SOCIALLY INDUCED SEMANTIC NETWORKS AND APPLICATIONS

Benjamin C. Markines

Submitted to the faculty of the Graduate School

in partial fulfillment of the requirements

for the degree

Doctor of Philosophy

in the School of Informatics, Department of Computer Science

Indiana University

May 2009

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements of the degree of Doctor of Philosophy.

Doctoral Committee

Filippo Menczer, Ph.D.
(Principal Advisor)

Katy Börner, Ph.D.

Randall Bramley, Ph.D.

May 5, 2009

Dennis Groth, Ph.D.

This work is dedicated to my loving wife Kasey Markines and my unborn daughter. Thank you Kasey for your love and support. Also, I dedicate my thesis to my parents, Edmund and Samarngsri Markines, and my older brother, Leonid Markines, for believing in me. Finally, I dedicate this work to Kasey's parents, Michael and Kathie Rager, and her sister Mindy Sadowski. Hopefully, I have made them all proud.

Acknowledgements

A very special thanks to my advisor Filippo Menczer for seeing the potential in me. His help and guidance has taken me places that I did not think were possible. Thanks Fil for the conversations, suggestions, and words of encouragement. Also, thanks Fil for taking me with you to Torino, Italy on your sabbatical. In my opinion, my finest work was produced there. Thanks to the Networks and agents Network (NaN) in the School of Informatics at Indiana University for helpful discussions and comments. Thanks to Randall Bramley for his discussions on efficiency, Lubomira Stoilova for spending so much time on the early development of `GiveALink.org`, Todd Holloway for his work on the recommendation engine, Anna Maguitman and Heather Roinestad for providing the semantic similarity network for grounding our resource-to-resource measures, Luis Rocha for discussions leading to our novelty measure, Katy Börner for discussions regarding the visualization component of our navigation interface, Dennis Groth for general discussions about the navigation interface, Justin Donaldson for implementing most of the navigation interface, and Nick Street for suggestions on the feature analysis for spam detection. Thanks to Ruj Akavipat for the feature distribution analysis suggestion, Mark Meiss for his efficient implementation of Kendall's τ, and Jacob Ratkiewicz for helping me with early drafts of my thesis. Thanks to Sidney Stamm for providing me with the latex style sheets to generate this document. Thanks to Jeremy Engle for your feedback on various parts of this work.

Thanks to all the publication co-authors (geniuses) that I had the opportunity to work with: Filippo Menczer, Ciro Cattuto, Dominik Benz, Andreas Hotho, Gerd Stumme, Justin Donaldson, Michael Conover, Heather Roinestad, Lubomira Stoilova, Todd Holloway, Anna Maguitman, John Burgoon, Katy Börner, Weixia (Bonnie) Huang, and Bruce Herr.

Also, many thanks to the facilities group at the Department of Computer Science at Indiana University for their invaluable technical support. Rob Henderson and Shing-Shong (Bruce) Shei are quite possibly the best system administrators on the planet. Thanks to the administrators of the Big Red Cluster at Indiana University especially Stephen Simms and Yu (Marie) Ma. Thanks to the staff of the School of Informatics at Indiana University for helping me with travel. A special thanks to the European Commission's TAGora project (`tagora-project.eu`) for use of their servers and to BibSonomy for use of their data.

Thanks to IBM for granting me an educational leave of absence to complete my studies. Also, thanks for allowing me to return during the long summers. Thanks Richard Heffel for granting me the educational leave of absence, and Manmohanjit Singh for extending the educational leave.

I acknowledge generous support from the Department of Computer Science at Indiana University, ISI and CRT Foundations, the Center for Complex Networks and Systems Research (`cnets.indiana.edu`), NSF Grant Number IIS-0513650 (`nwb.slis.indiana.edu`), and NSF Grant Number IIS-0811994 (`cnets.indiana.edu/groups/nan/givealink`).

Abstract

Social bookmarking systems allow Web users to actively annotate online resources. These annotations incorporate meta-information with Web pages in addition to the actual document contents. From a collection of socially annotated resources, we present various methods for quantifying the relationship between objects, i.e., tags or resources. These relationships can then be represented in a semantic similarity network where the nodes represent objects and the undirected weighted edges represent their relations. These relations are quantified through similarity measures. There are two challenges associated with assembling and maintaining such a similarity network. The first challenge is updating the relations efficiently, i.e., the time and space complexity associated with graph algorithms. The complexity of these algorithms is typically quadratic. We present an incremental process answering both space and time limitations. The second challenge is the quality of the similarity measure. We evaluate various measures through the approximation of reference similarities. We then present a number of applications leveraging socially induced semantic similarity networks. A tag recommendation system, a page recommendation engine, and a query result interface are evaluated through user studies. Finally, we design spam detection algorithms to enhance the functionality of social bookmarking systems.

Contents

List of Tables

List of Figures

1

Abstract

The links and relationships between online resources were once controlled by editors and authors alone. Today there are online tools, such as Delicious and Wikipedia, that enable users to markup any online resource or to contribute content to any topic. Collaborative tagging systems provide users with a mechanism to freely annotate pages, media files, or other objects. These annotations can be mined to discover relationships among users, resources, and tags. This research focuses on assembling semantic similarity networks from user annotations. These similarity networks can in turn lead to improved recommendation, Web search interfaces, and spam detection. We will present the development of these types of applications leveraging socially induced semantic networks. Our group developed a playground to prototype these applications at `GiveALink.org`.

We explore two user annotation representations for the purpose of assembling socially induced semantic networks. One is the organization of Web links in a personal hierarchy. Users employ a hierarchical representation when bookmarking favorite Web sites in their browsers. We introduce an entropy based similarity measure that exploits bookmark hierarchies for uncovering relationships among Web resources. An accompanying user study demonstrates that the resulting network captures meaningful relationships. Another representation is the free-style tagging of resources with keywords. We study several similarity measures based on this "folksonomy" representation to extract semantic relationships

among tags and resources. To address limitations in traditional evaluation techniques, we introduce a framework in which our similarity measures are grounded against reference datasets. This work has led to the design of a novel information-theoretic similarity measure outperforming all other measures in the literature for both hierarchical and flat tagging representations, and for both tags and resources.

As online collaborative systems grow and evolve, it is important that semantic relationships be captured in an efficient way. We introduce several methods to aggregate social annotations into semantic similarity networks. We achieve scalability by an incremental aggregation algorithm that updates the network in real time. We explore the tradeoff between efficiency and effectiveness, reaching an optimal balance by integrating a collaborative filtering component into our incremental approach. The incremental approach has been deployed in our GiveALink platform allowing continuous real-world testing with an ever growing user base.

We explore the design of novel interfaces for improving Web navigation. My idea is to visualize socially induced semantic links to expand the user's contextual view of a Web page. These connections are different from the embedded hyperlinks created by authors. We also explore the visualization of the relationships between the links within a page, for example the results from a Web search. Contemporary search engines return pages containing ranked lists. We found that users were able to find relevant information with fewer queries by extending the list view with our network visualization.

With the growing popularity of social tagging systems, "social spam" detection is becoming an important problem. My work identifies a number of features that are highly predictive of spam using a dataset from `BibSonomy.org`, a popular social bookmarking system. One feature examines the focus of the tags used to annotate a resource. Tags that are semantically dissimilar from one another are more likely to signal spam. Another feature is plagiarized content. Pages with content taken from another site, such as Wikipedia,

are more inclined to be from spammers. Other suspicious features include ads, broken links, and automatically generated HTML. Combining these features lead to a spam detection accuracy above 98%.

2

Introduction

We are transitioning from the "Web 1.0," where information consumers and providers are clearly distinct, to the so-called "Web 2.0" in which anyone can easily annotate objects (sites, pages, media, and so on) that someone else authored. These annotations take many forms such as classification, voting, editing, and rating. Social bookmarking tools [HHLS05] allow Web users to annotate online resources in a central online location. There are two methods of user annotation that will be discussed. The first method is the hierarchical organization of one's Web bookmarks. This process requires the users to organize their online resources in a strict hierarchical taxonomy. The presentation and management of these personal taxonomies are standard functions of all mainstream Web browsers. For many Web users, this is still the main mechanism for maintaining a personalized collection of online resources. The management systems in browsers have the added benefit of personal privacy, but requires additional steps for sharing with others. Some popular examples include the *Favorites* in Internet Explorer and the *Bookmarks* in Firefox. The second method is the tripartite organization of one's Web links. A tripartite structure emerges from a user action known as social tagging [FFvA$^+$06]. Tagging is the application of freeform labels to online links and resources. Tagging serves two main functions. First, it allows users to arrange their resources as they see fit for recalling in the future – in other words, bookmarking. Second, tagging allows the user to participate in a community by

sharing personal resources and tags with others. There are numerous Web sites facilitating this function, the most popular being `delicious.com`. Recently, Firefox has enabled users to tag online resources directly in the browser for future recall. This functionality in Firefox does not include the ability to directly share with others.

Either through tagging or categorizing in a hierarchy, Web users are establishing a relationship between Web resources. We see leveraging these links as a second major transition in the brief history of the Web. The first one occurred when researchers went beyond the textual analysis of content by taking into account the hyperlinks created by authors as implicit endorsements between pages, leading to effective ranking and clustering algorithms such as PageRank [BP98] and HITS [Kle99]. Now social bookmarking grants us access to a more explicit and semantically rich source of social annotation. They allow us to extend the assessment of what a page is about from content analysis algorithms to the collective "wisdom of the crowd" [Sur04]. Figure 2.1 illustrates this phenomenon. If many people agree that a page is about programming then with high probability it is about programming even if its content does not include the word 'programming.' This bottom-up approach of collaborative content structuring is able to counteract some core deficiencies of traditional knowledge management, such as the knowledge acquisition bottleneck. This research focuses on capturing these semantics and representing these relationships in a network.

Whether one wants to represent relationships among users [DI06, Mik05], tags [BKS06], or resources [MSM06c, MSM06b, HJSS06b, CBSL07], large scale systems will require efficient techniques for building and maintaining these networks. Furthermore, these techniques can be used by researchers and developers to perform a deeper evaluation of proposed algorithms. In our discussion, we will follow two main approaches for building a semantic similarity network from individual profiles. The first approach will assemble a network based on a global aggregation of users. The second approach will build the

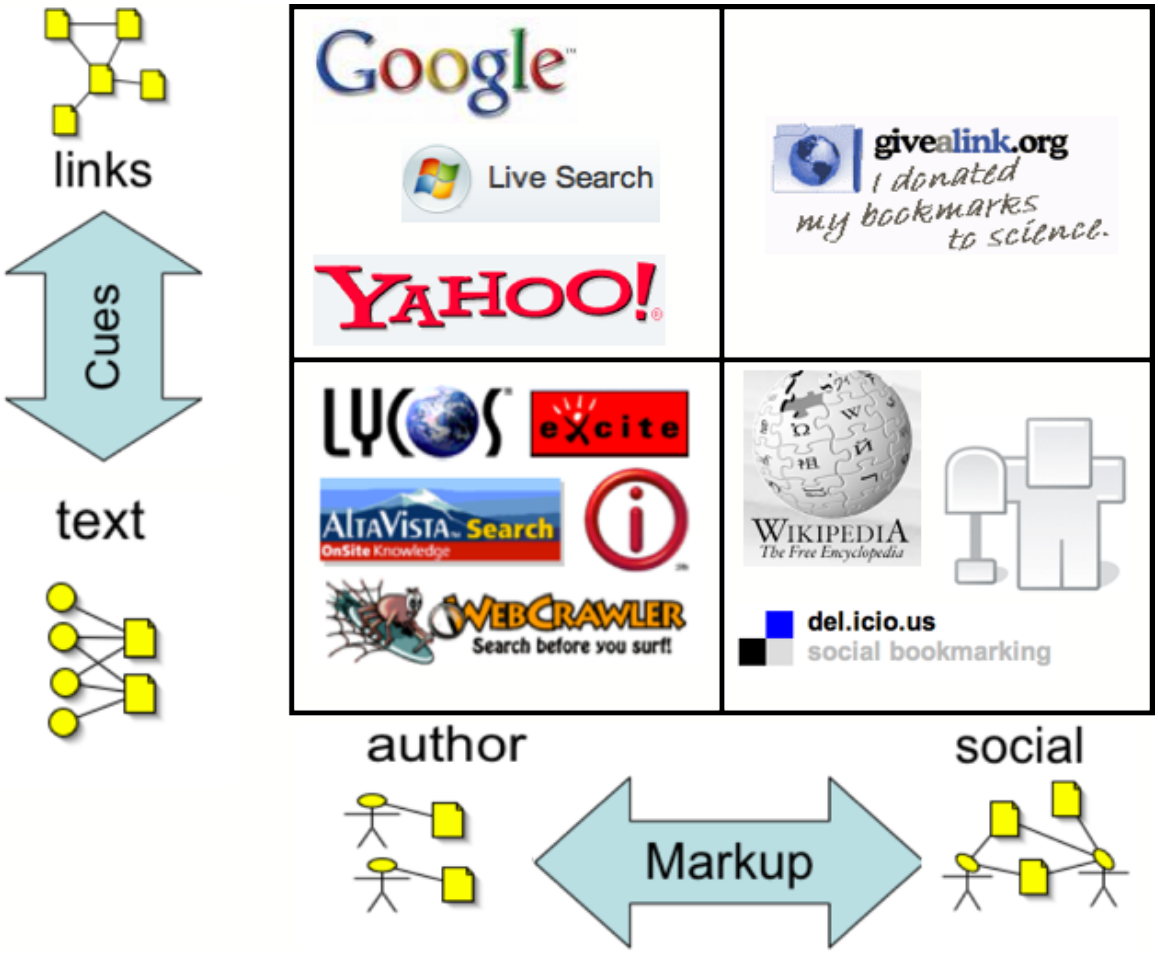


Figure 2.1: Different kinds of links between Web resources, and examples of applications leveraging such links. The vertical axis represents the cues used to represent pages (text alone in the first generation of search engines, and hyperlinks as well in the current generation). The horizontal axis illustrates the collaborative dimension where traditionally only the authors can determine the content and associations between pages, while in social systems any information consumer may contribute an opinion. The `GiveALink.org` system described here is an attempt to mine socially established semantic links between pages, users, or tags.

network incrementally as users participate in a social bookmarking system.

There are emerging applications exploiting socially induced semantic networks as illustrated in Figure 2.2. The semantic relationships between objects will continue to open the way to advancing online applications. Our discussion will introduce Web tools addressing link (resource) recommendation, tag recommendation, search engine interfaces, and spam detection. An obvious online tool that leverages a semantic network is recommendation: "given a link, provide related links" or "given a tag, provide related tags." Regarding search engine interfaces, we will explore a map of query results that visualizes the implicit links between online resources established by a community of users. Spam detection can also be enhanced by utilizing a tag by tag semantic similarity network.

In order to prototype and experiment with our ideas, we have developed the online social bookmarking tool `GiveALink.org`. At `GiveALink.org`, we assemble a socially induced semantic network. GiveALink will also be the site for deploying and experimenting with applications that leverage the semantic network.

Contributions and Outline

Leveraging semantic similarity networks induced from social media is the core idea in this research. The main contributions of this work are:

1. A framework for measuring semantic similarity between objects based on the annotations provided by Web users. (§ 4)

2. Techniques for assembling, storing, and maintaining a large socially induced semantic similarity network and runtime complexity analysis. (§ 5)

3. An evaluation of the effectiveness of various semantic similarity measures. (§ 6)

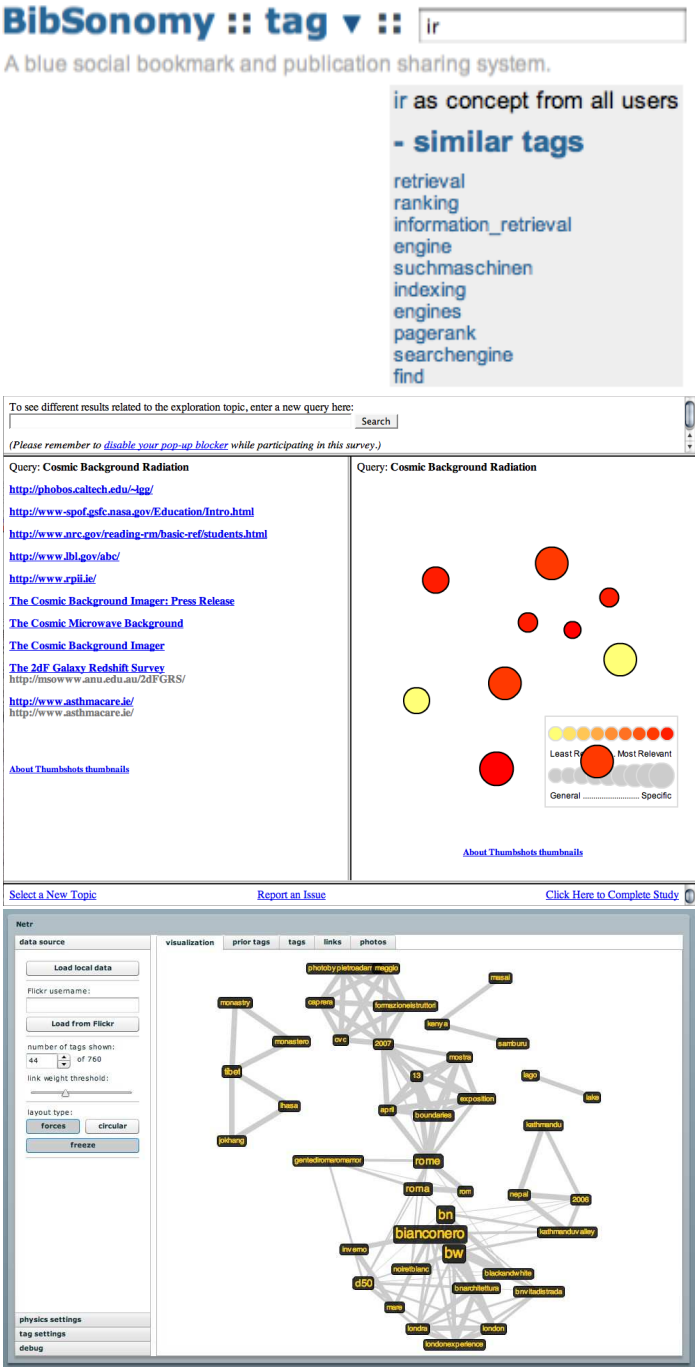


Figure 2.2: Three applications utilizing relationships between objects induced by social media. *Top:* Given a tag, the social bookmarking system `BibSonomy.org` suggests semantically similar tags. *Middle:* `GiveALink.org` leverages a similarity network of resources to support Web navigation [DCM+08]. *Bottom:* The online tool at `netr.it` visualizes tag relationships generated from a user's `Flickr` profile.

4. Novel applications leveraging a social semantic network that include tag recommendation, resource recommendation, an interface to map query results, and a system for detecting Web pollution and spam. (§ 7)

3

Background

3.1 Social Bookmarking

Social bookmarking is a way to manage bookmarks online, for easy access from multiple locations, and also to share them among a community. The most popular of these online tools `delicious.com` obtained instant popularity leading to being acquired by Yahoo in December 2005. The potential of social bookmarking has spurned interest for deployment in an enterprise [MFK06]. A number of social bookmarking tools are reviewed by Hammond et al. [HHLS05] and Lund et al. [LHFH05].

Although general search engines and directories are popular methods through which people find information, social bookmarking tools provide an interesting alternative. The classification schema they use are neither global (every user manages her own links) nor expert-based. In addition, traditional search and ranking algorithms like content similarity and PageRank empower information producers alone to affect the topic and importance of pages. Social bookmarking tools consider information consumers and become increasingly valuable as more people participate. Because of their user-centered design and quick feedback loop, they have become a major channel for knowledge propagation.

Many social bookmarking sites allow users to 'tag' (or associate words with) their URLs to describe their general topic. Users can learn about a topic either through visiting some

other user's bookmarks or by subscribing to a tag. Furthermore, some of these Websites provide services like RSS feeds and popularity rankings to facilitate the spreading of information. The emergent organization of information have become known as *folksonomies*.

While bookmarking and tagging may share several incentives [MNBD06], they are separate processes; many people bookmark without tagging [PP07]. The tagging approach has several limitations including lack of structure [BISI+06], lack of global coherence [PP07], polysemy, and word semantics [GH06]. Synonymy, the use of different languages, and spelling mistakes may force users to search through numerous tags. Navigation can be enhanced by suggesting tag relations grounded in content-based features [APH06].

Collaborative tagging is often contrasted with more traditional knowledge management approaches. Voss [Vos07] provides evidence that the difference between controlled and free indexing blurs with sufficient feedback mechanisms. The weaknesses and strengths of these different metadata mechanisms are compared by Christiaens [Chr06]. Schmitz et al. [SHJS06a] contrasted peer-to-peer knowledge management with tagging approaches.

As users of collaborative tagging systems continue to annotate links, their personal knowledge space will continue to grow. Users typically have difficulty recollecting past searches [AJK05]. Benz et al. [BTST06] has worked with organizing personal tag spaces, while Staff [Sta08] investigated assisting users with bookmark organization. Finally, researchers have directly addressed organization by presenting the foundation for cross-tool design across personal information collections [BS04]. The personal information collections in this research focuses on emails, files, and Web bookmarks.

Several studies and algorithms consider social annotations as a means of improving Web search. Examples include studies to compare the content of social networks with search engines [MD06, HKGM08] and enhancing search through the use of tags and rankings from social bookmarking systems [SHM+05, MSM06c, MSM06b, BXW+07]. Few have

addressed system efficiency for tag [SZL+08] or resource [MRM08] recommendation.

3.2 Semantic Similarity

Semantic similarity between Web sites describes the degree of relatedness between the Web sites, as it is perceived by human subjects. Measures of semantic similarity based on taxonomies (trees) and networks are well studied [GGMW03, Lin98, LKL93, RB89, RMBB89]. Maguitman [MMRV05] extended Lin's [Lin98] information-theoretic measure to infer similarity from the structure of general ontologies, both hierarchical and non-hierarchical. The Open Directory Project (ODP, `dmoz.org`) — a human-edited directory of the Web that classifies millions of pages into a topical ontology — can be used as a source of semantic similarity information between pairs of Web sites.

Search engine designers and researchers have made numerous attempts to automate the calculation of semantic similarity between Web pages through measures based on observable features, like content and hyperlinks. Studies conducted by Maguitman et al. [MMRV05] report, quite surprisingly, that measures relying heavily on content similarity (e.g. common words) are very poor predictors of semantic similarity. On the other hand, measures that only take into consideration link similarity (common forward and backward edges), or scale content similarity by link similarity, estimate semantic similarity with greater accuracy. Incidentally, neither content nor link similarity alone is a good approximation of semantic similarity, and they are also not strongly correlated with each other [Men05].

Measuring the semantic similarity among annotations/resources is an active research area. Mika [Mik05] provides a model of semantic-social networks for extracting lightweight ontologies from `delicious.com`. Tso-Sutter et al. [TSMST08] combined collaborative

filtering techniques for recommendation leveraging data crawled from `last.fm`. Cattuto et al. [CBSL07] use a variation of set overlap in the spirit of TF-IDF to build an adjacency matrix of Web resources. Wu et al. [WZM06] modify HITS to look for relationships between Web resources. Hotho et al. [HJSS06b] convert a folksonomy into an undirected weighted network, used for computing a modified PageRank algorithm called FolkRank for ranking query results. Semantic networks among tags have been built using co-occurrence [BKS06, SvZ08, LGZ08] and Jaccard's coefficient [HMHS06], also to reconstruct a hierarchy of concepts [HGM06]. Relationships among users can also be extracted from tagging systems. Diederich and Iofciu [DI06] and Marinho and Schmidt-Thieme [MST08] use a cosine variant to compute similarities between users. Others have proposed alternative approaches for extracting taxonomic relations or inferring global semantics from a folksonomy [Sch06, SHJS06b, HRS06, ZWY06]. In the context of social network analysis, Liben-Nowell and Kleinberg [LNK03] explore several notions of node similarity for link prediction.

