. Each of the terms found within the allowable site's A-Z index glossary listing were extracted. On each of the glossary pages, the mining process was performed by saving the words 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.

The same term extraction process was performed on other health categorical listings available on these Web sites. For example, several 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. Cleaning included removing non-alphanumeric symbols and common stop words such as: *the*, *of*, and *and*. We also split the anchor text that included several related topics, such as *Acute Coronary Syndromes (Heart Attack; Myocardial Infarction; Unstable Angina)*, into separate topics: *acute coronary syndromes*, *heart attack*, *myocardial infarction*, and *unstable angina*. The final data cleaning step included the conversion of all words to lower case. After eliminating duplicate topics, the set of total unique health topics extracted from the seed Web sites totaled $11,679$.

C. Extracting Domain Specific Tags

In order to target the subset of tags relating to our specific domain available from a collaborative tagging Web site, the seed set of topics discovered in the previous steps were used as search tags. Delicious was chosen as the source folksonomy for this research. A query for each one of the $11,679$ health topics was performed by utilizing its data feed API[3]. 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.

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.

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 several tags. For example, the tag *Heart_Disease* was split into the tags *heart* and *disease*. During the analysis of the tags generated by users, we discovered that users also concatenated more than one term without a delimiter. For

[2]http://caphis.mlanet.org/

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

example, the tag *HeartDisease* consists of more than one word; however, we did not attempt to split tags of this form.

D. Creating the Concept Hierarchy

After the tags have been retrieved and cleaned, the final step was to create a domain specific concept hierarchy. Discovering any frequent tag sequences is an important procedure within this framework step. 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 frequent tag sequences and edges are created between any two nodes, if there exist a topic/sub-topic relationship. The subsumption relationship described in [6] is used to establish an association from more general nodes to more inclusive nodes. The next two sections describe in detail the algorithms developed to discover frequent tag sequences and construct the concept hierarchy.

III. Frequent Tag Sequences

Mining sequential patterns within the folksonomy annotation records allows domain concepts spanning more than one tag to be identified. For example, the health topics of *heart attack* and *acute coronary syndrome* span more than one term and the sequential order of these terms is semantically important. Therefore, discovering sequences of varying lengths allows the relationships of topics, instead of simply single tag-to-tag relationships, to be investigated. In this section, we provide formal definitions to clarify the problem of discovering sequences of tags frequently found in folksonomy annotation records. We then discuss in detail the process of discovering these sequences.

A. Definitions

We define the data within a folksonomy in much of the same manner as it is defined in [2], [3], [5].

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 *date-times*, respectively.
- A is a set of annotation records. An item $a \in A$ has the following format: $(u, r, \{t_1, ..., t_n\}, d)$, representing 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 identifier. Users provide tags at their own discretion to annotate a resource. A tag can be any type of textual string whether the string consists of a word, a term, an acronym, 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 (e.g. Delicious), tag an image (e.g. Flickr), or tag a purchasable item (e.g. Amazon[4]).

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 [17]. 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} (e.g. $t_i =$ *acute*, $t_{i+1} =$ *coronary*, $t_{i+2} =$ *syndrome*) appear in the order specified by the user. Therefore, it is essential to consider the notion of a *text sequence* explained in [7] as well as the concept of an *item sequence* discussed [8].

A *tag sequence* is a consecutive 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 an n-sequence. A *tagset* of size n is synonymous with an n-sequence. Table I displays example tag sequences along with a unique tagset identifier, *tid*.

Within a database DB, the *support* for a tag sequence s, indicated by $\sigma_{DB}(s)$, is defined as the fraction of the total tagsets in DB containing the tag sequence. A tag sequence is said to be *frequent* if its support is greater than a minimum support threshold $0 \geq \sigma \geq 1$. A frequent n-sequence, is a sequence of length n that is frequent.

With respect to the length of the sequence, the differentiation between a constant support measure and a decreasing support measure are discussed in [16]. When mining sequences, a constant support measure will use the same σ to determine if a sequence is frequent, whereas with a length-decreasing support measure it is possible for the support measure to decrease as the length of the sequence increases. We will formally define the length-decreasing support measure below as well as its inverse and discuss how these functions are used in future sections.

From [16], given a database DB and a monotonically decreasing function f, such that $\forall l \in \mathbb{Z}^+ : 1 \geq f(l) \geq f(l+1) \geq 0$, a sequence s is frequent when measured with a *length-increasing, support-decreasing support constraint* iff $\sigma_{DB}(s) \geq f(|s|)$.

From [16], given a length-increasing, support-decreasing function f, the inverse of f, f^{-1}, is defined as $f^{-1}(\sigma) = \min(\{l | f(l) \leq \sigma\})$.