Closely related to the task of measuring the relatedness of tags and resources is also the application domain of recommendations in folksonomies. The literature is still sparse. Existing work can be broadly divided into approaches that analyze the content of the tagged resources with information retrieval techniques [Mis06, BM06], approaches that use collaborative filtering methods based on the folksonomy structure [XFMS06, JMH+07], and combinations of the two [HRGM08]. Sarwar et al. [SKKR01] built networks using variations of cosine and correlation based similarity measures. Each type of network was exploited after assembly for investigating two collaborative filtering techniques.

3.3 Collaborative Filtering

Collaborative filtering, also referred to as social information filtering, identifies and exploits common patterns in the preferences of users. Traditionally, it has been used to identify communities and build recommendation systems based on like users' opinions. Examples include Ringo [SM95] for personalized music recommendations and GroupLens [RIS+94] for filtering streams of net news, as well as e-commerce sites such as `amazon.com` [SLY03] that make personalized product recommendations. These systems are predicated on the assumption that individuals who have shared tastes in the past will continue to share tastes in the future.

Similar work known as Fab introduced a content-based, collaborative recommendation system [BS97]. Fab was based on text content and relied on direct user feedback for a given Web page. When the user marks the page as interesting, words from the document are extracted and added to the user's profile. I-Spy uses collaborative filtering to enhance Web search results based on feedback from other community members who submit the same query [CKS04]. Users identify relevant search results, and pages picked more often have a higher chance to appear as future answers to a query.

Collaborative filtering techniques are also used for inferring global structures in information domains. PageRank can be seen as a prominent example of this approach. To a first-degree approximation, PageRank assumes that the number of citations or inlinks to a Web page is a testimony for its importance and quality [FBFM09]. Of course, the hyperlink structure of the Web has been created by individual users adding links from their pages to other pages. Thus the count of citations to a Web page is in essence a collaborative filtering measure. More recently, sites such as `stumbleupon.com` and `digg.com` receive direct feedback from their users. Participants have the choice to vote up or down any Web link. These online tools are special cases of collaborative filtering where all votes are aggregated

for recommendation.

Despite their success and popularity, collaborative filtering techniques suffer from some well-known limitations [SKKR00]. One is the sparsity of user profiles: the number of items contained in one user profile is negligible compared to the entire dataset (the Web in our case). Thus the information contributed by one user is small and it is hard to infer communities because there is little overlap between profiles. Another critical limitation is the complexity of collaborative filtering algorithms. The latency associated with processing large data sets requires that similarity information is pre-computed offline. When a user submits a new profile, the data is not integrated into the system until the next time the database is rebuilt. The user profile is not updated until then either. Finally, collaborative filtering systems cannot generate predictions about new items. Since similarity is estimated based on existing user profiles, the system cannot generate recommendations for URLs that the users are not already familiar with. Some of these limitations also apply to our discussion, although we describe ways to work around them.

3.4 Web Interfaces and Visualization

Contemporary Web information retrieval (IR) applications present results in an ordered list according to some measure of relevance to a query. This mode of interaction has been proven to be very useful for conventional information retrieval tasks involving the look up of answers to specific questions, or finding specific pieces of information. On the other hand, list interfaces suffer some limitations when it comes to exploratory search. Marchionini compares and contrast conventional 'lookup' search with 'learning' and 'investigative' search [Mar06]. Methods for learning and investigative search often involve discovering abstract relationships between result items that go beyond similarities of content. If we were able to visualize these relationships, then we will be providing the user with a broader view or context while exploring the Web. Here, we define context as an expanded view of other Web objects that are highly related.

If we consider the Web, current internet browsers fail to provide an effective overview of the currently viewed page. Furthermore, when users submit queries to search engines, they may expand their view by opening multiple windows and tabs linked directly from the result list. Multiple windows and tabs demonstrate limited context, but this still requires the user to individually track each open page increasing the cognitive load. Additionally once a user navigates to a page, the links from that page are typically under the control of the Web page's author. For example, it is unlikely that a news article on `cnn.com` will ever be linked to a similar article on `abcnews.com`. If we are able to identify these similar Web resources, we can visualize these pages on a two-dimensional map providing the user with a broader view of the Web. The metadata supplied by collaborative tagging systems may be the data necessary to provide context when viewing and navigating the Web.

Information visualization provides users one way of providing broader views of data.

Used effectively, information visualization can reveal certain properties, relationships, and provide context that may otherwise be difficult to realize with ordered lists. A survey of information visualization and applications are presented by Börner [BCB03]. This research focuses much of its attention to assembling semantic networks, therefore, we focus on network visualization. There are methods for arranging search results using various graph visualization and interaction approaches. Herman et al. [HMM00] and Kules et al. [KWSS08] provide good reviews of graph visualization techniques. These approaches have been proposed in a variety of different domains including music [RPM03], medical/healthcare information [Bou03], and general search [BMW97, CK97, THA99, HMC99, LA00, DNFY05]. In particular, Hu et al. [HMC99], and Allan, Leuski et al. [ALSB01, LA04] provide direct comparisons of the effectiveness of list and visualization based interfaces in goal oriented IR tasks showing that hybrid visualization/list approaches improve retrieval task performance by reducing user query reformulations. Hu et al. [HMC99] also reveal that modulating size and color of result nodes improves user performance

There has been work with visualizing the Web. Lagus applies the WEBSOM method, which is based on self-organizing maps, to the online Encyclopedia Britannica collection for building a semantic map [LKK04]. In her thesis, she proposes that a semantic map may enhance a user's experience when viewing query results from a search engine such as `google.com` [Lag00]. Multidimensional scaling (MDS) is another technique leveraged for visualizing Web resources. Walter used (MDS) to induce a map of Web pages drawn from the internet movie database (`imdb.org`) [Wal02]. These techniques visualize Web resources providing the user with broader context of specific knowledge spaces. These algorithms rely on the Web page's content, which is one major difference from our approach that depends on metadata (user annotations).

Tag clouds have become the most prevalent visualization tool for tags and labels in

Web 2.0 applications as well as the primary navigation tool in social bookmarking systems. Tag clouds are presentations of varying size, weight, and color to reflect a label's usage. Studies have been conducted to understand the most effective [RGMM07] and the most beneficial [KL07] tag clouds. The social influence of tag clouds has also been studied [HR08] showing evidence of a strong bias to prevalent ideas. Despite their widespread use, tag clouds have been shown to be insufficient as a primary means of Web navigation [SCH08].

3.5 Social Spam

Long before social applications became popular, spam was a problem in other domains, first in email then in Web search [GGMP04, AS08]. Unfortunately, the countermeasures that were developed for email and Web spam do not directly apply to social systems [HKGM07]. Recently, Caverlee et al. [CLW08] introduced a trust system for combating spam on social networking sites. Benevenuto [BRA+08] examined spam detection on the social media site `YouTube.com`. Social annotations sites that enable voting have also been a target for spammers [BLAZ08]. Koutrika et al. [KEG+07] conducted one of the first studies for spam detection in social bookmarking systems, the problem on which we focus in § 7.4.

The dataset used in § 7.4 has been the focus of other social spam detection efforts. Gkanogiannis and Kalamboukis [GK08] use a Rocchio-like method to maximize the discrimination between spammers and non-spammers using tags. Chevalier et al. [CG08] use features such as number of tags per post, length of tags, and number of tags with a special character for classification. Kim and Hwang [KH08] compute the mutual information between a tag and a user for classification with naïve Bayes. Gkanogiannis achieved the top performance of these methods with an area under the ROC curve of 0.98. A perfect area under the ROC curve is 1.0. Krause et al. [KSHS08] extract features such as special characters in the email or username, number of account requests from a location (IP), average number of tags per post, tag blacklists, and other semantic features for detection. None of the six social spam features analyzed in § 7.4 are found in prior literature.

3.6 Network Management Techniques

Measuring similarity between objects will be captured in a network where the nodes represent items and the undirected weighted edges symbolize the relationships (semantic similarity) among them. An information retrieval application utilizing a semantic similarity network will require intelligent management of the data. Specifically for our applications, we concentrate on efficiently assembling, updating, and retrieving from the network. There are two main approaches for managing large networks: directly with network based libraries and indirectly with matrix tool kits.

There are various graph libraries freely available. Researchers have produced libraries that specialize in aesthetically pleasing network visualizations [GN99, HCL05], while Adai [ADWM04] and Munzner [Mun00] concentrated on interaction as well. There is also work focusing on network analysis [MN99, Knu93]. A few libraries address both analysis and visualization [BM98, OFWB03]. High performance graph analysis is also an active research area [Lee99, SLL02]. Computational needs for analysis of large networks such as those in social, molecular, and biological domains can quickly grow both in space and time complexity. There exist parallel implementations addressing these complexity concerns [AJR+01, CDT03, GL05, DG04].

Representing a weighted undirected network as a matrix is trivial. If each row and column represents a unique item (i.e., a Web resource, tag, or user), then each element in a matrix captures the relationship between two nodes. Furthermore because the networks we propose are undirected, only storing half of the matrix is necessary. Developers and researchers have addressed various aspects of matrix management and manipulation. Some of the developed libraries directly take on performance, e.g., Meschach [SL94] and Newmat [Edd96]. Others have achieved performance while providing the end user with flexibility, e.g., Matrix Template Library [SL99], BLAS [LHKK79], and Blitz++ [Vel95]. In

the sparse matrix literature, Duff [DER86] introduce the concept of a vector register that addresses space complexity. This concept entails partitioning the matrix where each partition is handled by a vector register window. Still other researchers have presented techniques addressing both time and space complexity by introducing computing cluster based solutions [GS93, BMS97]. The most widespread library to utilize clusters is POOMA [CH97].

4

Social Similarity

There are three primary methods of representing Web resource annotations: hierarchical, tripartite, and binary. The hierarchical representation is induced by users organizing Web resources in a strict hierarchy either through browser bookmarks or in an online directory. The tripartite representation is induced by a user annotating a Web resource. Binary annotation in the form of voting approval or disapproval on Web resources has become mainstream with the growth and popularity of sites such as `digg.com`. We view binary as a special case of the tripartite mode where each user would essentially have only two types of markup per object. In the discussion below, mining relationships in the hierarchical representation will precede mining relationships in the tripartite representation.

4.1 Hierarchical Representation

Many users organize their Web links hierarchically. The most common interface is the browser's bookmark manager although there exist online systems for this purpose such as `mybookmarks.com` and `bookmarks.yahoo.com`. For this section, we will refer to *bookmarks* as those Web resources organized in hierarchical categories. Bookmarks are a convenient source of knowledge about the interests of Internet users. The files that a typical browser bookmark manager keeps contains some useful information. The following

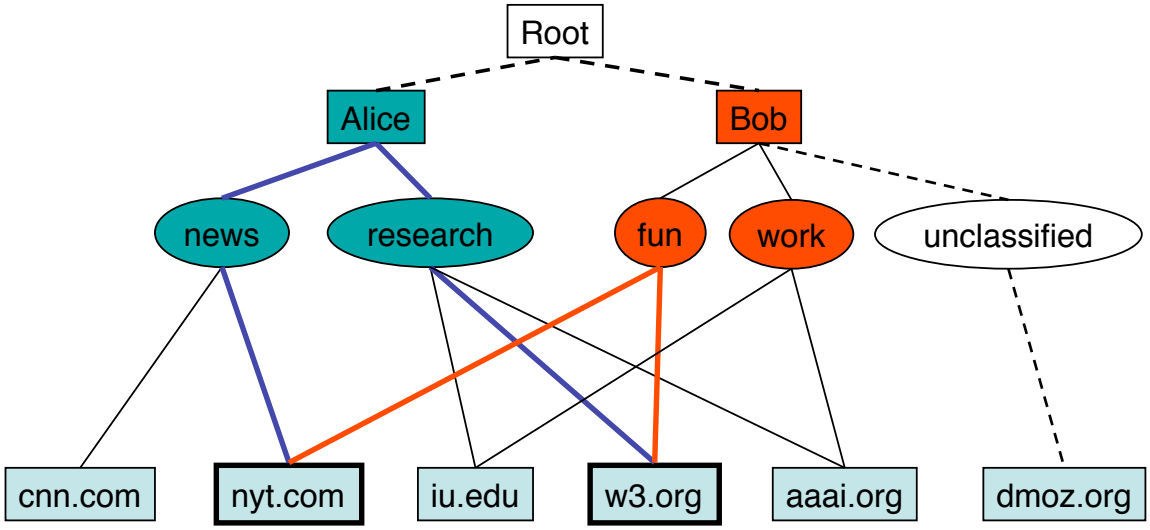


Figure 4.1: Combining bookmarks represented in hierarchies with collaborative filtering. Each user has a personal representation of the links that are important to them. GiveALink combines these representations for calculating semantic similarity. Here `iu.edu` and `aaai.org` will have a high similarity value because Alice and Bob organized them in close proximity.

attributes are explicit in the bookmark file: (1) URLs, (2) titles, (3) the hierarchical structure of the file, and (4) the browser and platform. Additionally, some provide the time when bookmarks are added and last accessed, as well as personalized title and description that users can edit themselves. We will now describe how we leverage a set of donated individual bookmark files to form a semantic similarity network.

4.1.1 Similarity Measure

We first present a method for extracting relationships among resources when the underlying structure is a tree [MSM06c, MSM06b]. Each bookmark file submitted by one user is viewed as a tree rooted at her username. Then we combine all of the user trees into a single tree by introducing a new root (super user) which is the parent of all user nodes. Figure 4.1 shows an example scenario in which only two users donated their bookmarks.

Lin

To exploit the structure of bookmark files, we use Lin's measure to calculate similarity between the URLs in a user u's tree [Lin98] . Let URL x be in folder F_x^u, URL y be in folder F_y^u, and the lowest common ancestor of x and y be folder $F_{a(x,y)}^u$. Also, let the size of any folder F, $|F|$ be the number of URLs in that folder and all of its subfolders. The size of the root folder is $|R|$. Then the similarity between x and y according to user u is:

$$s_u(x,y) = \frac{2 \times \log\left(\frac{|F_{a(x,y)}^u|}{|R|}\right)}{\log\frac{|F_x^u|}{|R|} + \log\frac{|F_y^u|}{|R|}}. \tag{4.1}$$

This function produces similarity values in $[0,1]$. For example, if two URLs appear in the same folder, their similarity is 1 because $F_x = F_y = F_{a(x,y)}$. Also, all other things being equal, the similarity between x and y is higher when F_y is a subfolder of F_x, than when F_x and F_y are siblings.

Many Web users do not organize their bookmarks in folders and subfolders and instead keep a flat list with their favorite links. If a user decides to leave some URLs in the root directory, we think of each URL as if it were in its own folder.

According to Equation 4.1, two URLs donated by different users have $s_u = 0$ because the lowest common ancestor is the root (super user). Thus Lin's measure is only appropriate for calculating the similarity of URL pairs according to a single user. To calculate the global similarity between URLs x and y, we sum the similarities reported by each user:

$$s(x,y) = \frac{1}{N}\sum_{u=1}^{N} s_u(x,y). \tag{4.2}$$

If a user has both URLs x and y, then he reports $s_u(x,y)$ according to Equation 4.1, otherwise he reports $s_u(x,y) = 0$. If a user has URL x in multiple locations, we calculate $s_u(x,y)$ for all locations of x and report the highest value. It is important to point out that here N is

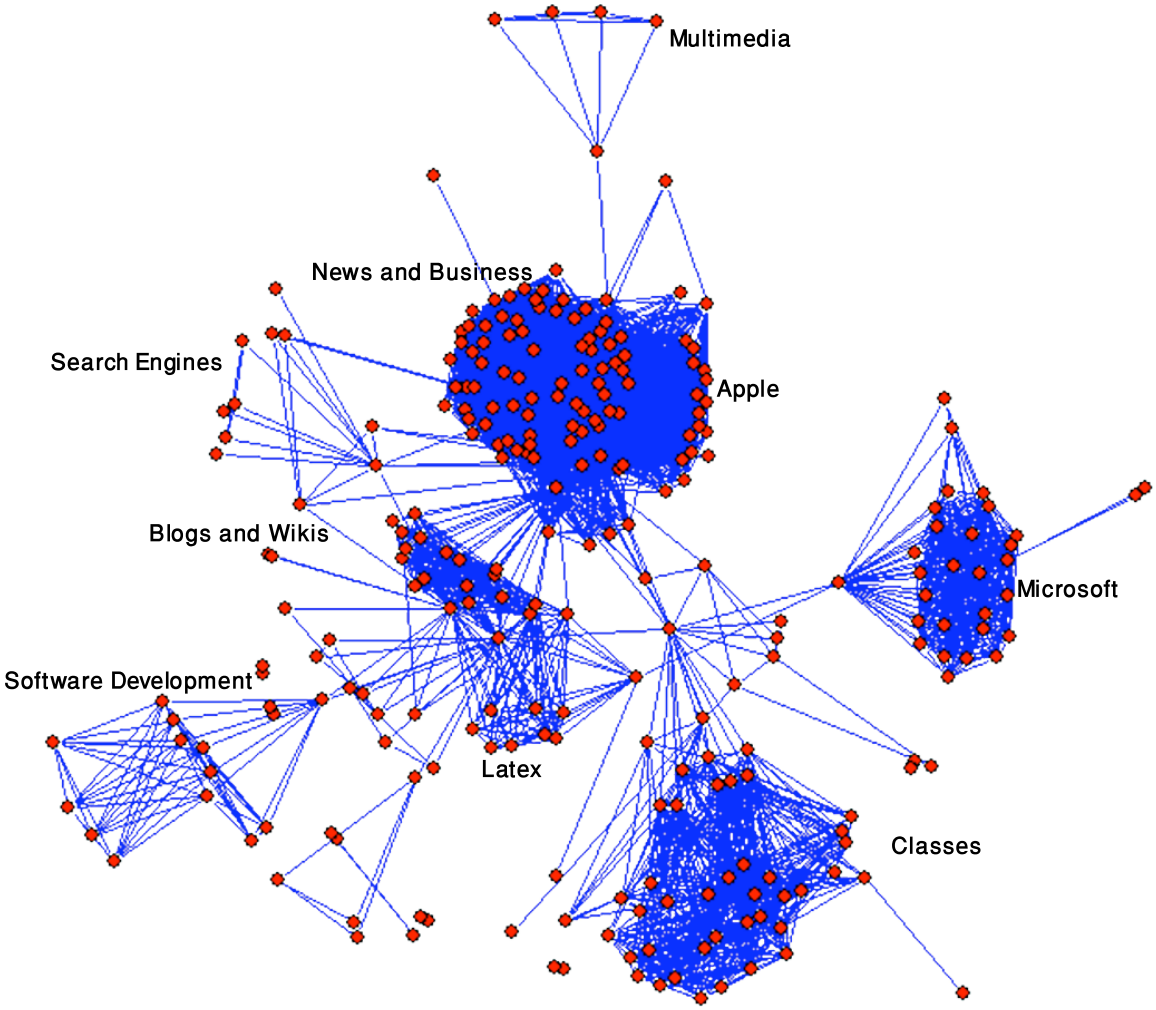


Figure 4.2: A visualization of the similarity network based on our modified Lin's measure. This illustration is generated using Pajek with the top 5,000 similarity scores ($s > 0.004$). Labels are added by hand to reflect cluster content.

the total number of users, not just those with $s_u(x, y) \neq 0$. Thus the more users share x and y, the higher $s_u(x, y)$. The final similarity matrix represents a weighted undirected graph where the nodes are URLs and the weight of an edge is the similarity of the two connected URLs. Figure 4.2 displays the GiveALink network in the early stages when a majority of users were from the Department of Computer Science at Indiana University. Note that for users whose bookmarks are unorganized (flat), the similarity measure reverts to standard

collaborative filtering [RIS+94]. We will refer to this similarity measure as *Lin*.

4.1.2 Incremental Similarity Measure

The similarity measure in Equation 4.1 relies on $|R|$, the size of the super tree that results from combining the users under a common root folder. The addition of the root folder makes each user's own contribution to the network dependent upon every other donation. As a result, each time a donation is added or modified, the entire network must be recomputed. If, however, the values each user contributes are independent of other donations, the network could adapt incrementally to reflect changes as they occur [MRM08].

Incremental Lin

To define this measure, we need to eliminate two dependencies on global parameters: $|R|$ in Equation 4.1, and N in Equation 4.2. With respect to the former, the superuser was originally introduced to make similarity values between any two bookmarks shared by a single user non-zero, thus capturing the notion of collaborative filtering. To preserve the collaborative filtering aspect of the similarity measure while removing the global dependency, we substitute $|R|$ in Equation 4.1 with the size of the user's bookmark tree $|u|$ plus a constant δ, yielding

$$s_u(x,y) = \frac{2 \times \log\left(\frac{|F^u_{a(x,y)}|}{|u|+\delta}\right)}{\log\frac{|F^u_x|}{|u|+\delta} + \log\frac{|F^u_y|}{|u|+\delta}}. \tag{4.3}$$

Here δ indicates that the user's donation is missing information available in other bookmarks. A larger value for the constant will approximate the superuser. A constant equal to 0 means the removal of the collaborative filtering aspect, allowing similarities of 0 even when the user has both resources. The question is how to set δ.

The second dependency is easily eliminated by removing the normalization factor N

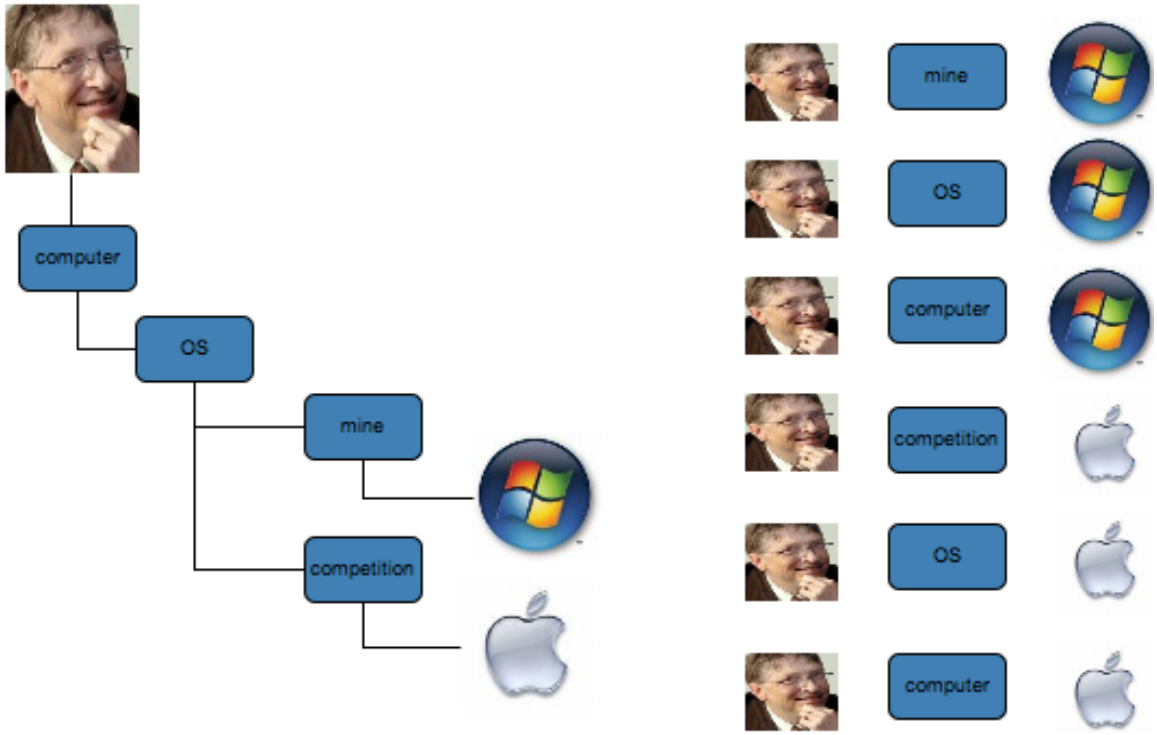


Figure 4.3: Conversion from hierarchical bookmarks to *triples*.

in the aggregation operation of Equation 4.2:

$$s(x, y) = \sum_{u=1}^{N} s_u(x, y). \tag{4.4}$$

We primarily use the similarities for ranking, therefore it is not necessary for the similarity values to be normalized between 0 and 1. Thus, we can store the sum s of the individual similarities, which can easily be updated in real time just for individual contributions from users with updated bookmarks. We will refer to this similarity measure as *incremental Lin*.

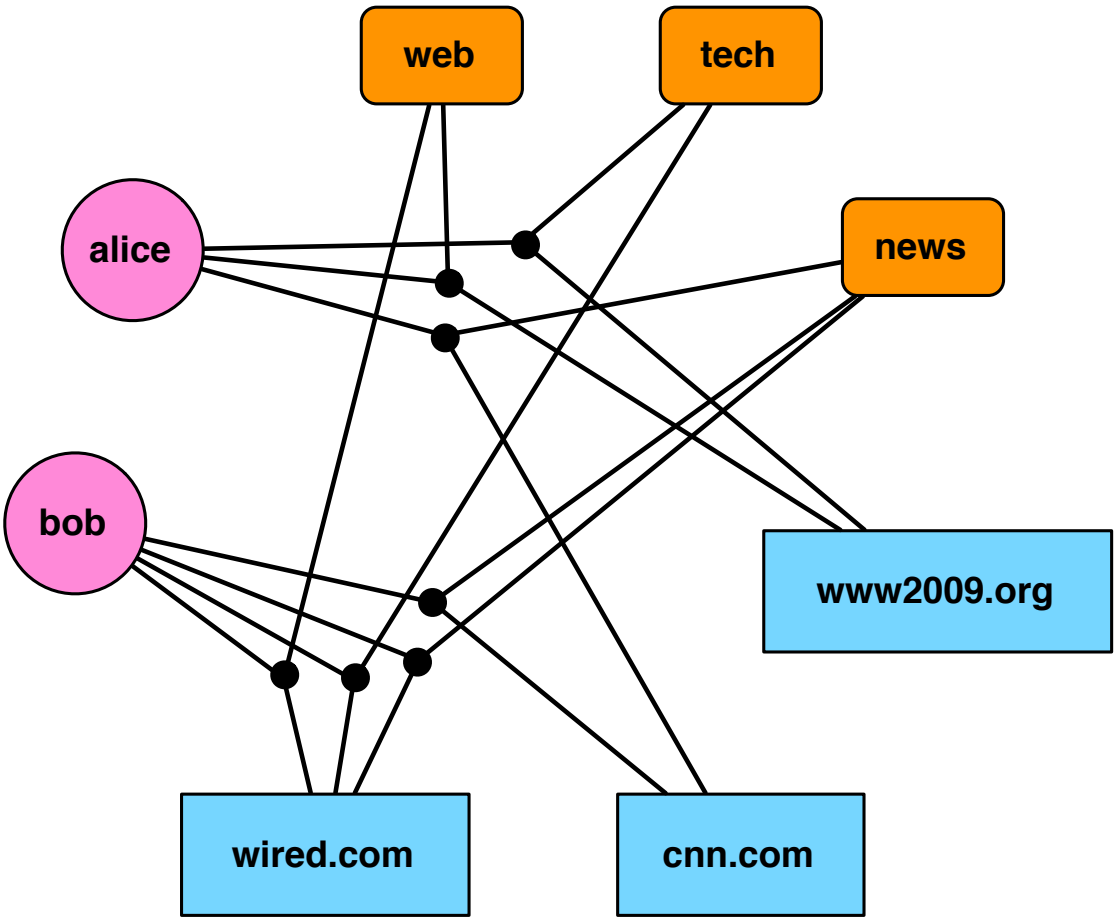


Figure 4.4: Example folksonomy. Two users (`alice` and `bob`) annotate three resources (`cnn.com`, `wsdm2009.org`, `wired.com`) using three tags (`news`, `web`, `tech`). The triples (u, r, t) are represented as hyper-edges connecting a user, a resource and a tag. The 7 triples correspond to the following 4 posts: (`alice`, `cnn.com`, {`news`}), (`alice`, `wsdm2009.org`, {`web`, `tech`}), (`bob`, `cnn.com`, {`news`}), (`bob`, `wired.com`, {`news`, `web`, `tech`}).

4.2 Tripartite Representation

The introduction of online collaborative tagging and voting induces a structure that is non-hierarchical. This structure involves users, resources (e.g., Web links), and annotations (e.g., tags and votes) and can be represented as a *tripartite* graph. A ternary relation between a user u, tag t, and resource r can be defined as a *triple*. The triple representation is widely adopted in the Semantic Web community [HJSS06b], which is closely related to the triadic context in formal concept analysis [LW95]. We then can establish a set of triples as a folksonomy F. A triple is a highly flexible representation for which efficient data store libraries exist. Folksonomies are readily represented via triples; a post $(u, r, (t_1, \ldots, t_n))$ is transformed into a set of triples $\{(u, r, t_1), \ldots, (u, r, t_n)\}$. Note that hierarchical classifications can also be represented by triples by equating categories (or folders) with tags and applying inheritance relationships in a straightforward way; a classification (u, r, t) implies $\{(u, r, t), (u, r, t_1), \ldots, (u, r, t_n)\}$ for all parent classes $t_i \supseteq t$. Therefore the triple representation subsumes hierarchical taxonomies and folksonomies. Figure 4.3 illustrates a conversion from a hierarchy to a set of triples. As an example, Figure 4.4 displays seven triples corresponding to a set of four posts by two users. In the following sections we use this running example to illustrate different definitions of similarity.

We will define similarity measures $\sigma(x, y)$ where x and y can be two resources (pages, media, etc.), tags (keywords, phrases, categories, etc.), and users. Since measures for similarity and relatedness are not well developed for three-mode data such as folksonomies, we consider various ways to obtain two-mode views of the data. In particular, we consider two-mode views in which the two dimensions considered are dual — for example, resources and tags can be dual views if resources are represented as sets of tags and vice-versa, or if tags are represented as vectors of resources and vice-versa. We focus on the development of information-theoretic similarity measures, which take into account the information/entropy associated with each item [MCM+09b].

4.2.1 Aggregation Methods

In reducing the dimensionality of the triple space, we necessarily lose correlation information. Therefore the aggregation method is critical for the design of effective similarity measures; poor aggregation choices may negatively affect the quality of the similarity by discarding informative correlations.

As mentioned above, we can define similarity measures for each of the three dimensions (users, resources, tags) by first aggregating across one of the other dimensions to obtain a two-mode view of the annotation information. Therefore we aggregate across users, and obtain dual views of resources and tags, yielding dual definitions for resource and tag similarity. To keep the notation a bit simpler, let us make explicit the dimension of users along which we aggregate, even though the discussion can be extended in a straightforward way to aggregate across tags or resources. Initially, we will consider two approaches to aggregate user information that depend on global frequencies, i.e., projection and distributional aggregation. Then, we will introduce two more aggregation methods free of any global dependencies, i.e., macro and collaborative aggregation.

4.2.1.1 Projection

The simplest aggregation approach is to project across users, obtaining a unique set of (r, t) pairs. If the triples are stored in a database relation F, this corresponds to the projection operator in relational algebra: $\pi_{r,t}(F)$. Another way to represent the result of aggregation by simple projection is a matrix with binary elements $w_{rt} \in \{0, 1\}$ where rows correspond to resources (as binary vectors, or sets of tags) and columns corresponds to tags (as binary vectors, or sets of resources). All similarity measures are then derived directly from this set information. As an example, the projected binary matrix for the folksonomy of Figure 4.4 is reported below. Given a resource and a tag, a 0 in the corresponding matrix element

means that no user associated that resource with that tag, whereas a 1 means that at least one user has performed the indicated association.

	news	web	tech
cnn.com	1	0	0
www2009.org	0	1	1
wired.com	1	1	1

4.2.1.2 Distributional

A more sophisticated form of aggregation stems from considering distributional information associated with the set membership relationships between resources and tags. One way to achieve distributional aggregation is to make set membership fuzzy, i.e., weighted by the Shannon information (log-odds) extracted from the annotations. Intuitively, a shared tag may signal a weak association if it is very common. For example, let r be the set of resources and r_x the resources annotated with x, we will use the information of tag (resp. resource) x defined as $-\log p(x)$ where

$$p(x) = \frac{|r_x|}{|r|}. \tag{4.5}$$

Another approach is to count the users who agree on a certain resource-tag annotation while projecting across users. This yields a set of *frequency-weighted* pairs (r, t, w_{rt}) where the weight w_{rt} is the number of users tagging r with t. Such a representation corresponds to a matrix with integer elements w_{rt}, where rows are resources vectors and columns are tag vectors. For the folksonomy of Figure 4.4, such a matrix is reported below. Similarity measures are then derived directly from the weighted representation.

	news	web	tech
cnn.com	2	0	0
www2009.org	0	1	1
wired.com	1	1	1

We will use both of the above distributional aggregation approaches, as appropriate for different similarity measures. The fuzzy set approach is appropriate when we want to perform row/column normalization of tag/resource probabilities to prevent very popular items from dominating the similarity. Other measures such as the dot product depend naturally on weighted vector representations.

4.2.1.3 Macro

By analogy to micro-averaging in text mining, distributional aggregation can be viewed as "micro-aggregation" if we think of users as classes. Each annotation is given the same weight, so that a more active user would have a larger impact on the weights and consequently on any derived similarity measure. In contrast, macro-aggregation treats each user's annotation set independently first, and then aggregates across users. This will allow the similarity calculation to be *incremental* breaking the dependency on global frequencies. In relational terms, we can select the triples involving each user u and then project, yielding a set of pairs for u: $\{(r,t)_u\} = \pi_{r,t}(\sigma_u(F))$. This results in per-user binary matrices of the form $w_{u,rt} \in \{0,1\}$. For the example folksonomy of Figure 4.4, we report below the matrices for the user alice (top) and bob (bottom).

	news	web	tech
cnn.com	1	0	0
www2009.org	0	1	1
wired.com	0	0	0

	news	web	tech
cnn.com	1	0	0
www2009.org	0	0	0
wired.com	1	1	1

The per-user binary matrix representations $w_{u,rt} \in \{0, 1\}$ are used to compute a "local" similarity $\sigma_u(x, y)$ for each pair of objects (resources or tags) x and y. Let us also define the probability of an object to be

$$p(x|u) = \frac{N(u, x)}{N(u)} \tag{4.6}$$

where $N(u, x)$ is the number of distinct attributes applied to object x according to user u, and $N(u)$ is the number of unique attributes used by user u. Finally, we macro-aggregate by voting, i.e., by summing across users to obtain the "global" similarity

$$\sigma(x, y) = \sum_u \sigma_u(x, y). \tag{4.7}$$

Macro-aggregation does not have a bias toward users with many annotations. However, in giving the same importance to each user, the derived similarity measures amplify the relative impact of annotations by less active users. It is an empirical question which of these biases is more effective.

4.2.1.4 Collaborative

Macro-aggregation lends itself to explore the issue of collaborative filtering in folksonomies while the computation remains incremental. Thus far, we have only considered feature-based representations when working with a tripartite representation. That is, a resource is described in terms of its tag features and vice-versa. If two objects share no feature, all of the measures defined on the basis of the aggregation schemes will yield a zero similarity.

In collaborative filtering, on the other hand, the fact that one or more users vote for (or in our case annotate) two objects is seen as implicit evidence of an association between the two objects. The more users share a pair of items, the stronger the association. We want to consider the same idea in the context of folksonomies. If many users annotate the same pair of resources, even with different tags, the two resources might be related. Likewise, if many users employ the same pair of tags, the two tags might be related even if they share no resources.

Macro-aggregation incorporates the same idea by virtue of summing user votes, if we assign a non-zero local similarity $\sigma_u(x, y) > 0$ to every pair of objects (x, y) present in u's annotations, irrespective of shared features. This is accomplished by adding a feature-independent local similarity to every pair (x, y) of resources or tags. In practice we can achieve this by adding a special "user tag" (resp. "user resource") to all resources (resp. tags) of u. This way all of u's items have at least one annotation in common.

Prior to macro-averaging, u's local similarity σ_u for each pair must be computed in such a way that the special annotations yield a small but non-zero contribution. This requires a revision of the information-theoretic similarity measures. For illustration, consider adding the special tag t_u^* to all resources annotated by u. The probability of observing tag t_u^* associated with any of u's resources is one, therefore the fact that two resources share t_u^* carries no information value (Shannon's information is $-\log p(t_u^*|u) = -\log 1 = 0$). Let us redefine user u's odds of tag (resp. resource) x as

$$p(x|u) = \frac{N(u, x)}{N(u) + \delta} \tag{4.8}$$

where $N(u, x)$ is the number of resources (resp. tags) annotated by u with x, $N(u)$ is the total number of resources (resp. tags) annotated by u, and δ is a constant. Notice that when

$\delta = 0$ the conditional probability reverts to Equation 4.6. This way,

$$-\log p(t_u^*|u) = -\log\left(\frac{N(u)}{N(u)+\delta}\right) > 0 \tag{4.9}$$

when $\delta > 0$. Similar to the δ in Equation 4.3, we can set δ to represent the missing information in other user profiles. Below we imply this construction in the definitions of the similarity measures with collaborative aggregation.

4.2.2 Similarity Measures

We wish to evaluate several information-theoretic, statistical, and practical similarity measures. Each of the aggregation methods requires revisions and extensions of the definitions for application to the folksonomy context, i.e., for computing resource and tag similarity from triple data.

Recalling that all measures are symmetric with respect to resources and tags, we simplify the notation as follows: x represents a tag or a resource and X is its vector representation. For example, if x is a resource, X is a tag vector with tag elements w_{xy}. If x is a tag, X is a resource vector with tag elements w_{xy} (note we do not switch the subscript order for generality). For projection aggregation, the binary vector X can be interpreted as a set and we write $y \in X$ to mean $w_{xy} = 1$ and $|X| = \sum_y w_{xy}$.

4.2.2.1 Matching

The *matching similarity* measure is defined, for the projection case, as

$$\sigma(x_1, x_2) = \sum_y w_{x_1 y} w_{x_2 y} = |X_1 \cap X_2|. \tag{4.10}$$

As an example, below we report the resulting similarity matrices for the resources and the tags of Figure 4.4:

	cnn.com	wsdm2009.org	wired.com
cnn.com	-	0	1
wsdm2009.org	0	-	2
wired.com	1	2	-

	news	web	tech
news	-	1	1
web	1	-	2
tech	1	2	-

The distributional version of the matching similarity is

$$\sigma(x_1, x_2) = - \sum_{y \in X_1 \cap X_2} \log p(y). \tag{4.11}$$

This and the other measures use the p definition of § 4.2.1.2. For the example case of Figure 4.4, the resources have the following probabilities:

$$p(\texttt{cnn.com}) = 1/3 \tag{4.12}$$

$$p(\texttt{wsdm2009.org}) = 2/3 \tag{4.13}$$

$$p(\texttt{wired.com}) = 1. \tag{4.14}$$

Here, out of 3 tags `cnn.com` is associated with 1 tag only, `news`. Also, notice that the probability of `wired.com` is 1 because the resource has been tagged by all available tags. The tag probabilities are as follows:

$$p(\texttt{news}) = 2/3 \tag{4.15}$$

$$p(\texttt{web}) = 2/3 \tag{4.16}$$

$$p(\texttt{tech}) = 2/3. \tag{4.17}$$

Here, out of 3 resources `news` is associated with 2 of them, `cnn.com` and `wired.com`. This yields the following similarity matrices for resources and tags (numeric values were truncated at the second decimal place):

	`cnn.com`	`wsdm2009.org`	`wired.com`
`cnn.com`	-	0	0.41
`wsdm2009.org`	0	-	0.81
`wired.com`	0.41	0.81	-

	`news`	`web`	`tech`
`news`	-	0	0
`web`	0	-	0.41
`tech`	0	0.41	-

Notice how the similarity of `news` with both `web` and `tech` is zero in the distributional case, whereas it is non-zero in the projection case above. This is due to the fact that the tag `news` shares only one resource, `wired.com`, with both `web` and `tech`. The resource `wired.com` has zero information content for tags, as it is associated with all of them. Thus it gives no contribution to tag similarities.

For macro and collaborative aggregation, an analogous definition applies to local (per-user) *matching* similarity:

$$\sigma_u(x_1, x_2) = -\sum_{y \in X_1^u \cap X_2^u} \log p(y|u). \tag{4.18}$$

Considering again the case in Figure 4.4, we need to compute the conditional probabilities for the two users. For `alice`, we have for resource probability:

$$p(\texttt{cnn.com}|\texttt{alice}) = 1/3 \tag{4.19}$$

$$p(\texttt{wsdm2009.org}|\texttt{alice}) = 2/3 \tag{4.20}$$

$$p(\texttt{wired.com}|\texttt{alice}) = 0 \tag{4.21}$$

and for tag probability:

$$p(\texttt{news}|\texttt{alice}) = 1/2 \tag{4.22}$$

$$p(\texttt{web}|\texttt{alice}) = 1/2 \tag{4.23}$$

$$p(\texttt{tech}|\texttt{alice}) = 1/2. \tag{4.24}$$

We compute the similarity matrices as we did above, separately for users `alice` and `bob`, and then we sum them to obtain the aggregated similarity matrices below:

	cnn.com	www2009.org	wired.com
cnn.com	-	0	1.10
www2009.org	0	-	0
wired.com	1.10	0	-

	news	web	tech
news	-	0	0
web	0	-	0
tech	0	0	-

Notice how computing per-user similarities and then aggregating over users produces more sparse similarity matrices that aggregating over users first. In the example of Figure 4.4, due to the tiny size of the folksonomy, the consequences are extreme: the contribution of user `alice` to both matrices is zero, and for tag similarities this is also true for user `bob`, so that all entries of the aggregated tag similarity matrix are zero.

Collaborative filtering is able to extract more signal when aggregating similarities over users, as it exposes the similarity that is implicit in the user context. In our example, when we modify the probabilities of tags and resources as described in § 4.2.1.4, we find for `alice`:

$$p(\texttt{cnn.com}|\texttt{alice}) = 1/4 \tag{4.25}$$

$$p(\texttt{www2009.org}|\texttt{alice}) = 1/2 \tag{4.26}$$

$$p(\texttt{wired.com}|\texttt{alice}) = 0 \tag{4.27}$$

$$p(\texttt{news}|\texttt{alice}) = 1/3 \tag{4.28}$$

$$p(\texttt{web}|\texttt{alice}) = 1/3 \tag{4.29}$$

$$p(\texttt{tech}|\texttt{alice}) = 1/3. \tag{4.30}$$

For `bob`:

$$p(\texttt{cnn.com}|\texttt{bob}) = 1/4 \tag{4.31}$$

$$p(\texttt{www2009.org}|\texttt{bob}) = 0 \tag{4.32}$$

$$p(\texttt{wired.com}|\texttt{bob}) = 3/4 \tag{4.33}$$

$$p(\texttt{news}|\texttt{bob}) = 2/3 \tag{4.34}$$

$$p(\texttt{web}|\texttt{bob}) = 1/3 \tag{4.35}$$

$$p(\texttt{tech}|\texttt{bob}) = 1/3. \tag{4.36}$$

The probabilities of the (per-user) dummy tag t^* and dummy resource r^* used in the construction of § 4.2.1.4 are:

$$p(t^*_{\text{alice}}|\texttt{alice}) = 2/3 \tag{4.37}$$

$$p(r^*_{\text{alice}}|\texttt{alice}) = 3/4 \tag{4.38}$$

$$p(t^*_{\text{bob}}|\texttt{bob}) = 2/3 \tag{4.39}$$

$$p(r^*_{\text{bob}}|\texttt{bob}) = 3/4. \tag{4.40}$$

The resulting similarity matrices for collaborative aggregation are:

	cnn.com	www2009.org	wired.com
cnn.com	-	0.41	0.81
www2009.org	0.41	-	0
wired.com	0.81	0	-

	news	web	tech
news	-	0.86	0.86
web	0.86	-	1.56
tech	0.86	1.56	-

Notice how collaborative filtering recovers non-zero values for the tag similarities.

4.2.2.2 Overlap

Projection-aggregated *overlap similarity* is defined as

$$\sigma(x_1, x_2) = \frac{|X_1 \cap X_2|}{\min(|X_1|, |X_2|)} \tag{4.41}$$

while distributional overlap is given by

$$\sigma(x_1, x_2) = \frac{\sum_{y \in X_1 \cap X_2} \log p(y)}{\max(\sum_{y \in X_1} \log p(y), \sum_{y \in X_2} \log p(y))}. \tag{4.42}$$

Local overlap for macro and collaborative aggregation is

$$\sigma_u(x_1, x_2) = \frac{\sum_{y \in X_1^u \cap X_2^u} \log p(y|u)}{\max(\sum_{y \in X_1^u} \log p(y|u), \sum_{y \in X_2^u} \log p(y|u))}. \tag{4.43}$$

4.2.2.3 Jaccard

Jaccard similarity aggregated by projection is

$$\sigma(x_1, x_2) = \frac{|X_1 \cap X_2|}{|X_1 \cup X_2|}. \tag{4.44}$$

Distributional Jaccard similarity is defined by

$$\sigma(x_1, x_2) = \frac{\sum_{y \in X_1 \cap X_2} \log p(y)}{\sum_{y \in X_1 \cup X_2} \log p(y)} \tag{4.45}$$

while the macro and collaborative versions are based on

$$\sigma_u(x_1, x_2) = \frac{\sum_{y \in X_1^u \cap X_2^u} \log p(y|u)}{\sum_{y \in X_1^u \cup X_2^u} \log p(y|u)}. \tag{4.46}$$

4.2.2.4 Dice

The projected *dice coefficient* is

$$\sigma(x_1, x_2) = \frac{2|X_1 \cap X_2|}{|X_1| + |X_2|} \tag{4.47}$$

with its distributional version defined as

$$\sigma(x_1, x_2) = \frac{2 \sum_{y \in X_1 \cap X_2} \log p(y)}{\sum_{y \in X_1} \log p(y) + \sum_{y \in X_2} \log p(y)} \tag{4.48}$$

and the macro and collaborative dice built upon

$$\sigma_u(x_1, x_2) = \frac{2 \sum_{y \in X_1^u \cap X_2^u} \log p(y|u)}{\sum_{y \in X_1^u} \log p(y|u) + \sum_{y \in X_2^u} \log p(y|u)}. \tag{4.49}$$

4.2.2.5 Cosine

Cosine similarity with binary projection is given by

$$\sigma(x_1, x_2) = \frac{X_1}{\sqrt{|X_1|}} \cdot \frac{X_2}{\sqrt{|X_2|}} = \frac{|X_1 \cap X_2|}{\sqrt{|X_1| \cdot |X_2|}}. \tag{4.50}$$

For the distributional version of the cosine, it is natural to use the frequency-weighted representation:

$$\sigma(x_1, x_2) = \frac{X_1}{||X_1||} \cdot \frac{X_2}{||X_2||} = \frac{\sum_y w_{x_1 y} w_{x_2 y}}{\sqrt{\sum_y w_{x_1 y}^2} \sqrt{\sum_y w_{x_2 y}^2}}. \tag{4.51}$$

The macro and collaborative aggregation versions are based on local cosine

$$\sigma_u(x_1, x_2) = \frac{X_1^u}{\sqrt{|X_1^u|}} \cdot \frac{X_2^u}{\sqrt{|X_2^u|}} = \frac{|X_1^u \cap X_2^u|}{\sqrt{|X_1^u| \cdot |X_2^u|}} \tag{4.52}$$

where in the collaborative case the construction of § 4.2.1.4 is applied without need of log-odds computations.