The inverse function, f^{-1}, measures the minimum length the sequence must be to become frequent. Because it is possible for an infrequent sequence to become frequent, [16] also introduced the *smallest valid extension* (SVE) property. The SVE property states that if a sequence s is currently infrequent, $\sigma_{DB}(s) < f(|s|)$, then for a sequence $s' \supset s$, $f^{-1}(\sigma_{DB}(s))$ is the minimum $|s'|$ such that s' can become frequent.

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

tid	sequence
1	⟨ heart, attack, acute, coronary, syndrome ⟩
2	⟨ acute, coronary, syndrome, diagnosis ⟩
3	⟨ heart, attack, symptoms ⟩

Table I: Example tag sequences

B. Tag Sequence Suffix Tree

The construction of a tag sequence suffix tree is an essential aspect in finding frequent tag sequences. The tag sequence suffix tree is a modified *generalized* suffix tree constructed using the naive algorithm described in [18]. Excluding the root node, each node stores additional information to aid in determining frequent tag sequences. Along with the current tag, t_i, each node stores the set of tagset ids, $tids$. A tagset id, tid, is added to the $tids$ set during construction of the suffix tree, if the tid does not already exist in the set. Therefore, $|tids|$ indicates the number of unique tagsets containing the sub-sequence. The second additional item stored in each node is l, the length of the longest path from this current node to a descendant leaf node. This value is utilized when searching the tree to find frequent sub-sequences. Our search algorithm for finding frequent sub-sequences will be able to disregard all outgoing paths from a node in the tree based on analyzing this value. Figure 2 provides a graphical representation of the additional information stored at each node.

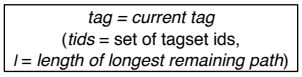

Figure 2: Data at each node of tag sequence suffix tree

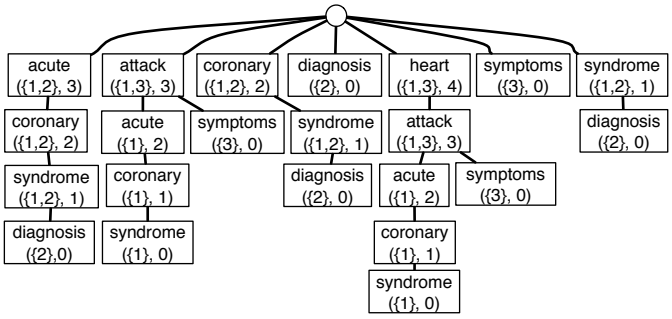

Figure 3: Tag sequence suffix tree from data in Table I

1) Suffix Tree and Frequent Sequence Algorithms: **generate-suffix-tree**. The `generate-suffix-tree` function creates the tag sequence suffix tree nodes and edges based on the tag sequence records stored in ΔDB. The function takes as two parameters: $root$ and ΔDB. $root$ is a pointer to the root node of the tag sequence suffix tree. ΔDB is a set of tag sequence records. ΔDB can be a full set of tag sequence database records or it can be an incremental set of records to add to the suffix tree. `generate-suffix-tree` loops through each record $(tid, \langle t_1, ..., t_m \rangle)$ in ΔDB adding each tag sequence suffix of this record to the tree. The function will traverse the tree, adding nodes when appropriate, and updating each node with the information tied to this record. Each node in the tag suffix tree contains tree pieces of information: (1) tag - the current tag in the suffix; (2) $tids$ - the set of tagset ids containing the sub-sequence to this node; and (3) l - the length longest remaining path in the tree originating from this node. Figure 3 displays the suffix tree generated from example data found from the example tag sequences in Table I.

find-frequent. The `find-frequent` function discovers the frequent tag sequences within the tag sequence suffix tree. It takes as parameters $node$ and $path$. $node$ is the current node in the tag suffix tree. $path$ is the sequence of tags created by traversing the tree to get to the current node. The function will perform a depth-first search on the tree determining whether the node is frequent by calculating the support value (the length-increasing, support-decreasing constraint measure) using the function f. `find-frequent` also utilizes the inverse function, f^{-1}, to determine if any of the child nodes should be visited. Therefore, if the current node is frequent or if there is a path long enough from this node that could become frequent then the child nodes must be visited. If the current node is neither frequent nor there is a path long enough for a sequence to become frequent then `find-frequent` will discontinue the depth-first search on this path. If a frequent node is found then the algorithm adds a tuple $(s', tids)$ to the set of frequent sequences F. The tuple $(s', tids)$ contains s', the path within the suffix tree to this current node, and $tids$, the tagset ids contained at this current, frequent node.