4.2.2.6 Mutual Information

Next, we consider *mutual information*. With projection and distributional aggregation we define

$$\sigma(x_1, x_2) = \sum_{y_1 \in X_1} \sum_{y_2 \in X_2} p(y_1, y_2) \log \frac{p(y_1, y_2)}{p(y_1)p(y_2)} \tag{4.53}$$

where for the projection case the probabilities $p(y)$ are defined in the usual manner (§ 4.2.1.2), and the joint probabilities $p(y_1, y_2)$ are also based on resource/tag (row/column) normalization:

$$p(y_1, y_2) = \frac{\sum_x w_{xy_1} w_{xy_2}}{\sum_x 1}. \tag{4.54}$$

With distributional aggregation, the joint probabilities must be matrix rather than row/column normalized; we compute fuzzy joint probabilities from the weighted representation:

$$p(y) = \frac{\sum_x w_{xy}}{\sum_{r,t} w_{rt}}, \quad p(y_1, y_2) = \frac{\sum_x \min(w_{xy_1}, w_{xy_2})}{\sum_{r,t} w_{rt}} \tag{4.55}$$

where $\min$ is a fuzzy equivalent of the intersection operator. Finally, macro and collaborative aggregation of local mutual information use

$$\sigma_u(x_1, x_2) = \sum_{y_1 \in X_1^u} \sum_{y_2 \in X_2^u} p(y_1, y_2|u) \log \frac{p(y_1, y_2|u)}{p(y_1|u)p(y_2|u)} \tag{4.56}$$

where simple and joint probabilities are resource/tag (row/column) normalized for each user's binary representation, and collaborative mutual information uses the construction and probability definition of § 4.2.1.4.

4.2.2.7 Maximum Information Path

The last measure we consider is *maximum information path* (mip) [MM09]. The maximum information path similarity is an extension of traditional shortest-path based similarity

measures [RMBB89] and Lin's similarity measure [Lin98]. Maximum information path differs from traditional shortest-path similarity measures by taking into account Shannon's information content of shared tags (resp. resources). Lin's similarity measure only applies to hierarchical taxonomies, such as is the case for bookmarks organized in folder and subfolders (cf. § 4.1.1). If the folksonomy is derived from such a hierarchy, the two measures are equivalent. However when the folksonomy includes non-hierarchical annotations, Lin's measure is undefined while maximum information path similarity is well defined and captures the same intuition. Figure 4.5 is an example. Namely, that the semantic association between two objects is determined by the ratio between the maximum information they have in common and the information they do not share. In the hierarchical case the maximum shared information coincides with the lowest common ancestor, which is unique; in a general folksonomy there may be many paths between two objects, and the maximum information path passes through the most specific shared tag. Because of the dependency on log-odds, we ignore maximum information path with projection aggregation. We define maximum information path for the distributional case as

$$\sigma(x_1, x_2) = \frac{2\log(\min_{y \in X_1 \cap X_2}[p(y)])}{\log(\min_{y \in X_1}[p(y)]) + \log(\min_{y \in X_2}[p(y)])}. \tag{4.57}$$

Maximum information path for macro and collaborative aggregation is

$$\sigma_u(x_1, x_2) = \frac{2\log(\min_{t \in X_1^u \cap X_2^u}[p(t|u)])}{\log(\min_{t \in X_1^u}[p(t|u)]) + \log(\min_{t \in X_2^u}[p(t|u)])}. \tag{4.58}$$

4.3 Discussion

We have presented various techniques for extracting semantic similarity from social annotations. There are two main representations of user annotations that we consider for

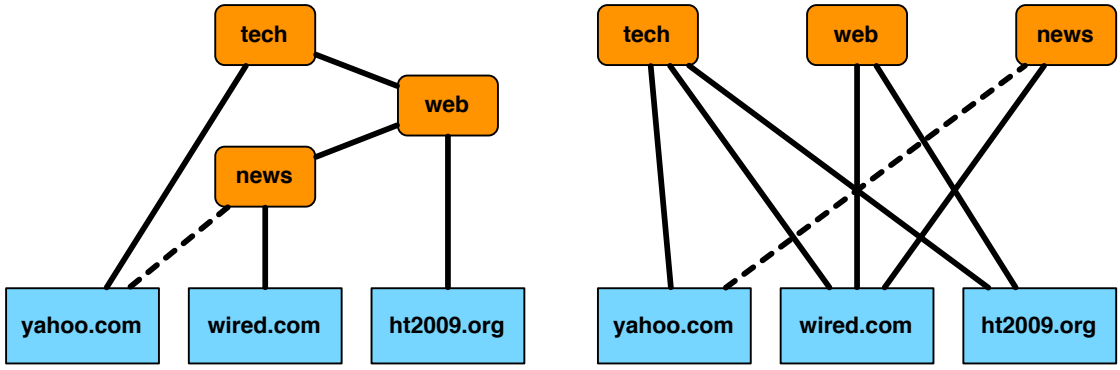


Figure 4.5: The annotations of user Charlie are visualized on the left in a hierarchical taxonomy and on the right as a flat folksonomy. If Charlie does not tag `yahoo.com` with `news`, the annotations are hierarchical. The maximum information path similarity between `wired.com` and `ht2009.org` is determined by the lowest common ancestor on the left, the tag `web`. Indeed we see on the right that this is the most specific tag shared by the two resources (`tech` is also shared but it is more general and therefore less informative). If Charlie tags `yahoo.com` with `news`, the hierarchy breaks as multiple paths connect `wired.com` and `ht2009.org`. The lowest common ancestor is no longer defined, however in the folksonomy we can still identify `web` as the most specific shared tag.

extracting the similarity: hierarchical and tripartite. The hierarchical similarity measures are suitable only to gauge relationships among resources, while the tripartite similarity measures are suitable for measuring relationships among resources, tags, and users. We will now explore in § 5 how to assemble a large scale network with these measures. Then in § 6, we will ground these measures against a set of reference similarities. Later in § 7.1 and § 7.2, we will directly evaluate a subset of the similarity measures through tag relation prediction and resource recommendation.

5

Assembly of Socially Induced Semantic Networks

The semantic similarity measures in § 4 cannot be properly evaluated or deployed without a discussion dedicated to network assembly. There are some measures such as Lin (cf. § 4.1.1) and the tripartite similarity measures (cf. § 4.2.2) with projection or distributional aggregation (cf. § 4.2.1) that depend on global quantities. As collaborative systems continue to grow and evolve, the network assembly operation can become quite expensive as we will illustrate.

Here we present a number of techniques for building a semantic similarity network [MRM08, MCM+09b]. In § 5.1 and § 5.2 we present methods for assembling a semantic network where the similarity measures have global dependencies. In § 5.3, we will describe a technique for building a semantic similarity network incrementally, i.e., free of global dependencies. The benefits of incremental assembly are presented in § 5.4.

5.1 Naïve Assembly

Naïve assembly entails the construction of the semantic similarity network on a per user basis where the similarity measure has global dependencies. When considering Lin

(cf. § 4.1.1), it is natural to visit each user's bookmark file, compute the similarities, and update the global similarity network. For the tripartite similarity measures, it only makes sense to visit each user to update the projection and distributional aggregation matrices. Updating the aggregation matrix is a constant operation with runtime complexity $O(1)$. Incremental Lin and the tripartite similarity measures with macro and collaborative aggregation naturally fit into naïve assembly, but they are free of global dependencies. We will ignore naïve assembly for similarity measures free of global dependencies, deferring this discussion to § 5.3.

Naïve assembly with Lin was the original method for constructing GiveALink's semantic similarity network. As of January 2008, GiveALink's collection included $1,883,722$ Web resources (links), $520,856$ categories (folders), and $4,725$ unique donations to the system. As each user was processed, the results were accumulated in a separate global representation of the network. This global network representation is required to be readily accessible because as each user is processed, the relationships of resources (edge similarities) affected are unknown *a priori*. Algorithm 1 illustrates this method of assembling the semantic similarity network where r is a single resource and N the number of resources in the system. Additionally, let D be the set of donations (users) and A the global similarity matrix. $A_{i,j}$ refers to the weighted edge between resources i and j.

Algorithm 1 Naïve approach to building a semantic network from a hierarchy

```
N ⇐ number of Web resources
Initialize A, |A| = N^2
for each d ∈ D do
  for each r_x ∈ d do
    for each r_y ∈ d do
      if r_x ≠ r_y then
        A_{x,y} ⇐ A_{x,y} + s_d(r_x, r_y)
      end if
    end for
  end for
end for
```

Table 5.1: Adjacency List Representation

x_1	x_2 $s_{(x_1,x_2)}$	x_3 $s_{(x_1,x_3)}$	...	x_N $s_{(x_1,x_N)}$
x_2	x_1 $s_{(x_2,x_1)}$	x_3 $s_{(x_2,x_3)}$	...	x_N $s_{(x_2,x_N)}$
x_3	x_1 $s_{(x_3,x_1)}$	x_2 $s_{(x_3,x_2)}$	...	x_N $s_{(x_3,x_N)}$
...			...	
x_N	x_1 $s_{(x_N,x_1)}$	x_2 $s_{(x_N,x_2)}$	...	x_{N-1} $s_{(x_N,x_{N-1})}$

The primary limitation is the storage while aggregating the similarity values. The space complexity of Algorithm 1 is $O(N^2)$. Established tools such as the matrix template library (`osl.iu.edu/research/mtl`), the matrix market library (`math.nist.gov/MatrixMarket/index.html`), and the boost graph library (`boost.org/libs/graph/doc/`) were tested, but still suffer from the space complexity. With the emergence of these problems materializing with only $60,000$ resources in the collection, alternative methods were developed to build our semantic similarity network.

5.2 Item Centric Assembly

In order to construct the network, while controlling memory complexity, we draw upon a technique used in sparse matrix computation. Duff [DER86] introduces the concept of vector register windows for large scale matrix computations. For our purposes, we define each window to be a row in the matrix V, and each entry representing an incident edge to another node. Table 5.1 illustrates this matrix where x is a node, and $s_{x,y}$ the similarity measure. In this example, $s_{(x,y)}$ implies $s(x,y)$ or $\sigma(x,y)$ from § 4. This technique limits space complexity to $O(N)$ where N is the number of items (resources/annotations). At the conclusion of each row, the window is stored and then cleared. This solution represents a cache friendly method for assembling the network. For the following algorithms, let A be the global semantic similarity matrix.

5.2.1 Hierarchical Representation

Algorithm 2 details the method for building the semantic similarity network based on a hierarchical representation as presented in § 4.1. We need to define D_{r_x} to be the set of donations (users) containing resource r_x, and function s as Lin. V_{r_x} refers to the entry in V representing r_x.

Algorithm 2 Item centric assembly with *Lin*

```
for each resource r_x do
    Initialize V, |V| = N
    for each d ∈ D_{r_x} do
        for each r_y ∈ d do
            V_{r_y} ⇐ V_{r_y} + s_d(r_x, r_y)
        end for
    end for
    Insert V into the corresponding row A for r_x
end for
```

5.2.2 Tripartite Representation

The item centric assembly fits naturally with the tripartite similarity measures. With a global aggregation matrix (projection and distributional), we can visit each pair of resources/tags in order. For example let us consider computing a semantic similarity matrix based on *Matching* with distributional aggregation (cf. Equation 4.11). Let us define x to be an object (tag, resource, or user) and X its vector representation. V_x refers to the entry representing x. The probability function $p(y)$ is the same as Equation 4.5. Finally, let N be the number of objects in the system. Algorithm 3 illustrates this technique of assembling a semantic network.

Algorithm 3 Item centric assembly with *matching*

for each object x_1 **do**
  Initialize V, $|V| = N$
  for each object x_2 **do**
    $\sigma = 0$
    for each attribute y in $X_1 \cap X_2$ **do**
      $\sigma \Leftarrow \sigma + \log p(y)$
    end for
    $V_2 \Leftarrow -\sigma$
  end for
  Insert V into the corresponding row of A for x_1
end for

5.2.3 Limitations

Algorithm 2 and Algorithm 3 are able to hold the space complexity constant $O(N)$, but they still suffer from a runtime complexity of $O(N^2)$. When a new donation or change is made to an existing profile enters the system, A needs to be completely reconstructed.

For tripartite similarity, Algorithm 3 can easily be generalized to the other measures with the exception of mutual information. Mutual information requires the calculation of joint probabilities between attributes. Algorithm 4 illustrates computing joint probabilities with distributional aggregation. We refer to an object as having a set of attributes, but also notice that an attribute has a set of objects. Let J be the joint probability matrix, L the set of attributes, and T the set of triples. $J_{x,y}$ refers to the entry in the joint probability matrix corresponding to the row for attribute x and the column for attribute y. Finally, O_x is the vector representation of attribute x. Notice that this calculation requires additional runtime and space complexity $O(|L|^2)$.

Algorithm 4 Computation of the joint probabilities of attributes for mutual information with distributional aggregation.

```
Initialize joint probability matrix J, |J| = |L|^2
for attribute a_1 ∈ L do
  for attribute a_2 ∈ L do
    f = 0
    for object o ∈ O_{a_1} ∩ O_{a_2} do
      f = f + min(o_{a_1}, o_{a_2})
    end for
    J_{a_1,a_2} = f / |T|
  end for
end for
```

5.3 Incremental Assembly

All of the techniques presented so far allow the network to be out of date once a user adds or changes her profile, until the entire similarity matrix is recomputed. Unfortunately, all of the assembly techniques presented so far suffer from quadratic space complexity, quadratic time complexity, or a combination thereof. To maintain the similarity network in real time, we need a measure for a user's contribution to the network that does not depend on global quantities. A more agile technique for maintaining a large similarity network will also allow users to observe their contributions to the global network immediately, encouraging active participation. In order to achieve this, let us explore an incremental approach in which each user's contribution to the global semantic similarity network is able to be computed independently.

Incremental Lin (cf. § 4.1.2) and the tripartite similarity measures (cf. § 4.1.2) with macro and collaborative aggregation (cf. § 4.2.1) have the benefit of being computable on a per user basis without global dependencies. This alleviates the space and time complexity problem by localizing the similarity computation. Also, since we are primarily interested in ranking, the similarities do not require normalization to $[0, 1]$. Algorithm 5 illustrates this idea of incrementally assembling a semantic similarity network. Let us define M^u to

be a user u's similarity matrix, X_u to be u's object set, and $M^u_{i,j}$ to be the entry in M^u for the similarity between objects i and j.

Algorithm 5 Assembling a similarity matrix incrementally

```
Initialize user similarity matrix M^u, |M^u| = |X_u|^2
for each item x_i in X_u do
  for each item x_j in X_u do
    M^u_{i,j} = s_u(x_i, x_j)
  end for
end for
for each item x_i in M^u do
  for each item x_j in M^u do
    if i ≠ j then
      A_{x_i,x_j} = A_{x_i,x_j} + M_{x_i,x_j}
    end if
  end for
end for
```

Algorithm 5 uses incremental Lin as an example, but one can easily plug in the tripartite similarity measures. Furthermore, if this is an existing user who has contributed to A, then the past similarity contributions must be subtracted from A prior to executing Algorithm 5.

5.4 Runtime Analysis

The end goal is to develop a *scalable* algorithm for assembling a social semantic similarity network from an evolving dataset. From a practical perspective, we consider as *scalable* those measures that can be updated to reflect new annotations at a pace that can keep up with the stream of incoming annotations. The problem with similarity measures that have global dependencies is that the similarities must be recomputed from scratch as user profiles and frequency weights are updated. This is not scalable because its complexity clearly grows with the size of the system (e.g., number of bookmark donations or number of triples) as we will discuss.

5.4.1 Item Centric Assembly

Recall from § 5.2 that item centric assembly requires the semantic similarity network to be represented in a matrix. Each entry in this matrix is the similarity between two objects. Item centric assembly visits each row in the matrix, while computing the similarity to all incident objects. The item centric method allows the similarity calculation to control space complexity overcoming a major limitation of naïve assembly (cf. § 5.1). Figure 5.1 illustrates item centric assembly.

We implemented in C++ item centric assembly. The hierarchical information captured from user donations was stored in a MySQL database. To optimize the implementation, we cached all necessary hierarchical information into local Berkeley Databases.

This implementation ran on Big Red (The Big Red Cluster, `rc.uits.iu.edu/hps/research/bigred/index.shtml`), a resource available to the research community at Indiana University. The Big Red cluster consists of 768 nodes in which each node has two dual-core PowerPC 970 MP processors with 8GB of memory. We distributed the Web links among separate processes for two reasons. First, each resource and all of its incident nodes can be created independent of one another. Second, assembling the entire network on one node is impossible due to cluster runtime quotas. As a result, each node was in charge of building part of the graph, followed by storing the *subgraph* to an exclusive file unknown to the other processes. Each subgraph may be indexed such that given a Web resource, a list of incident nodes and their corresponding similarities are retrieved. Table 5.1 illustrates this representation. For our purposes we combined all of the subgraphs into a single text file for upload to the database. Of course, one may choose not to combine the subgraphs at all.

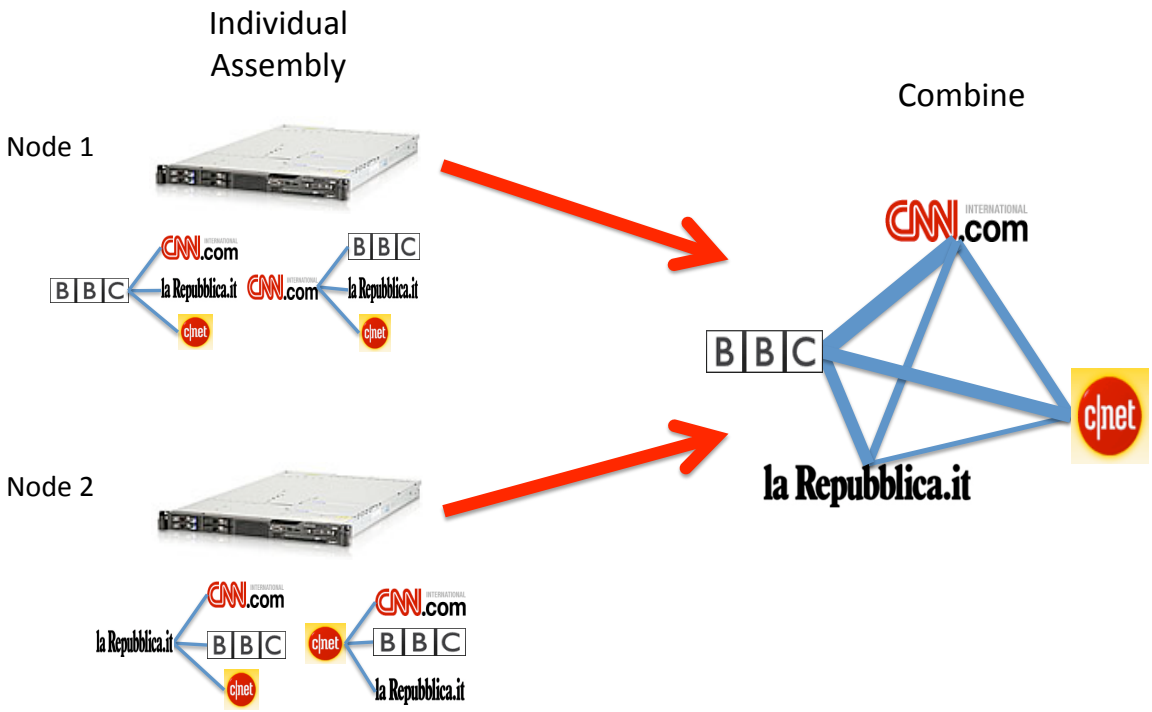


Figure 5.1: Item centric implementation with two nodes. Node 1 computes the similarities of all incident edges to `bbc.com` and `cnn.com`, while node 2 is charged with computing the similarities with `repubblica.it` and `cnet.com`. When all nodes complete their assigned urls, we combine the subgraphs to form the global semantic similarity network.

Analysis

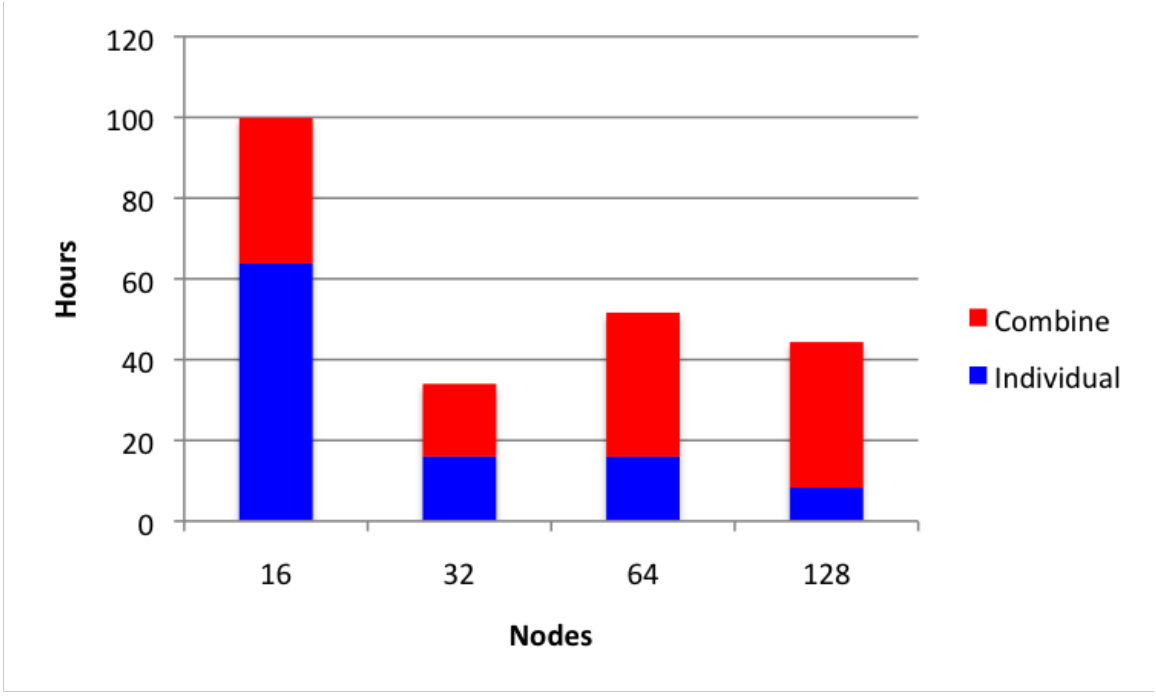
The failure to construct the network with one node on BigRed forced us to explore distributing the assembly onto separate processors. As stated earlier, each processor is in charge of building a part of the similarity network. Here we look at the different timings when dividing the work among cluster nodes. These numbers reflect a system where each worker in charge of building a subgraph receives exclusive access to the CPU core, but must share the large storage disk with all other users of the cluster. Thus we present the actual time it took to assemble the matrix in hours as well as the system time in minutes in Figure 5.2.

The distributed technique allows us to build a large semantic similarity network of about 1.9 million nodes and 18.9 billion edges in under 3 days. Because Big Red is a general use machine these numbers vary with system load, as evident from Figure 5.2. It is also important to note that it is completely possible that the nodes in the individual assembly stage do not start simultaneously.