IV. Domain Specific 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 [6]. Using conditional probabilities, the concept hierarchy algorithm creates a directed edge between nodes n_i and n_j, denoted $n_i \longrightarrow n_j$, if the conditional probabilities satisfy the following criteria, $\mathrm{P}(n_i|n_j) > \theta$ and $\mathrm{P}(n_i|n_j) > \mathrm{P}(n_j|n_i)$, where θ is the subsumption threshold.

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 ⟨'acute', 'coronary', 'syndrome'⟩ then the label for this node is 'acute coronary syndrome.'

A. Algorithm

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

generate-nodes. The `generate-nodes` function takes as a parameter, F, the set of frequent tag sequences discovered in the `find-frequent` function. 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. The nodes generated in this function are considered *candidates* because it is possible one or more nodes may not appear in the hierarchy, if they are not connected to another node through an edge.

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 `find-frequent` function. N is the set of candidate nodes generated in `generate-nodes`. θ is the conditional probability used to create an edge. 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 designated as the *source* of the edge and n_j is designated as the *target* of the edge.

The two conditional probability calculations of nodes n_i and n_j, P$(n_i|n_j)$ and P$(n_j|n_i)$, are calculated using the tagsets associated with each node, which have been provided by the set F. Each element in F contains the tuple $(s', tids)$, where s' is the frequent tag sequence and $tids$ is the set of tagset ids where this tag sequence appears.

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 V display this virtual node.

V. Results and Discussion

During this study we queried Delicious several times using the seed set of $11,679$ health topics originally extracted from the health Web sites. From the tag sequences downloaded we created the tag sequence suffix tree for the annotation records created before April 23, 2009. `find-frequent`, `generate-nodes`, and `generate-edges` functions were run using this subset of downloaded annotation records. This subset of records, totaling $154,308$ annotation records, is denoted as db in Figure 4. Next, the annotation records created on or after April 23, 2009 were added to the tag sequence suffix tree. There were a total of $8,820$ records within this additional set, bringing the total set of annotation records to $163,128$. This set of records is denoted as DB, as seen in Figure 5. This date was chosen based on approximately when the Centers for Disease Control (CDC) added the H1N1 swine flu information to their Web site[5]. As it can be seen additional topics revolving around *swine flu* and *H1N1* became frequent. It should be noted that within the small set of $8,820$ records there were a large set of flu-related concepts that appeared.

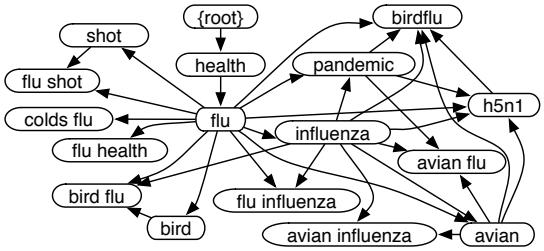

Figure 4: Flu concept hierarchy for data up to April 23, 2009 (db), using conditional probability $\theta = 0.50$

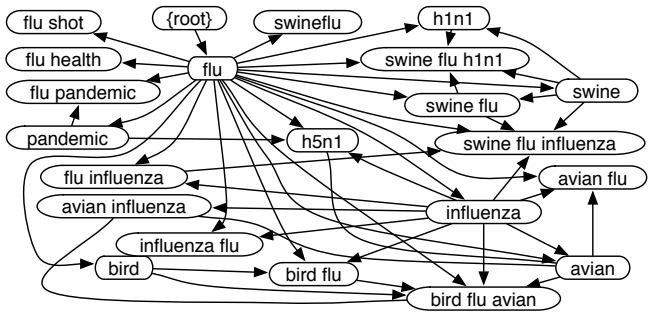

Figure 5: Flu concept hierarchy including data on or after April 23, 2009 (DB), using conditional probability $\theta = 0.50$.

To discover frequent tag sequences with the decreasing support constraint, we tried a variety of decreasing support functions, for the graphs that are produced in this results section, we used the function, $f(1) = 0.001, f(2) = 0.0005, f(i) = 0.00025$ for all $i > 2$.

When executing the `generate-edges` function, we used $\theta = 0.50$. For two nodes, n_i and n_j, we stored the following tuple (n_i, n_j), if P$(n_i|n_j) > \theta$ and P$(n_i|n_j) >$ P$(n_j|n_i)$. The storage of these records and the usage of a low θ value was done so we can easily compare the concept hierarchies created for different conditional probability values.

The interconnectedness between several topics can be seen within the graphs. 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, just to name a few. This methodology can be generalized to track the progression user-generated annotation tags for domain specific concepts.