For much larger systems, even the technique discussed here may not scale appropriately. While the space problem is solved, the network assembly algorithm still suffers from a runtime complexity $O(N^2)$.

5.4.2 Incremental Assembly

Let us now analyze the complexity of the measures with global dependencies versus those that do not. The measures without global dependencies are incremental Lin (cf. § 4.1.2) and the tripartite measures (cf. § 4.2.2) with macro/collaborative aggregation (cf. § 4.2.1). Because the hierarchical representation is a specific case of the tripartite representation (cf. § 4.2), we will concentrate on analyzing network assembly with mutual information applied to the tripartite structure.



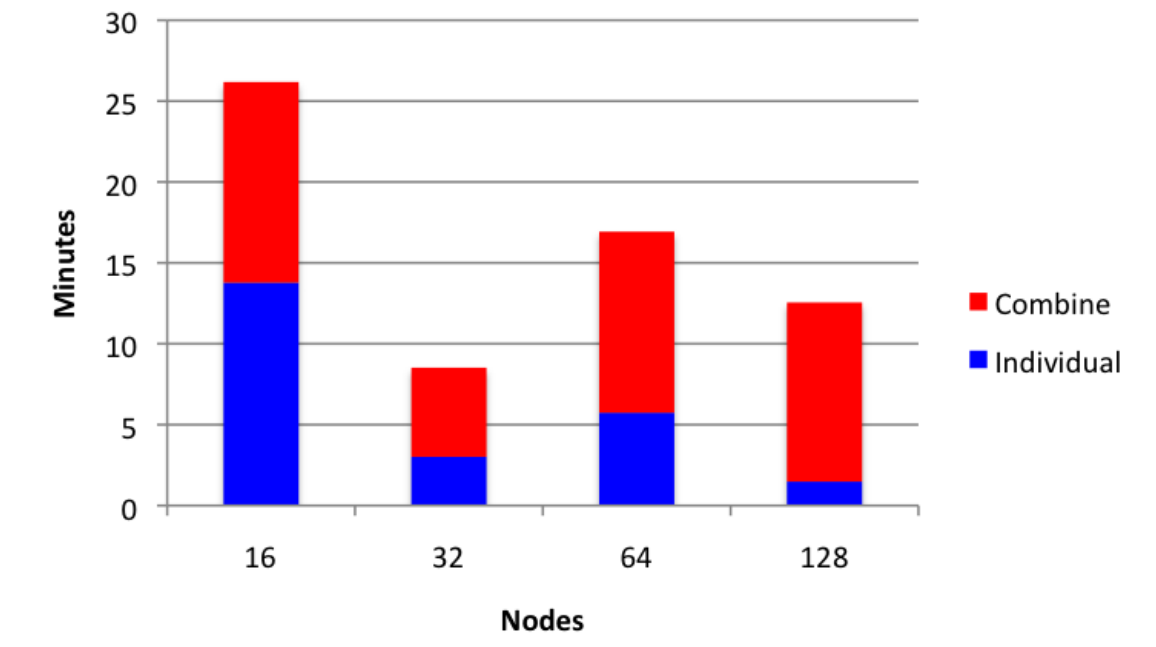


Figure 5.2: The actual time to assemble `GiveALink`'s semantic similarity network according to the number of nodes. The top chart contains the *real* world times in hours. The bottom chart are the *system* times in minutes. The *individual* bars represent the `maximum` time a node took to complete a subgraph. The *combine* bars are the times to integrate the individual subgraphs.

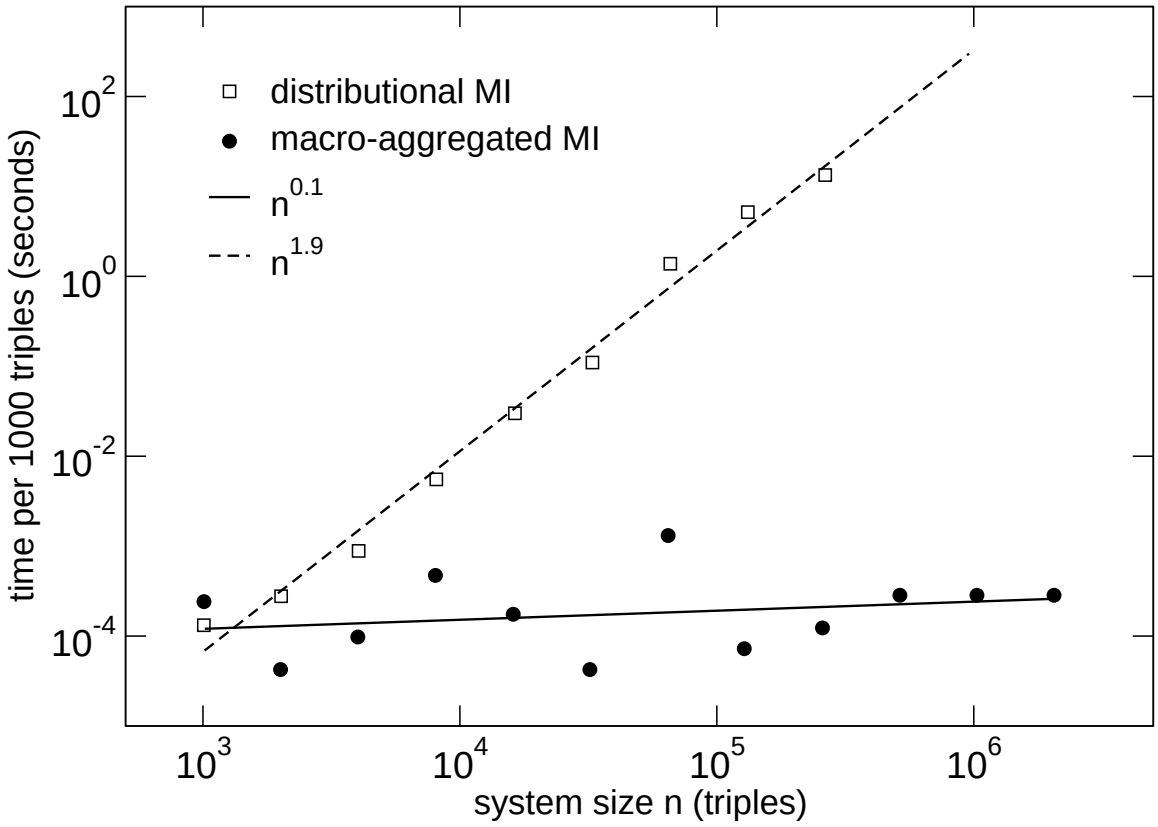


Figure 5.3: Scalability of the mutual information computation of resource similarity for different aggregation methods on the tripartite representation. We measured the CPU time necessary to update the similarities after a constant number of new annotations are received, as a function of system size n. Best polynomial fits $time \sim n^{\alpha}$ are also shown.

An average-case complexity analysis is problematic given the long-tailed distributions typical of user profiles; quantities like average densities and average overlap are not necessarily characteristic of the system, given the huge fluctuations associated with broad distributions. Therefore we turn to an empirical analysis to examine how update complexity scales with system size.

Figure 5.3 compares the computation of mutual information between resources using two aggregation methods, namely distributional (or projection) versus macro (or collaborative) aggregation. As the plot shows, distributional aggregation measures scale almost quadratically with system size, while macro-aggregated measures can be updated incrementally in almost constant time. Therefore incremental assembly outperforms the other assembly techniques in terms of runtime performance. Furthermore, space complexity can be held constant by applying Duff's vector register window technique to incremental assembly.

6

Evaluation with Reference Similarities

The previous section quantified the runtime performance of the similarity calculations. Now, we will explore the effectiveness of the similarity measures. We assess the similarity measures by how well they approximate a reference set of similarities. Since we are not interested in the actual similarity values, we turn to a non-parametric analysis that only looks at the *ranking* of the pairs by similarity. This reflects the intuition that while it is problematic for someone to quantify the similarity between two objects, it is natural to rank pairs on the basis of their similarities, as in "a chair is more similar to a table than to a computer." Indeed in most applications we envision (e.g. search, recommendation) ranking is key: given a particular tag or resource we want to show the user a small set of most similar objects. We thus turn to *Kendall's* τ correlation between the similarity vectors whose elements are similarities among objects.

Kendall's τ has a range of $[-1, 1]$ with 1 being a perfect correlation and -1 being a perfect negative correlation. If the order of the similarities for the edges agree, Kendall's τ increases, and if they disagree, τ decreases. We compute τ efficiently with Knight's $O(N \log N)$ algorithm as implemented by Boldi et al. [BSV05]. Assuming we have access to reference similarity values for all pairs considered, a higher correlation is to be interpreted as a better agreement with the grounding and thus as evidence of a better similarity measure. As an illustration, let us consider the resources of Figure 4.4 and two similarity

measures Measure A and Measure B. Below we show the ranked similarities and the resulting τ values obtained by comparison with the reference similarity measure. We conclude in this example that Measure B is better then Measure A because it agrees more with the reference measure, as reflected by the higher τ.

Rank	Reference	Measure A	Measure B
1	wired.com–www2009.org	cnn.com–wired.com	cnn.com–www2009.org
2	cnn.com–www2009.org	wired.com–www2009.org	wired.com–www2009.org
3	cnn.com–wired.com	cnn.com–www2009.org	cnn.com–wired.com
τ	1	1/3	2/3

6.1 Hierarchical Similarity Measures

We will now evaluate the hierarchical similarity measures Lin (cf. § 4.1.1) and incremental Lin (cf. § 4.1.2). Incremental Lin overcomes a major shortcoming of Lin by removing the dependency on the size of the super tree, i.e., $|R|$ in Equation 4.1. The construction of the super tree is due to our desire to capture collaborative filtering (cf. § 3.3).

The reference semantic resource network is based on a URL collection of the Open Directory Project (dmoz.org). In particular we rely on Maguitman et al.'s [MMRV05, MME+06] graph-based similarity measure, which takes both hierarchical and non-hierarchical structure into account. The ODP graph similarity is an appropriate reference because it was shown to be very accurate through a user study [MME+06].

We use incremental Lin and set $\delta = 0$ and $\delta = 10^6$ to create two similarity networks. The value 10^6 is chosen as an approximation of the missing information from other users' bookmarks, i.e., the size of the super tree $|R|$. Additionally, we use Lin (cf. § 4.1.1) to induce a third network. These networks are then rank correlated with the reference network using Kendall's τ.

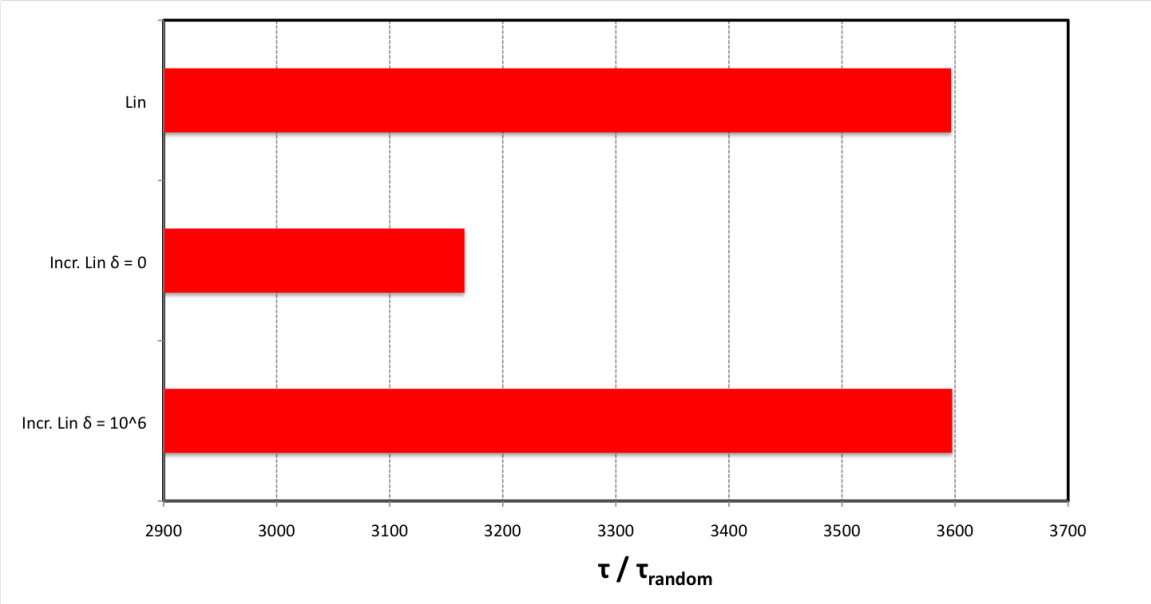


Figure 6.1: Resource-resource similarity accuracy, according to Kendall's τ correlations between similarity vectors generated by Lin, incremental Lin with $\delta = 0$, and incremental Lin with $\delta = 10^6$. All similarity measures perform significantly better than the randomly generated set of similarities $\tau = 2 * 10^{-5}$.

For the purposes of the experiment, we first identified the resources that have a corresponding node in the reference network. We then took the top 3.2×10^4 resources sorted by *strength* for use in our evaluations. The strength of each resource in the sample list is taken from the original similarity matrix defined in § 4.1. These resources represent the densest component of the network, and the resources with the most social activity. We define the strength of resource r_i as:

$$\kappa(r_i) = \sum_j s(r_j, r_i). \tag{6.1}$$

Considering two weighted edges (similarities) from the proposed network, the corresponding edges are taken from the reference network.

Figure 6.1 illustrates our findings. The results show that collaborative filtering provides a better indicator of semantic similarity, and it is worth capturing. Additionally, the

incremental similarity measure with $\delta = 10^6$ seems to perform as well as the original, non-scalable measure. All similarity measures perform significantly better than the randomly generated set of similarities.

6.2 Tripartite Similarity Measures

Next we analyze the tripartite similarity measures described in § 4.2.2 combined with the various aggregation methods in § 4.2.1. This investigation has the added benefit of including both tag and resource similarities in the evaluation. We look at two distinct datasets. The first dataset is a snapshot of the `BibSonomy.org` corpus. `BibSonomy.org` is a social bookmark and publication management system. The BibSonomy data represents a 'pure' folksonomy meaning the tool was designed from inception to be a collaborative tagging system. The second dataset is a snapshot of `GiveALink.org`. The earlier versions of GiveALink stored the hierarchical information provided by browser bookmarks. In the Spring of 2007, GiveALink's data was converted to triples. Given an individual's bookmark file, we visit each resource and follow the folders to the root. Each folder then becomes an annotation for that resource to derive the triple (cf. Figure 4.3). We define a GiveALink 'tag' to be a folder name replacing white spaces in the name with an '_'.

Let us provide more details regarding the BibSonomy and GiveALink datasets. The BibSonomy dataset is from December 2007, as available on the BibSonomy site (`BibSonomy.org/faq#faq-dataset-1`). We focused on the bookmark part of the system. The BibSonomy snapshot that we used contains $128,500$ bookmarks annotated by $1,921$ users with $58,753$ distinct tags. The GiveALink dataset is from September 2008. The most current user, tag, and resource information is available on the GiveALink site (`GiveALink.org/main/download`). The snapshot we used consist of $975,611$ bookmarks annotated by $5,036$ users with $95,187$ distinct tags.

Table 6.1: GiveALink/BibSonomy WordNet Details

Dataset	Overlap	% of Dataset	Attributes (Resources) Considered
GiveALink	15,253	35%	30,000
BibSonomy	17,041	29%	2,000

6.2.1 Tag Similarity Evaluation

We use the WordNet term collection for the semantic grounding of the tag similarity measures. In particular we rank tag pairs by their Jiang-Conrath distance [JC97], which combines taxonomic and information-theoretic knowledge. This WordNet distance measure is an appropriate reference as it was validated experimentally [BH06, CBHS08].

For GiveALink and BibSonomy, we focus on the subset of annotations whose tags overlap with the WordNet dataset. Table 6.1 details the samples taken from GiveALink and BibSonomy used in the evaluation. Similarities are computed between all pairs of tags in this set, however it was not possible to use the full annotation data from either dataset due to the space complexity and density of the tag networks when computing the joint probabilities for mutual information. For a fair comparison, let us evaluate the effectiveness of the measures by limiting the analysis to the most used resources. More specifically we select, among the resources in the overlap subset, those resources that appear in the larger number of triples across the entire folksonomy. These resources represent the subset with the most social activity and the least amount of noise. We then compute the similarities using all the folksonomy annotations relative to these top tags.

Figure 6.2 plots Kendall's τ correlation between each measure introduced in § 4.2.2 and the WordNet reference. To interpret maximum information path's performance with the GiveALink dataset, one must consider that a significant portion of the GiveALink folksonomy is based on a perfect hierarchy of folders and resources. In light of this, the results are consistent across datasets in many respects, the most obvious being the measures with macro aggregation underperforming their counterparts with other aggregation methods

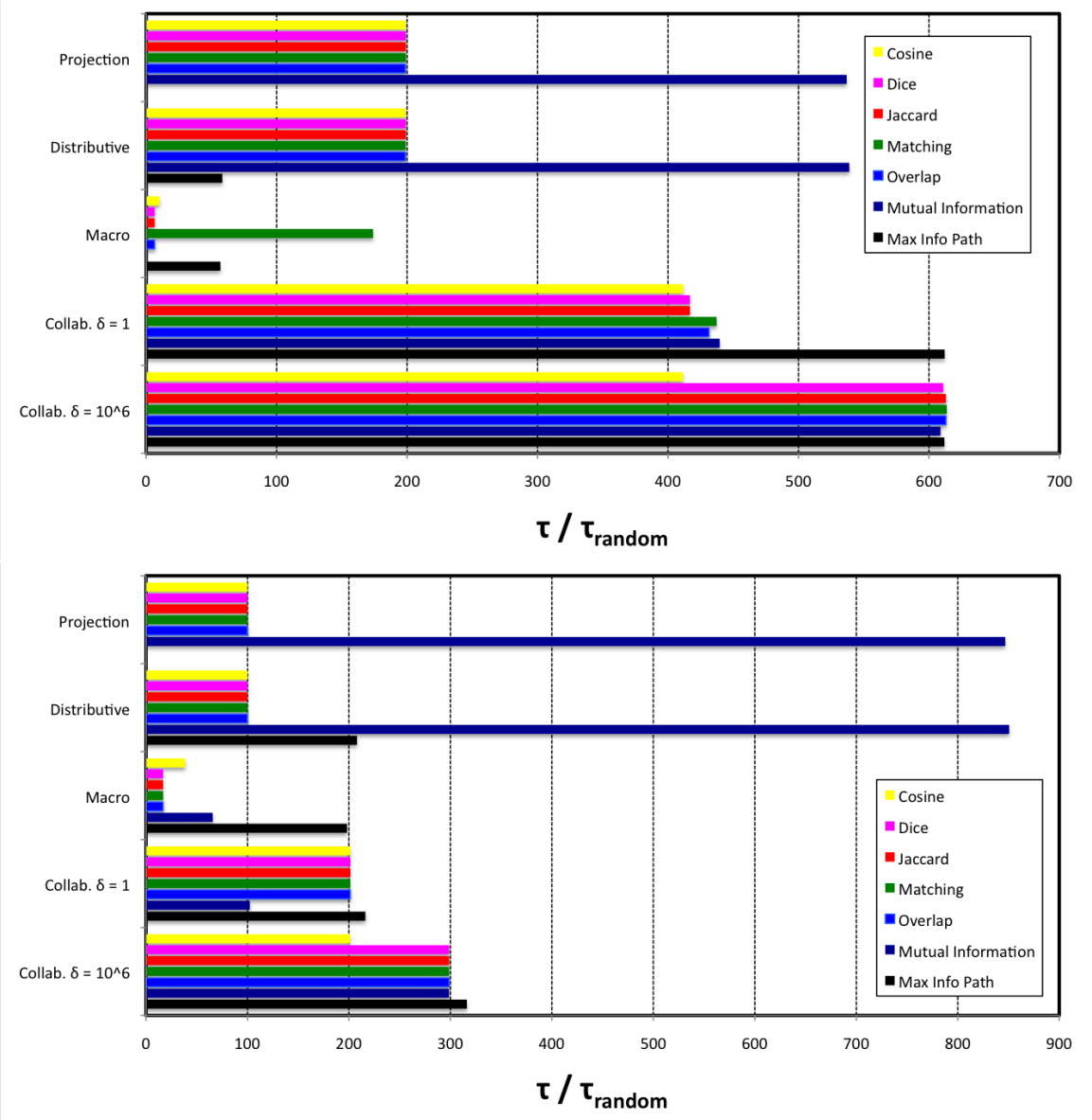


Figure 6.2: Tag-tag similarity accuracy, according to Kendall's τ correlations between the similarity vectors generated by the various measures and the reference similarity vector provided by the WordNet grounding measure. *Top:* Results using the BibSonomy tags. *Bottom:* Results using the GiveALink tags. All similarity measures perform significantly better than the randomly generated set of similarities $\tau = 10^{-4}$ for BibSonomy and $\tau = 10^{-4}$ for GiveALink. Maximum information path is only applied to the distributive, macro, and collaborative aggregation methods. Furthermore, because cosine does not depend on log-odds, the measure is excluded from collaborative where $\delta = 10^6$.

(projection, distributive, and collaborative). Furthermore, all similarity measures significantly outperform a randomly set of similarities (τ_{random}).

Focusing on the aggregation methods that depend on global frequencies (projection and distributive), mutual information with distributive aggregation give the best result. Cosine, dice, jaccard, matching, and overlap do not differ significantly from one another. The similar performance among the similarity measures is a result of limiting the total number of possible attributes (resources) during the similarity calculation. Limiting the number of attributes makes it more difficult to differentiate objects. For example, let us consider two tags a and b where a has links `cnn.com` and `politico.com` and b is a tag for `cnn.com`. If we select on the most popular links, we may remove `politico.com` from the similarity calculation. As a result of ignoring `politico.com`, *Matching* with projection would result in $\sigma(a, b) = 1$, while *Jaccard* with projection would also result in $\sigma(a, b) = 1$. The restriction on the number of attributes has adverse effects on the normalization inherent in cosine, dice, jaccard, and overlap regardless of the aggregation method.

When focusing on the aggregation methods without global dependencies (macro and collaborative), we find some consistency in the analysis. The most noticeable find is that collaborative aggregation provides a significant boost over macro. Furthermore, maximum information path with collaborative filtering performs well. The results in § 6.1 led us to explore changing the constant δ in Equation 4.8. Collaborative with $\delta = 10^6$ yields the best results among all aggregation methods free of global dependencies. In fact when applied to the GiveALink dataset, collaborative aggregation with maximum information path when $\delta = 10^6$ sets the standard for scalable, effective similarity measures.

Table 6.2: GiveALink/BibSonomy ODP Details

Dataset	Overlap	% of Dataset	Attributes (Tags) Considered
GiveALink	$30,000$	3%	$95,187$
BibSonomy	$3,323$	2.6%	$42,253$

6.2.2 Resource Similarity Evaluation

Similar to the evaluation with hierarchical similarity, we use the URL collection of the Open Directory Project (`dmoz.org`) for the semantic grounding of the resource similarity measures. We again rely on the graph-based similarity measure by Maguitman et al. [MMRV05, MME+06].

For our evaluation of resource similarity, we focus on the subset of the GiveALink and BibSonomy annotations whose resources overlap with the ODP. For the analysis using the GiveALink data, the overlap resulted in $975,611$ resources. The quadratic space complexity from assembling the resource network made us focus on the top $30,000$ most frequently annotated resources. Table 6.2 details the samples taken from GiveALink and BibSonomy used in the evaluation. Similarities are computed between all pairs of resources in this set, using the full annotation data from the folksonomies.

Figure 6.3 plots the Kendall's τ correlation between the tripartite similarity measures and the ODP similarity reference. As a baseline we computed τ with a randomly generated set of similarity values. The macro-aggregation similarity measures again are the worst-performing when compared to the corresponding measures with projection, distributive, or collaborative aggregation. Again, all similarity measures significantly outperform a randomly generated set of similarities (τ_{random}).

Let us focus on the measures dependent on global frequencies, i.e., the similarity measures with projection and distributional aggregation. The similarity measures with distributional aggregation performs either as good or better than the corresponding techniques with projection. Matching experiences a large boost with the BibSonomy dataset, but only

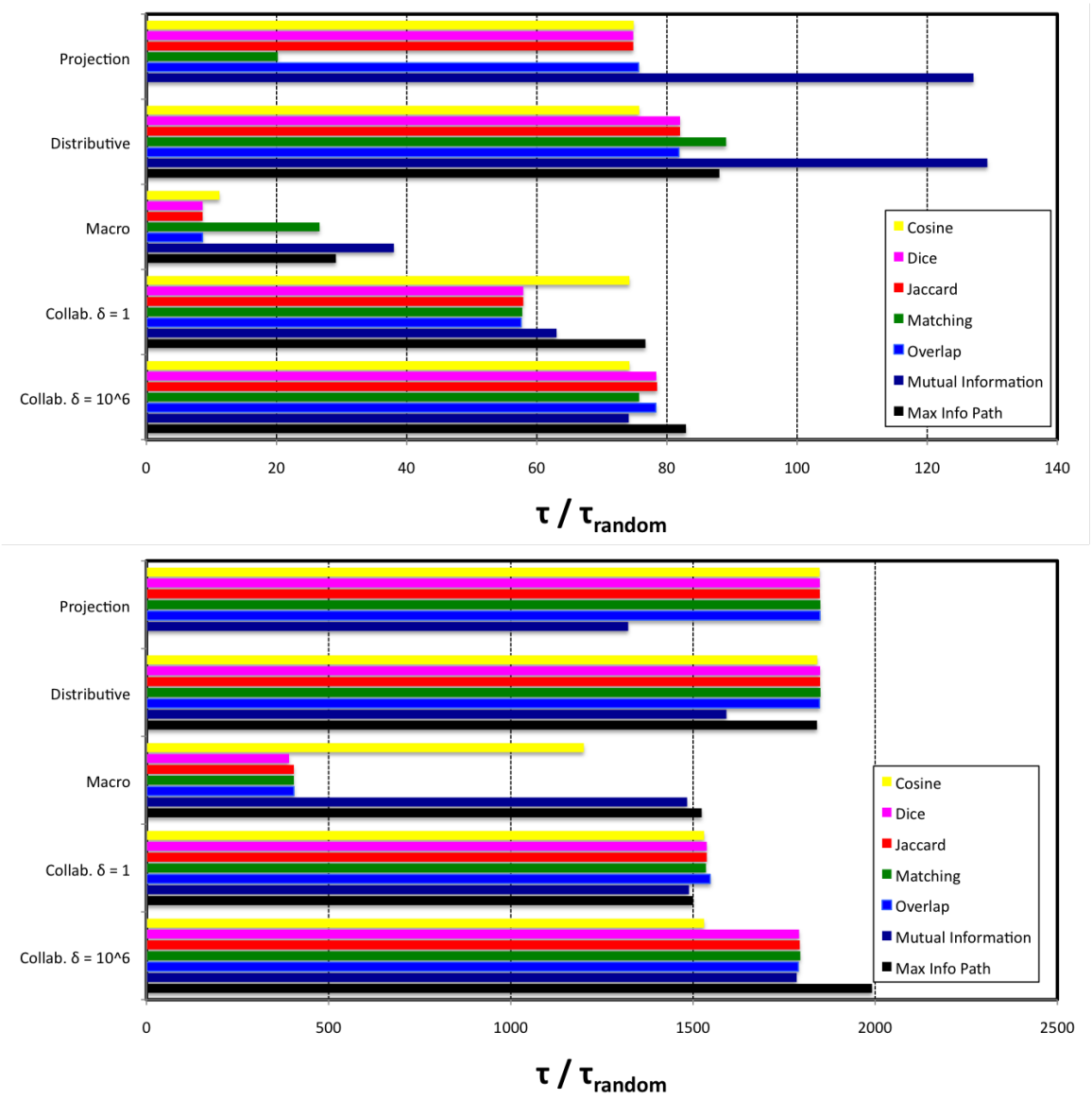


Figure 6.3: Resource-resource similarity accuracy, according to Kendall's τ correlations between the similarity vectors generated by the various measures and the reference similarity vector provided by the ODP grounding measure. *Top:* Results using the BibSonomy resources. *Bottom:* Results using the GiveALink resources. All similarity measures perform significantly better than the randomly generated set of similarities $\tau = 4 * 10^{-5}$ for BibSonomy and $\tau = 2 * 10^{-5}$ for GiveALink. Maximum information path is only applied to the distributive, macro, and collaborative aggregation methods. Furthermore, because cosine does not depend on log-odds, the measure is excluded from collaborative where $\delta = 10^6$.

a small increase with GiveALink. Furthermore, mutual information is by far the most accurate measure for BibSonomy, but underperforms the other measures when applied to the GiveALink data. This is a direct consequence of the conversion to *triples* from a hierarchy. Because mutual information depends on joint probabilities of attributes, in this case folder names, the conversion from a hierarchy results in the tags often being mutually exclusive from one another resulting in a sparse network. Additionally, the annotations that are shared are potentially very general (high entropy) since the common folders are closer to the root of the hierarchy.

Let us switch our focus to the aggregation methods free from global dependencies (macro and collaborative). There is clear evidence that collaborative aggregation provides a boost over macro-aggregation with the exception of maximum information path on the GiveALink dataset when $\delta = 1$. These results again suggest that collaborative filtering captures important semantic information in the folksonomy; the fact that two resources are annotated by the same user is telling about the relationship between these resources, beyond any tags they share. As we did with the tag similarity evaluation and likewise with the hierarchical similarity measure evaluation, we explored setting $\delta = 10^6$ in Equation 4.8. Furthermore, it should be noted that maximum information path performs competitively with the other measures for both datasets. Maximum information path with $\delta = 10^6$ outperforms all other incremental methods with the BibSonomy dataset and *all* other methods with the GiveALink dataset.

Again, much of the GiveALink folksonomy is based on a perfect hierarchy of folders and resources. This may have helped maximum information path perform better than distributional mutual information.

6.3 Discussion

The evaluations address both the computational complexity (cf. § 5.4) and the effectiveness of the similarity measures. These measures are based on various representations of user annotations, i.e., hierarchical (cf. § 4.1) and tripartite (cf. § 4.2).

The hierarchical similarity measure Lin has global dependencies. In order to address these dependencies, we presented incremental Lin (cf. § 4.1.2), which allows the resource similarities to be updated in real time. Both Lin and incremental Lin outperform a randomly generated set of similarities. On the other hand, the evaluation of incremental Lin shows that the accuracy of Lin can be recovered with a large improvement in runtime performance (cf. § 5.4.2).

In comparison to the corresponding tripartite similarity measures (cf. § 4.2.2), the ones with distributional aggregation (cf. § 4.2.1) perform well in reproducing a set of reference similarities. Unfortunately, distributional (or projection) has global dependencies. In order to address these dependencies, we introduced macro and collaborative aggregation. We found that collaborative aggregation consistently provides a boost over macro for all corresponding similarity measures. There are even some cases where collaborative aggregation outperforms the similarity measures with distributional aggregation in accuracy. It seems therefore important for folksonomy-derived similarity measures to capture this form of social information, which differs from the more obvious notions of similarity based on shared features. Indeed we show there is actionable information in annotation data even if we do away with tags when computing resource similarity and vice-versa (i.e., removing resources when computing tag similarity).

Similarity measures must also take into account runtime complexity. An effective similarity measure must accurately quantify relationships efficiently. In § 5.4.2, the case for efficiency is clear. The similarity measures with global dependencies are inefficient.

The state of the art for scalable collaborative similarity measures for social annotation systems is set with maximum information path (mip). Our evaluation shows that among scalable measures, the new mip similarity measure outperforms all others in accuracy. These results are consistent across tags and resources, and across datasets from two different social tagging sites. Furthermore, mip generalizes to the hierarchical representation. When maximum information path with collaborative aggregation is not the best in reference dataset approximation, mutual information with distributional aggregation is the best at reference dataset approximation. The added complexity from the computation of attribute joint probabilities necessary in mutual information (cf. § 5.2.3) and the timing results (cf. Figure 5.3) must be taken seriously. Maximum information path requires only to compute the minimum attribute probability, leading to linear space and time complexity.

7

Applications of Social Semantic Networks

We will now present three applications primarily based on socially induced semantic networks. These applications are tag recommendation (§ 7.1), resource (link) recommendation (§ 7.2), and a Web search interface that utilizes a map of query results (§ 7.3). In § 7.4, we identify and analyze six features for spam detection. One feature, *TagBlur*, is based on the social similarity among tags. The other five features address other aspects of spam detection in social tagging systems.

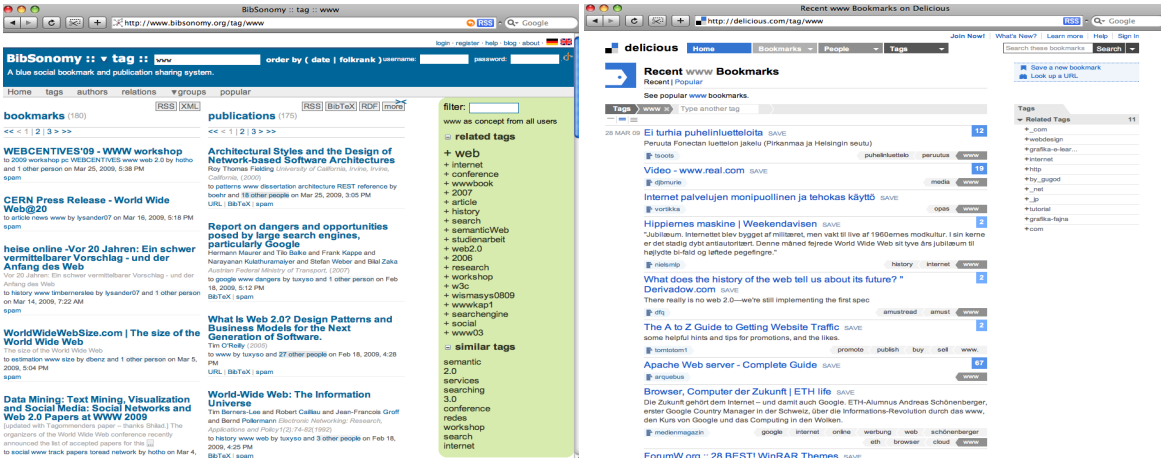


Figure 7.1: Examples of tag relations by `BibSonomy.org` (left) and `delicious.com` (right). Both social bookmarking tools place the related tags along the right hand column.

7.1 Tag Recommendation

An obvious application of social similarity (§ 4) is a recommendation system. In other words, given a query item (resource, tag, or user), retrieve all items ordered by similarity to the query. Here, we will explore the prediction of tag relations. An effective measure will be able to predict, or recommend related tags. This has many uses. For example, when a user is in the process of tagging a resource, the system can suggest tags to help the user organize or classify the resource. Other useful applications of tag recommendation are tag navigation, query expansion, and automatic tag assignments. These processes can lead to search engines aware of keywords related to a resource that do not appear in the document [SZL+08]. We illustrate two prominent applications that use tag relations in Figure 7.1.

The analysis of tag recommendation is based on how well our tripartite similarity measures (cf. § 4.2) predict user defined relationships among tags [MCM+09b]. `BibSonomy.org` allows users to input directed relations such as `tagging` → `web2.0` between pairs of tags. These relationship are suitable for, but not limited to, defining *is-a* relationships.

The semantics can thus be read as "`tagging` is a `web2.0`" or "`tagging` is a subtag of `web2.0`" [HJSS06a]. The most straightforward evaluation of our similarity measures therefore consists in using them to predict user-defined relations between tags. Such a prediction task requires that we set some threshold on the similarity values, such that a similarity above threshold implies a prediction that two tags are related and vice-versa. To determine which similarity measures are more effective predictors, we plot in Figure 7.2 for BibSonomy and Figure 7.3 for GiveALink the ROC curves. The corresponding areas under the curves are in Figure 7.4.

The results suggest that distributional aggregation does not provide any increase in performance compared to projection. Macro aggregation consistently underperforms the other methods, but experiences a boost with collaborative aggregation. These results also suggest that mutual information outperforms the other measures with distributional aggregation. These results are roughly consistent with those discussed in the indirect evaluation (§ 6.2.1).

The ROC analysis highlight some limitations with this evaluation approach:

- The available user data about tag relations is very sparse. For example we considered $2,000$ BibSonomy tags (4×10^6 candidate relations) and among these we found only 142 tag relations provided by BibSonomy users. For the GiveALink tag prediction evaluation, we considered $2,335$ tags (5.4×10^6 candidate relations) and among those we found 70 tag relations. With such little labeled data, assessments are bound to be extremely noisy.

- Similarity values are broadly distributed, spanning several orders of magnitude. The tag relation prediction task forces us to turn this high resolution data into binary assessments, potentially losing precious information. The results are very sensitive

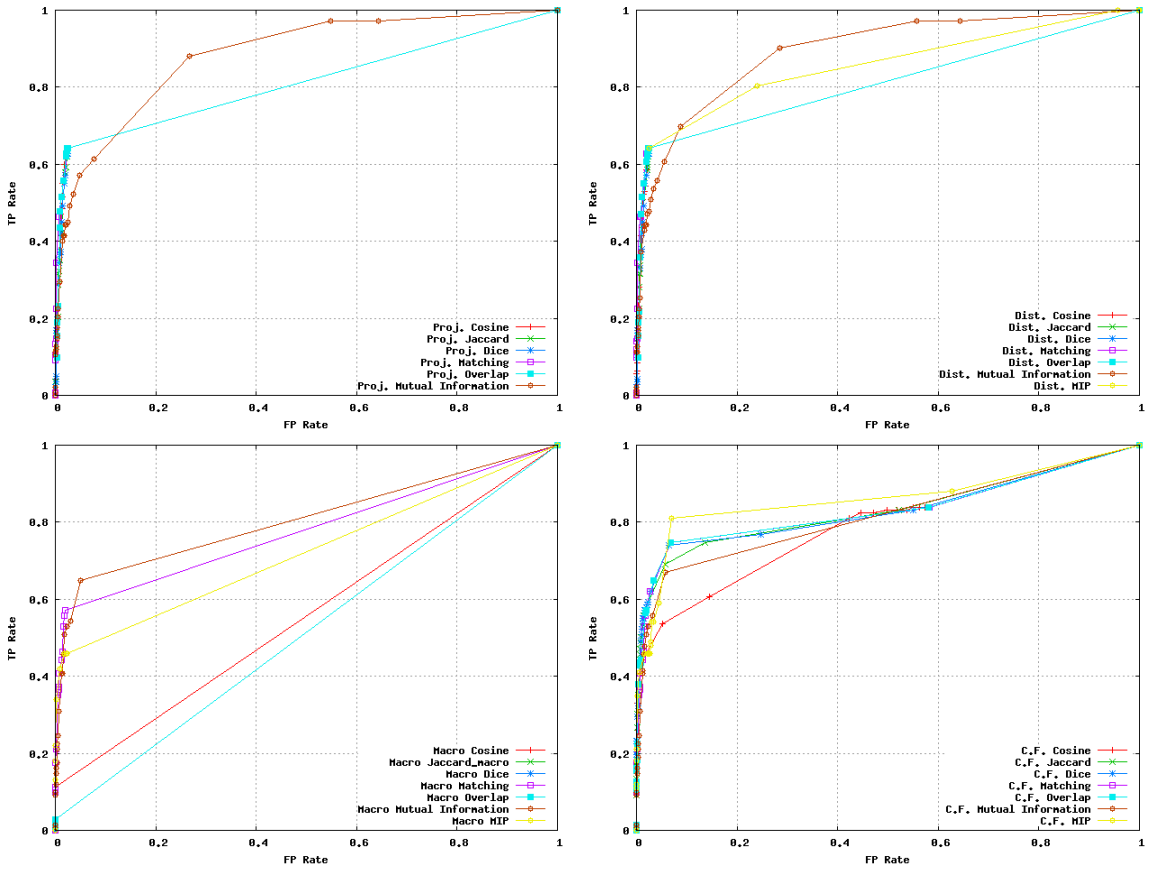


Figure 7.2: ROC curves for tag relation predictions based on similarity measures with BibSonomy tags. In this and the next figure, the plots are ordered according to the aggregation method starting with *projection* in the upper left, *distributional* in the upper right, *macro* in the lower left, and finally *collaborative* in the lower right. The true positive rate is plotted against the false positive rate. Similarity values for different measures are normalized to the unit interval. As the similarity threshold is lowered, both false and true positive increase. A good similarity measure can select many true positives with few false positives, yielding a higher ROC curve. Because of the nature of maximum information path, this measure is evaluated only in the distributional, macro, and collaborative cases.

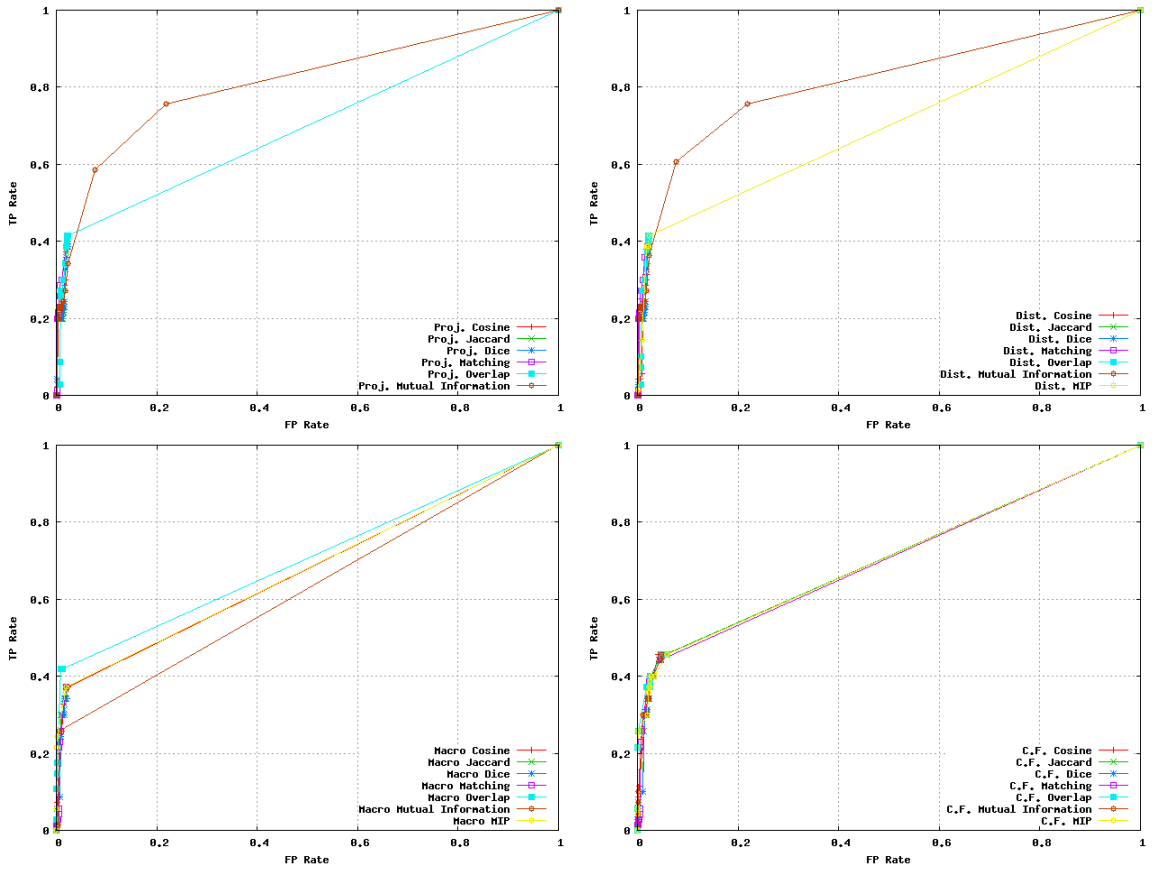


Figure 7.3: ROC curves for tag relation predictions based on similarity measures with GiveALink tags. The plots are ordered as in Figure 7.2.

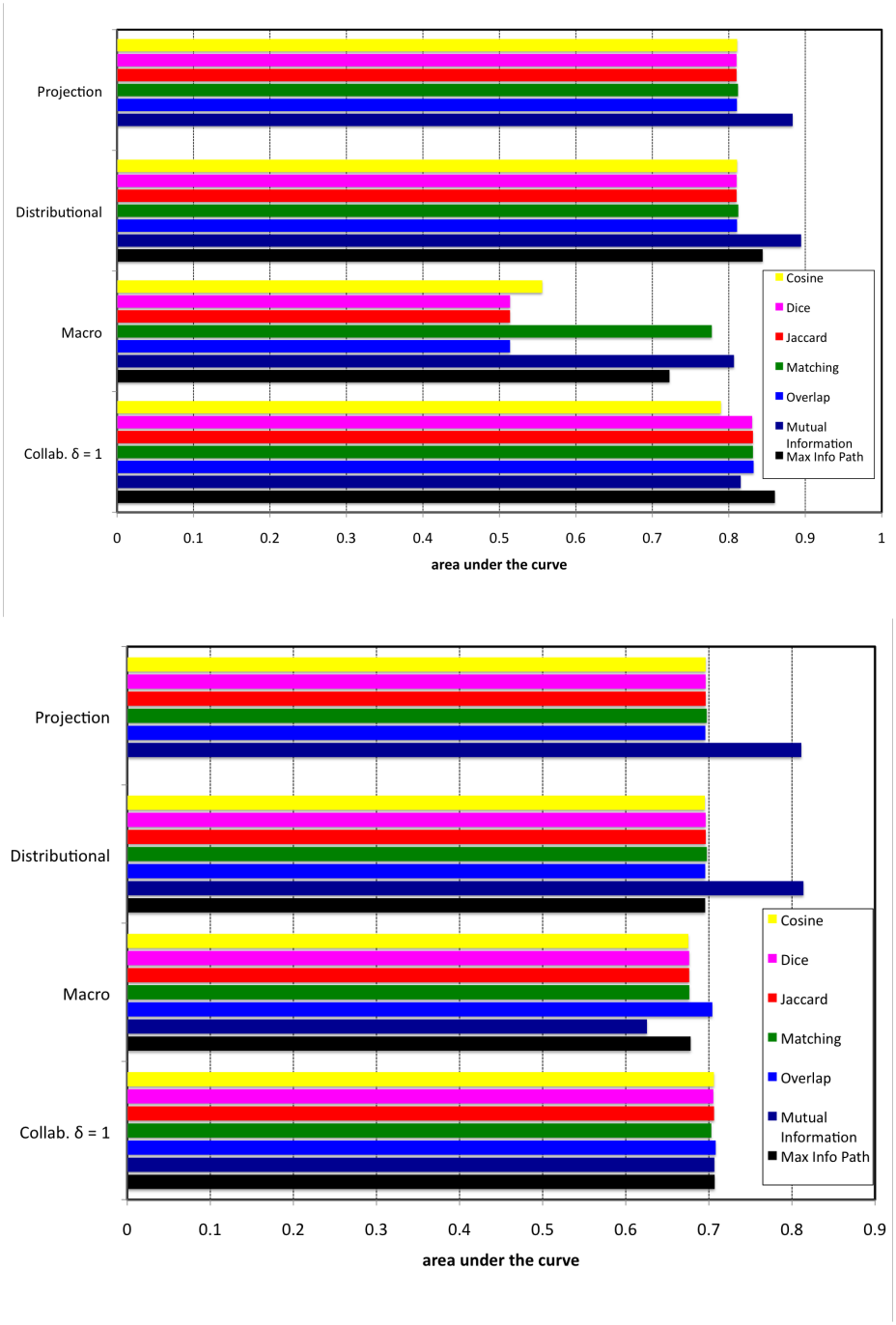


Figure 7.4: The area under the ROC curves (AUC) for tag relation predictions based on the similarities between BibSonomy tags (top) and GiveALink tags (bottom). Because of the nature of maximum information path, this measure is evaluated only in the *distributional*, *macro*, and *collaborative* cases.

to small changes in the similarity threshold; for example when considering BibSonomy's tags increasing the threshold from 0 to 10^{-7} decreases the false positive rate from 1 to less then 0.1. Such sensitivity suggests that fine-grained information is critical, and negatively affects the reliability of the evaluation results.