[5]http://www.cdc.gov/H1N1flu/

VI. Conclusion

In this paper we presented a means for creating a domain specific concept hierarchy from user generated taxonomic information. A multi-step approach was employed to extract domain specific tag sequences from a popular folksonomy. A tag sequence suffix tree was then loaded for the discovery of frequent sub-sequences. By applying a monotonically decreasing support constraint on length-increasing tag sequences, a higher support threshold for shorter sequences and a lower support threshold for longer sequences could be used in the discovery of interesting sequences at a variety of lengths.

Once the frequent sequences were discovered, a concept hierarchy was created using conditional probabilities. Directed edges were created within the concept hierarchy, if a subsumption property was met. 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.

VII. Future Work

In the future we plan to extend this work in the following directions. It is possible to expand our methodology to find the first set of tag sequences and URLs from a set of topics and terms and then find the additional tag sequences tied to newly retrieved URLs. Additionally, we plan to detect and remove tagsets highly regarded as spam. It has been discovered through this research 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.

References

[1] T. Eda, M. Yoshikawa, and M. Yamamuro, "Locally Expandable Allocation of Folksonomy Tags in a Directed Acyclic Graph," in *Proc. of the 9th Intl. Conf. on Web Information Systems Engineering (WISE '08)*, 2008, pp. 151–162.

[2] P. Heymann and H. Garcia-Molina, "Collaborative Creation of Communal Hierarchical Taxonomies in Social Tagging Systems," Stanford University, Tech. Rep. InfoLab Technical Report 2006-10, 2006.

[3] C. Schmitz, A. Hotho, R. Jäschke, and G. Stumme, "Mining Association Rules in Folksonomies," in *Data Science and Classification*, 2006, pp. 261–270.

[4] P. Schmitz, "Inducing ontology from flickr tags," in *Proc. of the Collaborative Web Tagging Workshop (WWW '06)*, 2006.

[5] A. Shepitsen, J. Gemmell, B. Mobasher, and R. Burke, "Personalized recommendation in social tagging systems using hierarchical clustering," in *Proc. of the 2008 ACM conference on Recommender systems*, 2008, pp. 259–266.

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

[7] H. Ahonen-Myka, "Finding All Frequent Maximal Sequences in Text," in *16th Intl. Conf. on Machine Learning (ICML-99) Workshop on Machine Learning in Text Data Analysis*, D. Mladenic and M. Grobelnik, Eds., 1999, pp. 11–17.

[8] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proc. of the 1995 IEEE Intl. Conf. on Data Eng.*, 1995, pp. 3–14.

[9] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," in *Proc. 5th Int. Conf. Extending Database Technology (EDBT)*, 1996, pp. 3–17.

[10] S. Parthasarathy, M. J. Zaki, M. Ogihara, and S. Dwarkadas, "Incremental and Interactive Sequence Mining," in *Proc. of Eighth Intl. Conf. on Information and Knowledge Management (CIKM '99)*, 1999, pp. 251–258.

[11] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation:a frequent-pattern tree approach," in *Proc. of the 2000 ACM SIGMOD Intl. Conf. on Management of Data (SIGMOD '00)*, Dallas, TX, 2000, pp. 1–12.

[12] M. J. Zaki, "Spade: An efficient algorithm for mining frequent sequences," *Machine Learning*, vol. 42, no. 1-2, pp. 31–60, 2001.

[13] M. Zhang, B. Kao, D. Cheung, and C. lap Yip, "Efficient algorithms for incremental update of frequent sequences," in *Proc. of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2002)*, 2002, pp. 186–197.

[14] H. Cheng and J. Han, "IncSpan: Incremental Mining of Sequential Patterns in Large Database," in *Proc. of Intl. Conf. on Knowledge Discovery in Databases (KDD '04)*. ACM Press, 2004, pp. 527–532.

[15] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M. chun Hsu, "Mining sequential patterns by pattern-growth: The prefixspan approach," *IEEE Trans. Knowl. Data Eng.*, vol. 16, p. 2004, 2004.

[16] M. Seno and G. Karypis, "Finding frequent patterns using length-decreasing support constraints," *Data Mining and Knowledge Discovery*, vol. 10, no. 3, pp. 197 – 228, 2005.

[17] R. Agrawal, T. Imielinski, and A. N. Swami, "Mining Association Rules between Sets of Items in Large Databases," in *Proc. of the 1993 ACM SIGMOD Intl. Conf. on Management of Data (SIGMOD '93)*, P. Buneman and S. Jajodia, Eds., Washington, D.C., 1993, pp. 207–216.

[18] D. Gusfield, *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, 1997.