- Finally, although there is no such requirement, users' tag relations usually focus on hierarchical relationships and thus may miss many potentially strong non-hierarchical relations. For example, we may have `python` → `programming` and `perl` → `programming` but no relation between `python` and `perl`. This may unfairly penalize our measures.

The reference similarity approximation analysis (cf. § 6) effectively addresses these limitations, and are consistent with the results of Figure 6.2.

7.2 Resource Recommendation

Resource recommendation is a pivotal tool in `GiveALink.org`. Figure 7.5 is a screenshot of this application. Here we present the results from a user study conducted to evaluate the hierarchical similarity measure Lin (cf. § 4.1.1) for resource recommendation. This type of recommendation has many uses with the most obvious being given a link, recommend related links. For example, if a user is looking to travel and is only aware of `expedia.com`, the traveler would benefit from a tool that recommends alternate travel sites. Another example is a user reading a news article on `reuters.com`. Perhaps the reader can benefit from a completely different perspective of a related article at `foxnews.com`. It is highly unlikely that two commercial sites will directly link to one another, but a recommendation engine based on user collaboration can reveal *social* links.

For the user study, each subject submitted query URLs and determined whether each resulting URL was relevant or not. From the data collected, precision and recall for each

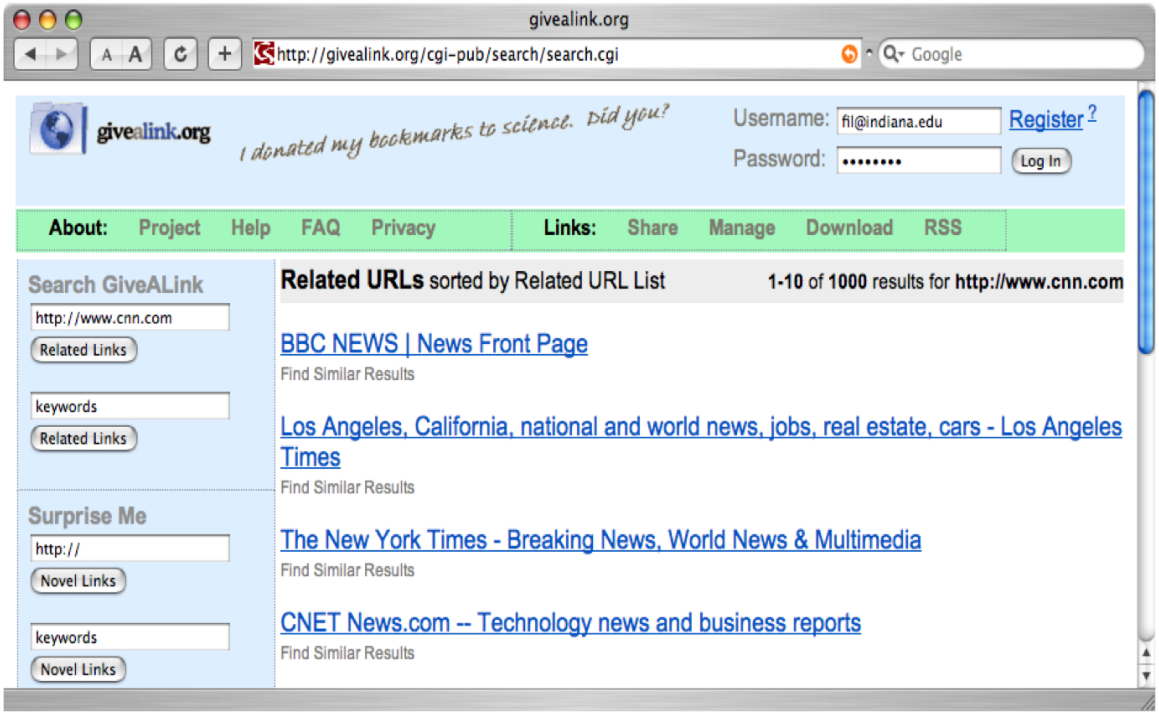


Figure 7.5: A screenshot of the GiveALink recommendation system. Given the query `www.cnn.com`, GiveALink retrieves a set of related resources based on socially induced relationships.

rank were calculated and averaged across all queries. We included Google's `related` service (`google.com/help/features.html`) in the study to gauge the performance of our collaborative filtering and ranking techniques. Our intention is not to suggest a direct competition with traditional search engines, but rather to set a context in which to interpret the performance of our system. One could incorporate the GiveALink similarity measure to improve the richer ranking algorithms employed by search engines.

Figure 7.6 presents the results of the user study normalized for the relative difference in size between the two systems. At the time of the experiment, GiveALink had less than $30,000$ URLs, which was at least five orders of magnitude less than the number of pages indexed by Google. To factor out the effect of such different coverage on performance, we restricted the relevant sets to only include URLs that, in addition to being deemed

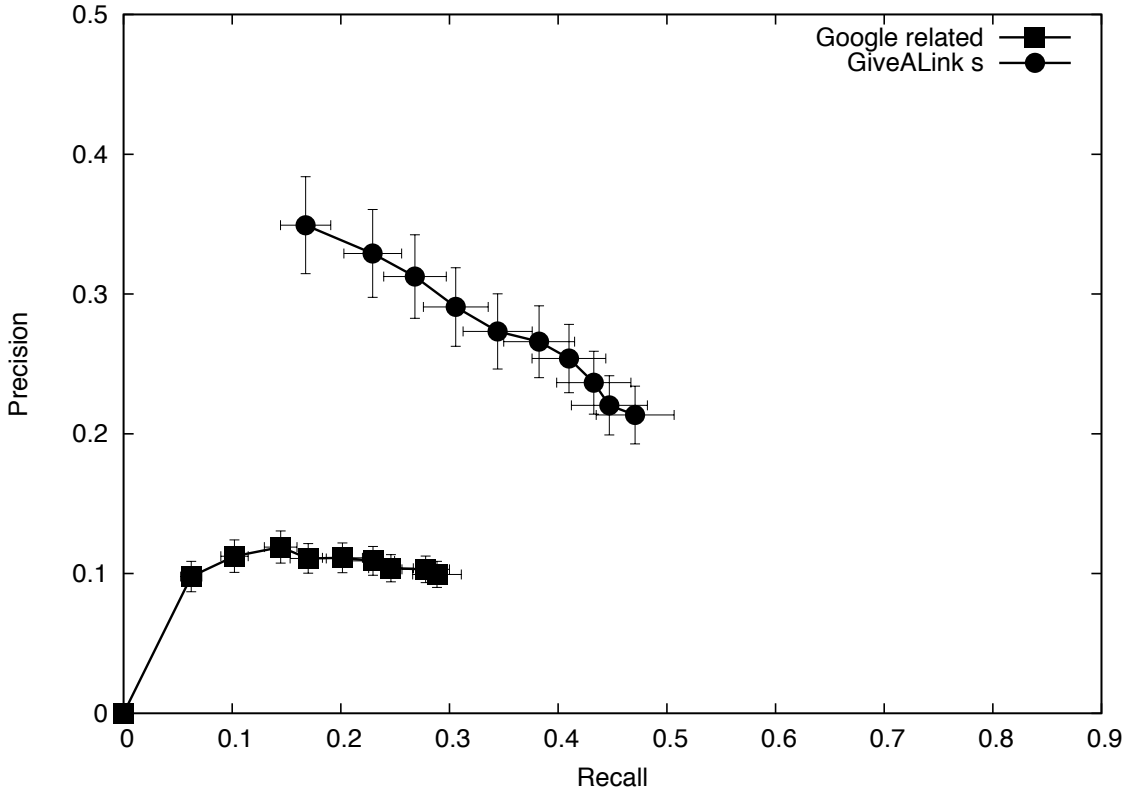


Figure 7.6: Precision-recall plots for our user study. The data is based on 86 subjects who submitted 336 query URLs and were asked to identify "relevant" pages. Error bars correspond to ± 1 standard error in precision and recall, micro-averaged across all user queries for each rank. The reason why Google's curve starts at the origin is that for all user queries Google returned a URL in the first position that was deemed irrelevant. In most cases, this was the query URL itself. The plot's relevant sets include only URLs in *both* GiveALink and Google.

relevant, appear in *both* GiveALink's and Google's indices. Under the reasonable assumption that GiveALink and Google contain independent samples of the Web, this leads to a comparison that normalizes for the relative sizes of the systems to focus on the quality of the ranking functions. In this view, GiveALink's similarity measure seems to outperform Google.

7.3 Web Navigation

We now leverage the resource semantic similarity network to study *exploratory search*. This type of search serves a different need than the conventional 'lookup' search. Exploratory search is defined as the goal of one wanting to learn or investigate a certain topic. For example, consider a user looking to learn about fuel efficient cars. After submitting a query to a search engine, one may profit more from an interface exposing the interrelationships between results as opposed to a ranked list. A map can visualize these relationships. Results can be grouped according to type (hybrid or compact), fuel mileage, etc. We will use social semantic similarity to build relationships and to determine the results proximity to one another in the visualization. Here we explore the semantic similarity between objects for a map interface in an exploratory search modality. The results show that users find relevant information faster with the help of a semantic map.

7.3.1 Network Visualization

We used the social bookmarking site `GiveALink.org` as an environment in which to experiment with the visualization of social semantic networks in exploratory search. At the time of our experiment, the technique described in § 4.1 was used for establishing resource relationships. The semantic similarity network supported the retrieval of pages related to a given URL query. To handle keyword queries, GiveALink parsed and indexed annotations — tags, categories, titles and descriptions — from the user profiles. Results were retrieved by matching the query to the indexed keywords. GiveALink then computed a *relevance* score by combining the number of keywords matching the query with the *strength* of each page. The strength of a node, defined as the sum of the similarities associated with its incident edges, is a measure of the node's centrality in the network as described in § 8.2.1. A central page tends to be of general interest, being related to many

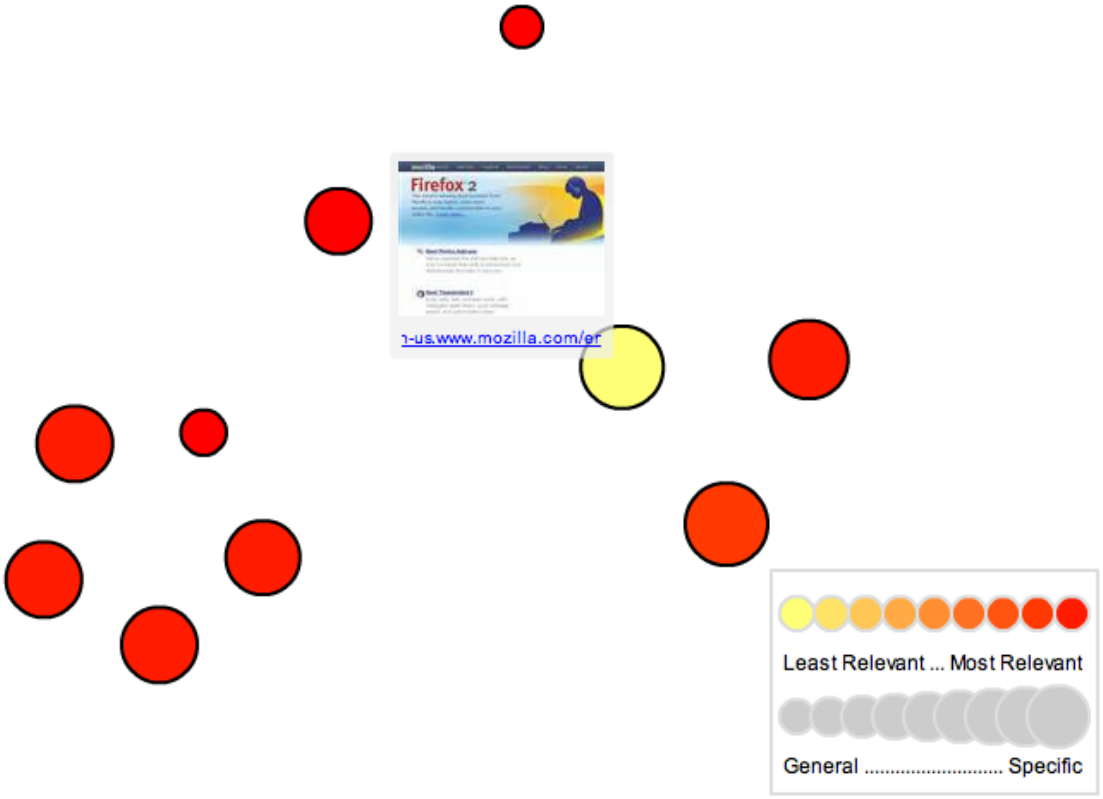


Figure 7.7: The network visualization applet.

others. For our experiment we limit the result set to the top ten relevant resources per query.

We designed and developed a visualization applet in Flash to render search results as a network. Given a query, the applet retrieves an XML formatted result set from a GiveALink Web service `GiveALink.org/main/download` and displays each document as a node in the map. The result set contains information about relevance scores, node strengths, and similarity among results. To visualize the social links between pages, their pairwise similarity is applied as a spring acting between the nodes according to a force

directed graph placement method [FR91]. The applet calculates the position of each document as a real time physical simulation of attracting and repelling forces.

Each result node is colored according to the relevance of the page using a simplified 'heat' color gradient from yellow to red. The coloration was chosen to continuously alter both the hue and saturation so as to be detectable by individuals with color vision impairments. The size of each node is also scaled according to its centrality; more central nodes are smaller. This was done to emphasize nodes that are more specific to the query topic.

Finally, page previews were made available by hovering the mouse cursor over each node, allowing for a 'details on demand' approach recommended by Schneidermann [Shn03]. This effect, along with the results of a typical queried Web page network, are visible in a screenshot of the visualization interface in Figure 7.7.

7.3.2 Experiment Setup

Visualizing a set of search results as a network is not a trivial rearrangement of a list in two dimensions. In addition to selecting and ranking the results, search engines also incorporate the display of additional information about each result. This typically includes, in addition to the title and link to the result, a summary or 'snippet' of the result text: a few descriptive sentences, or a relevant section of the page that relates to the query terms [LG98].

The inclusion of additional per-result information has become a standard feature for commercial search engines such as `google.com` and `yahoo.com`. Such a combination of ranked-list arrangement with extended result context has shaped the expectations of search engine users. However, the snippets of text limit novel arrangements of results, due to the difficulty of accommodating legible paragraphs of text in a dynamic two-dimensional layout. Furthermore, snippets have been shown to be helpful in informational tasks, but

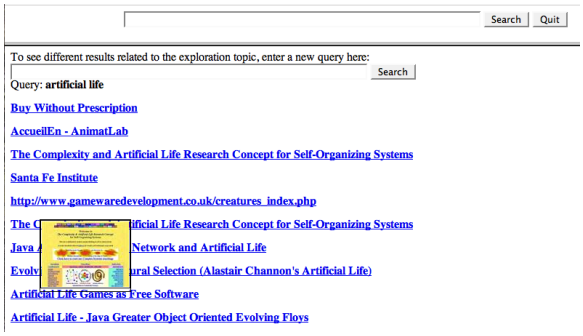


Figure 7.8: The list interface.

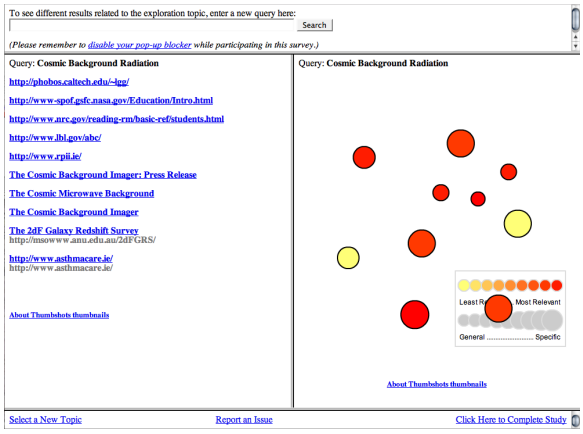


Figure 7.9: The hybrid interface.

degrade performance for navigation tasks [GC07]. Therefore, rather than attempting a direct comparison between the network layout and the conventional search engine display of results, we made an experimental design decision to remove snippets from all interfaces. We naturally expect this to affect the perceived utility of the interfaces, but feel it necessary to better focus the study on the issue of structural positioning of results based on social links, independently of the textual context that any interface might add.

Study participants were solicited through a call for participation distributed to departmental and IR interest mailing lists (Indiana University IRB study #07-12006). One of three search interfaces was selected at random and assigned to each participant. The interfaces included a 'map' interface (cf. Figure 7.7), a 'list' interface (cf. Figure 7.8), and a hybrid

interface combining the two (cf. Figure 7.9). We attempted to control the experiment as much as possible by providing the results in each of the interfaces with the same features: title text, hyperlink, and a pop-up preview of the result page. Relevance ranking was represented by the ordering in the list and by node color in the map. The title text appeared only upon hovering on the nodes in the map (along with the preview), providing for a weaker textual context than available in the list. On the other hand, the map had the additional context of the layout based on social links, and the result specificity represented by node size.

To evaluate these interfaces in the setting of an exploratory search task, each subject was asked to consider various topics from the following list:

- American presidential elections electoral colleges
- Alternative energy sources
- Artificial life
- Impressionism
- Partial differential equations
- Communism socialism fascism democracy
- Lung cancer
- Cosmic background radiation

The topics chosen reflect areas with adequate coverage in the corpus indexed by GiveALink and concerning domains deemed relatively unfamiliar to most participants.

We recorded relevant content discoveries as 'annotations.' We asked the subject to copy and paste helpful Web page content while they were viewing a resource linked from

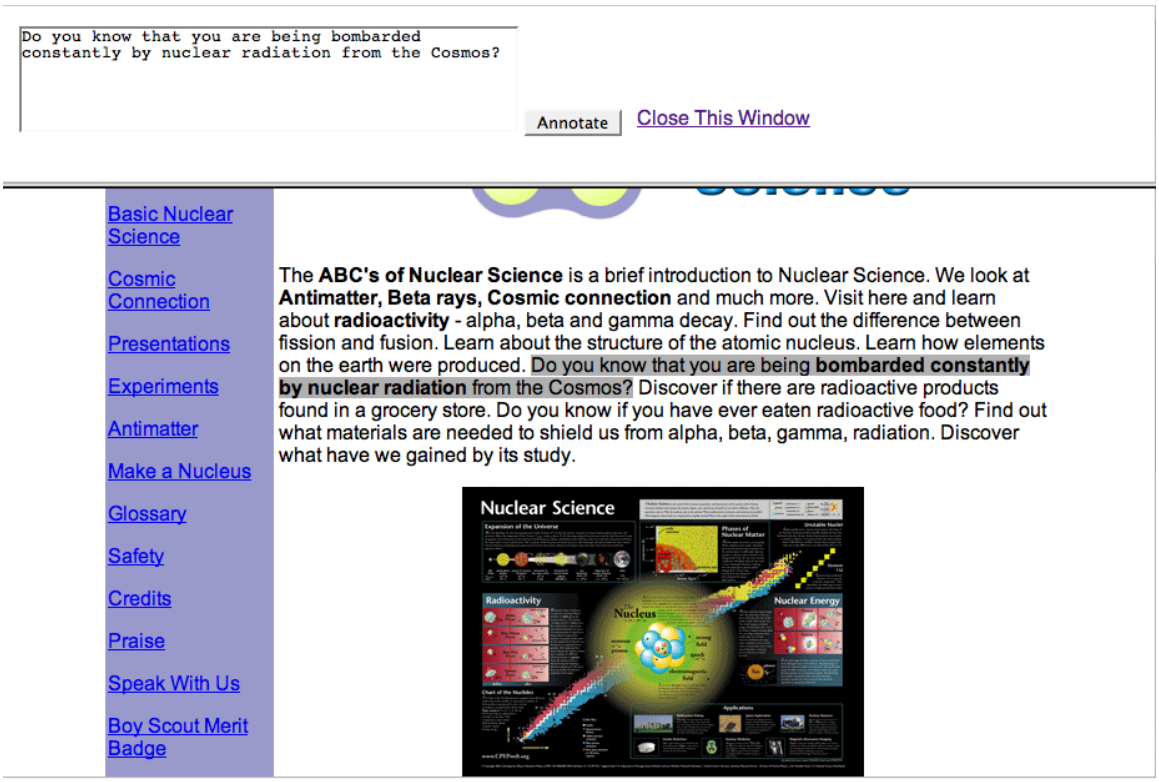


Figure 7.10: The annotation panel.

our interface. This content could be text, image links, or whatever they deemed to be important. Once a user navigated to a target Web page, two frames were presented, one containing the target Web page and the other a text box to record relevant information. Figure 7.10 illustrates this interface. At any time, the user could enter a new query in the assigned navigation interface to retrieve another result set for the current topic. When done exploring one or more of the topics, the user answered a brief exit survey before ending the study.

7.3.3 Results

We recorded each user's queries, URLs visited, and annotations made during topic exploration. We then compared groups of users based on which interface they were assigned. Specifically we looked at the number of queries performed per topic, the number of search results visited per query, number of annotations per topic, and the number of topics explored by each participant. We also asked them to rate the usefulness of the interface and the quality of the information in the search results. Overall, we collected results from 65 participants, providing a total of 219 queries and 161 annotations.

The experimental results in Figure 7.11 indicate that subjects using the hybrid navigation interface performed significantly fewer queries per topic than those using the list interface. On the other hand, no statistically significant differences were found in the number of annotations users provided per topic (Figure 7.12). Similarly, users followed approximately the same number of results per query and explored the same number of topics, regardless of which interface they used.

Therefore users of the hybrid interface were able to gather similar amounts of relevant knowledge using fewer queries. These findings suggest that those who saw both views of the results side by side (their social links in addition to their rankings) explored the topics more efficiently.

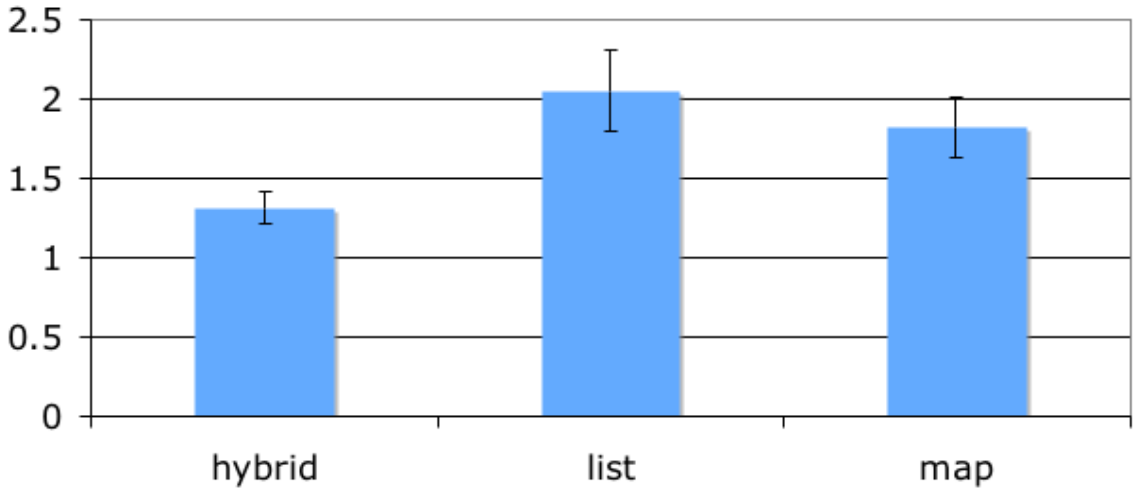


Figure 7.11: Number of queries submitted by the average user per topic, including the original topic query. Users who explored more than one topic contribute multiple data points. The error bars in this and the following charts represent ± 1 standard error around the mean. The number of queries for the hybrid group is significantly lower than for the list group with $p = 0.03$. We cannot statistically differentiate the other pairs ($p = 0.15$ for hybrid vs. map and $p = 0.40$ for map vs. list).

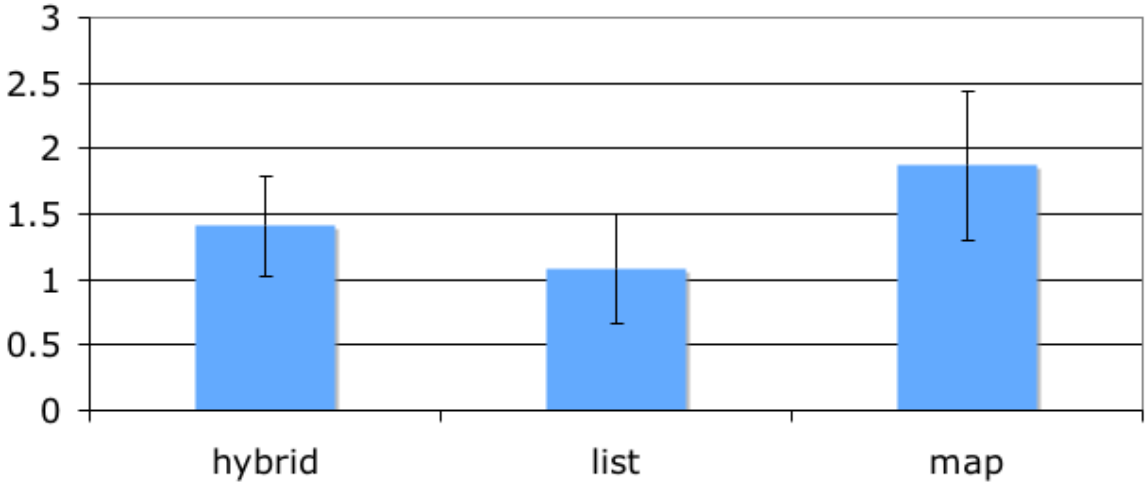


Figure 7.12: Number of annotations submitted per user, per topic. Users contribute a data point for each topic they explored. There are no statistically significant differences between the means ($p > 0.7$).

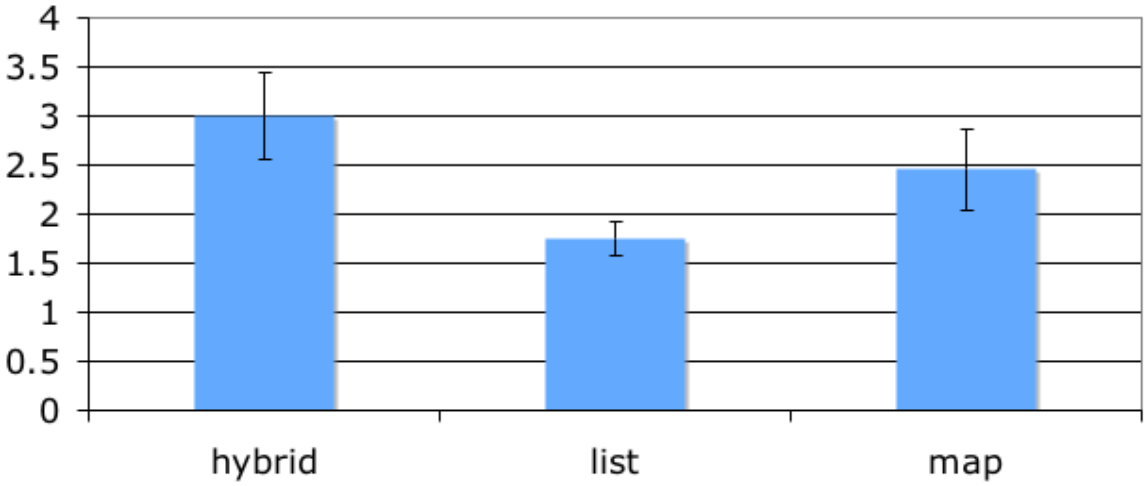


Figure 7.13: Ratings by group when asked about how much the interface helped exploration of the topics. The hybrid interface was rated more highly than the list interface ($p = 0.024$). The ratings for the map interface cannot be statistically differentiated from either the hybrid or the list interfaces ($p = 0.3$ and $p = 0.13$ respectively).

Among users who submitted only one query, ratings varied from the strongly negative to the strongly positive for all interfaces. Those who interacted more with the interfaces had more consistent responses. For evaluating the closing questionnaire we chose to include only results from users who performed more than one query. This ensures that the qualitative evaluations of the subjects were based on a baseline level of experience with the interface. The result of this decision is a reduction in the variance of likert-scored results, without a statistically significant change in mean.

When asked about the usefulness of the interface for the task, users in the hybrid group gave significantly higher ratings than those in the list group (Figure 7.13). Thus, it appears that the addition of the visualization of the social links between search results assists in the exploration task.

The rating for the quality of information is similar across all interfaces (Figure 7.14). This indicates that the interfaces alone did not affect the user's perception of the search result quality — indeed the results presented were identical.

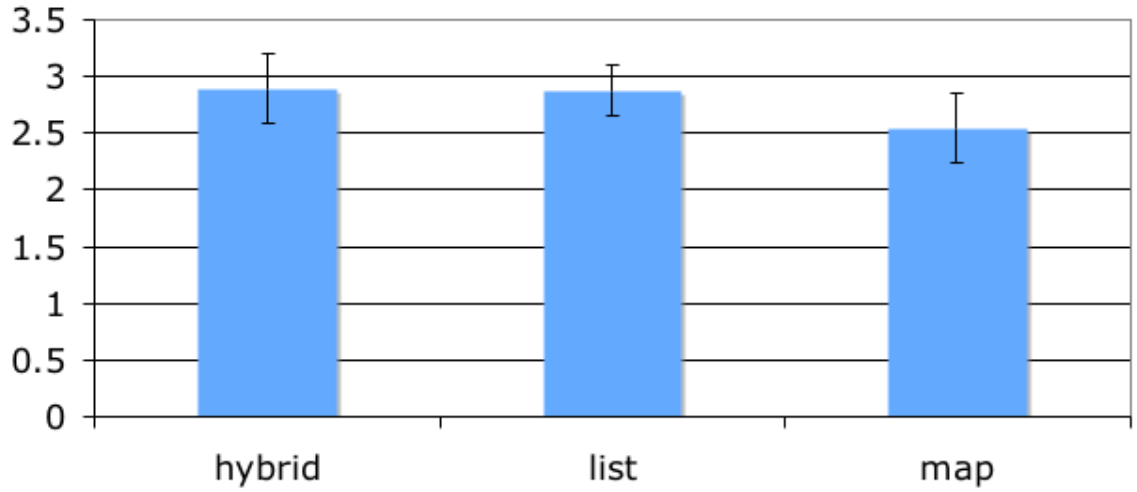


Figure 7.14: Ratings by group when asked about the usefulness of the data. The differences are not statistically significant ($p > 0.4$).

7.3.4 Discussion

The results are promising. By comparing the interfaces solely on the basis of how they lay out the results, in the absence of text snippets, we find that participants prefer a hybrid approach. Furthermore, the hybrid method is more 'efficient' in that users generate the same number of annotations with significantly fewer queries. Our findings for the exploratory search task are consistent with those of prior information visualization research applied to conventional 'lookup' retrieval tasks.

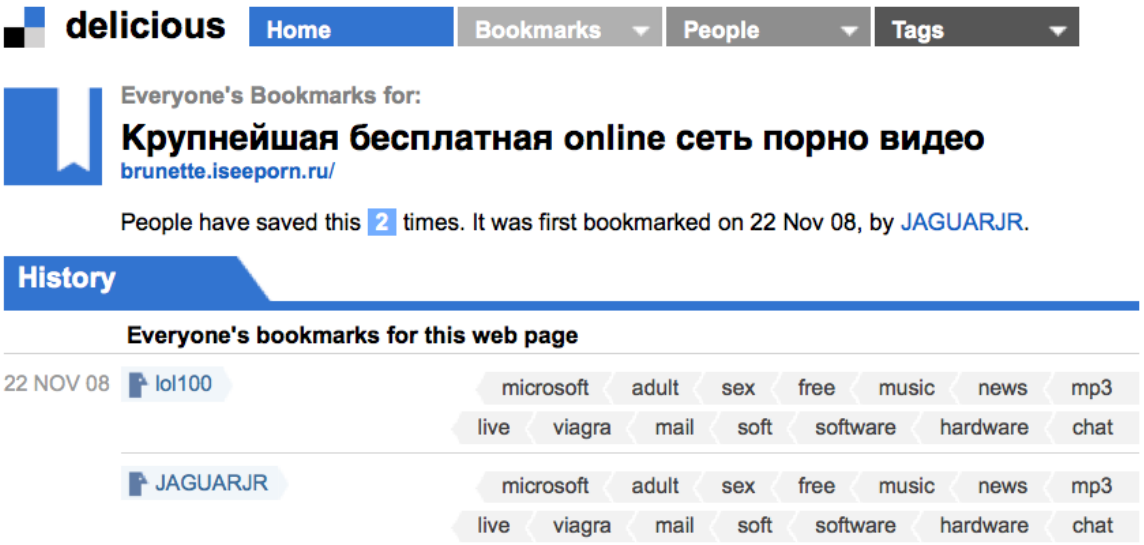


Figure 7.15: Illustration of social spam. In this example we see a spammer using two identities to post a Russian porn site on the popular social tagging site `delicious.com`. The spammer uses popular tags such as "music," "news" and "software," which are obviously unrelated to the Web site and to each other.

7.4 Social Spam Detection

The popularity of social bookmarking sites has made them prime targets for spammers [MCM09a]. Since every user can easily add to the folksonomy, the structure of the graph is entirely user-driven, and a malicious user can exploit this control to make some content more prominent, drive user traffic to chosen targets, and in general pollute the folksonomy. We refer to these kinds of exploitations of collaborative annotation systems as *social spam*. Identifying social spam automatically and efficiently is a key challenge in making social annotations viable for any given system and for the Web at large. Many collaborative tagging systems require an administrator's time and energy to manually filter or remove spam. Figure 7.15 shows a typical example of social spam on the popular collaborative tagging system `delicious.com`. Here we present a study of automatic spam detection in a social bookmarking system. We make the following contributions:

- We discuss the main motivation and incentives behind social spam, relating it to the phenomena of click fraud and Web pollution. (§ 7.4.1)
- We identify six features of collaborative tagging systems capturing various properties of social spam. (§ 7.4.2)
- We analyze how the proposed features can be used to detect spammers, showing that each feature has predictive power for discriminating spammers and legitimate users. (§ 7.4.3.1)
- We evaluate various supervised learning algorithms that use these features to detect spam, and show that the resulting classifiers achieve accuracy above 98% while maintaining a false positive rate near 2%. Further feature selection analysis reveals that the best performance is obtained using all six features. These promising results provide a first baseline for future efforts on social spam detection. (§ 7.4.3.2)
- We make our dataset publicly available to stimulate further open research on social spam detection.

7.4.1 Motivations of Social Spam

The first step toward the design of effective measures to detect and combat social spam is an understanding of the motivations behind it. Based on our experience as well as judging from past history of spam in other contexts, we argue that the most threatening motivation is financial gain. How can someone make money by abusing social tagging systems? This question has not yet been thoroughly explored. The spammer probably makes money when a user visits site X, and therefore the spammer needs to attract users to site X. Social spam is a cheap way to attract users. Other methods include email spam, search engine manipulation, and placing ads. The first is more expensive because there is already an

infrastructure in place against email spam: filters, black lists, and so on. Search manipulation is more expensive because search engines have a financial interest in preventing rank manipulation and thus invest in spam detection algorithms. Finally, advertising has obvious monetary and disclosure costs. Social tagging systems are therefore a target of opportunity; an abuser can submit many spam annotations effectively, efficiently, cheaply, and anonymously.

Once the spammer has attracted users to site X, the easiest and most effective way to make a profit is to place ads. The widespread adoption, low entrance cost, and ease of use of advertising platforms and networks such as Google AdSense (`google.com/adsense`) and Yahoo APT (`apt.yahoo.com`) have created a market for online traffic and clicks. Abuse is therefore to be expected. All the spammer needs to do is create some content, place ads, and use social sites to attract traffic. Much of this can be done automatically, and therefore cheaply. For instance tools are available to identify a set of keywords that, if used to tag the target website, are likely to generate traffic to it. Such tools include Google Trends (`trends.google.com`) and Google Keyword Tool (`adwords.google.com/select/KeywordToolExternal`). It is important at this point to briefly discuss the relationship between social spam and click fraud.

Advertising networks and keyword tools are legitimate when used as intended. If a user tags with helpful keywords a legitimate site containing ads, this is not a case of spam. We consider social spam only those abusive uses of social tagging in which misleading tags are used, and/or a fraudulent or malicious site is tagged. Examples include phishing sites, pages that install malware, and so on. Click fraud may occur when a site uses ads in a way that is inconsistent with the terms of service of the ad provider, for instance simulating clicks to generate revenue or harm a competitor. However, there is a large gray area between the extreme of blatant click fraud (which an ad network would censure) and the extreme of legitimate use. This gray area includes target sites with fake or plagiarized

content, whose exclusive purpose is to draw profit from ads. Such sites may not violate the advertising terms of service. Yet a reasonable user would consider annotations that lead to them as social spam.

Content can be manipulated to attract users to ads. From the spammer's perspective, there is no need for text to be coherent or meaningful. The only important criterion is for content to attract visitors. To this end one would generate text that contains popular keywords and that looks genuine, at least to search engines' crawlers and content analysis algorithms. The expression "original content" is sometimes used to refer to text that *looks* original enough to trick a search engine into thinking it is a genuine page written to convey information — as opposed to its hidden, exclusive purpose of attracting clicks to revenue-generating ads.

There are at least three ways to generate "original content." One is to hire cheap labor. Another approach is automated generation of text via natural language techniques. The third approach is to plagiarize the content from legitimate sources such as Wikipedia — sources abound. There is already a marketplace for such services; questionable sites like `adsenseready.com` allow users to download pre-made "content-rich" sites on which to place ads.

In January 2006 a security mailing list circulated a message by Charles Mann, a writer who had authored a piece on click fraud for *Wired* magazine. Mann reported on a letter that he received in response to his article. The anonymous letter writer told his story as a former author of "original content." The story detailed how a disreputable company developed software to automatically create fake Web sites to capitalize on Google AdSense. In particular, the software would automatically: (1) find relevant keywords using suggestion tools such as those mentioned above; (2) register domain names based on the keywords; (3) create hosting accounts for those domain names; (4) create complete websites full of bogus content using keywords in generic sentences, with embedded ads; (5)

upload these websites to the hosting accounts for the corresponding domain names; and (6) cross-link the websites to boost PageRank. The software allowed the creation of hundreds of sites per hour, each with hundreds of pages, generating significant revenues from advertising. This testimonial clearly demonstrates that there is both an incentive and an industry to generate online junk, and consequently to promote it through social spam.

As a demonstration of the ease with which revenue can be generated this way, we wrote a simple script that generates a dynamic websites for a fake "Gossip Search Engine" (`homer.informatics.indiana.edu/cgi-bin/gossip/search.cgi`). When the user submits a celebrity name, the site acts as if it had searched for news about the query. In reality, a fake news item is created on the fly with a grammar-based text generation algorithm (`dev.null.org/dadaengine`). Additional news and images are obtained from RSS feeds and search APIs. Although the page seems to provide original content, the information presented is either fake or stolen. The script simulates the experience of navigating through pages with different content by linking back to itself with keywords scraped from other sites. The ads on the generated pages are often relevant to the contextual information (celebrity names and gossip keywords). As a result the demo generates revenues.

It is important to ask, *Who is harmed by spammers who generate fake content?* The advertiser gains, because real users click on ads and thus visit the advertiser's page, which is the desired outcome. The intermediary gains its fees from the advertiser. The publisher (spammer) of course gains its cut from the clicked ads. And one might argue that if a user clicks, he/she was ultimately interested in the ad, thus no harm is done. Paradoxically it may seem there are no victims. This leads to a lack of incentives for the intermediary to curb this kind of abuse. In reality, when bogus content is generated to play the system, good information resources become diluted in junk and become harder to be found.

Search engines and folksonomies direct traffic in the wrong directions. Information consumers end up with less relevant or valuable resources. Producers of relevant resources receive less cash as a reward (lower click-through rate) while producers of junk receive more cash. One way to describe this is *pollution.* Virtual junk pollutes the Web environment by adding noise. Everybody but the polluters pays a price for Web pollution: search engines work less well, users waste precious time and attention on junk sites, and honest publishers lose income. The polluter spoils the Web environment for everybody else.

The above discussion provides us with a clear financial motive for social spam. Understanding the incentives for social spam is essential to the design of effective countermeasures. In the remainder of this paper we describe a detection approach whose key ingredients are a set of features directly inspired by such insight.

7.4.2 Features

The first issue to address when we set out to design a social spam detection system is what class of object should be seen as a potential candidate for spam labeling. Broadly speaking, spam can be injected into a tagging system at three different levels. The traditional view is to classify Web pages or sites as spam based on their content, that is, resources that users of the system perceive as non relevant or as "junk." The problem with this perspective is its subjectivity: what is spam to one person can be interesting to another. Second, we can think or spam posts, i.e., malicious associations between resources and tags. Associating a spam resource with many and/or popular tags creates pathways that lead to that resource. This type of spam pollutes the system by creating artificial links between resources and tags that would be otherwise unrelated to one another. This kind of pollution thus affects the measures of tag and resource similarity that are grounded in the social annotations, altering recommendation, ranking and search mechanisms in the

folksonomy. Finally, one can look at user accounts created with the goal of injecting foreign content into the system. Such accounts may or may not mix legitimate content with spam, in order to mask the spamming activity. Flagging users as spammers is the approach taken by some social tagging system, such as BibSonomy. This approach is intuitive and easy from an administrator's point of view, but it uses a broad brush. It may be exceedingly strict if a user happens to post one bit of questionable content on the backdrop of otherwise legitimate annotations.

When detecting spam, one can focus on each of the different levels mentioned above, and design features that selectively target spam resources, spam posts (tag-resource associations), or spammer users. Our opinion is that the most appropriate level of resolution for classifying spam is often that of a *post.* When a resource's content is hard to classify, observing how a user annotated it should reveal the user's intent. Consider for example the site of Figure 7.15: different users may disagree on whether the *resource* itself is spam, but any reasonable user would recognize the *posts* as spam based on the misleading tags used. Conversely, appropriate tags might suggest legitimate annotation of a questionable resource. On the other hand, we can "escalate" spam labels from posts to users when necessary, by aggregating across the posts of a user. Next we define six different features designed to capture social spam according to several criteria. Two of the features operate at the level of posts, three at the level of resources, and one at the level of users. For each feature we discuss the underlying strategy, the technique used for computing it, and the complexity entailed by such computation.

Let us repeat the definition of a folksonomy (cf. § 4.2) for the feature descriptions. A folksonomy F is a set of *triples*. The triple representation is widely adopted in the Semantic Web community [HJSS06b]. Each triple (u, r, t) represents user u annotating resource r with tag t. A *post* $(u, r, (t_1, \ldots, t_n))$ can be represented as the set of triples $\{(u, r, t_1), \ldots, (u, r, t_n)\}$. Figure 4.4 illustrates an example.

7.4.2.1 TagSpam

A simple feature can be built on the notion that taggers share a prevalent vocabulary that they use to annotate resources [XFMS06]. Spammers may therefore use tags and tag combinations that are statistically unlikely to appear in legitimate posts. If a body of annotations labeled for spam is available, we can use this data source to estimate the probability that a given tag is associated with legitimate content. The support used to define this probability depends on the granularity of spam labels. In the present case we have labels on a per-user (spammer) basis (cf. 7.4.3). Let U_t be the set of users tagging with tag t ($U_t = \{u : (\exists r : (u, r, t) \in F)\}$) and $S_t \subset U_t$ the subset of these users labeled as spammers. We can then define the *TagSpam* feature via the probability that a tag t is used to spam, estimated by the fraction of users tagging with t who are spammers: $\Pr(t) = |S_t|/|U_t|$. These probabilities are then aggregated across a post's tags to obtain the *TagSpam* feature. Let us consider a user u who annotated resource r with a set of tags $T(u, r) = \{t : (u, r, t) \in F\}$. We define the *TagSpam* feature as

$$f_{TagSpam}(u, r) = \frac{1}{|T(u, r)|} \sum_{t \in T(u,r)} \Pr(t).$$

This feature can be computed in constant time for any new post entering the system, assuming that the tag probabilities are precomputed and that the number of tags in a post does not grow with the number of annotations in the system. However *TagSpam* suffers from a cold start problem, since a body of labeled annotations is needed to bootstrap the tag probabilities.

7.4.2.2 TagBlur

In spam posts, the spam resource is usually associated with a large number of popular tags that may be unrelated to the resource and are often semantically unrelated to one another.

This happens because spammers gain from associating the spam resource with a number of high-frequency tags, regardless of their meaning in the context of the folksonomy. This is one of the malicious behaviors described by Koutrika et al. [KEG+07]. We define a new feature that captures this degree of unrelatedness of tags belonging to a post, i.e., the "semantic blur" of the post. The underlying assumption is that spam posts tend to lack semantic focus.

To define a measure of semantic blur we need to build on a notion of semantic similarity for tags. To this end, we draw upon our previous work on social similarity [MRM08, MCM+09b], where a systematic characterization of measures for tag and resource similarity was carried out and validated against manually-created taxonomies of concepts and resources. In this prior work, we have shown that mutual information is one of the best measures of tag similarity (we omit the definition in this context for brevity, see [MCM+09b]). Let the similarity $\sigma(t_1, t_2) \in [0, 1]$ be defined as the mutual information between two tags t_1 and t_2, computed on the basis of a set of annotations (cf. § 7.4.3) and normalized into the unit interval. The *TagBlur* feature is obtained from a measure of distance (dissimilarity) between tags, averaged across all the pairs of tags in a post:

$$f_{TagBlur}(u, r) = \frac{1}{Z} \sum_{t_1 \neq t_2 \in T(u,r)} \frac{1}{\sigma(t_1, t_2) + \epsilon} - \frac{1}{1 + \epsilon}$$

where Z is the number of tag pairs in $T(u, r)$, ϵ is a small constant ensuring that the distance is defined (and large) when $\sigma = 0$, and the last term ensures that the distance is zero when $\sigma = 1$. The computation of *TagBlur* is quadratic in the number of tags per post. This can still be considered constant time if the number of tags in a post does not grow with the annotations in the system. This feature relies on the availability of a precomputed similarity for any two tags in the folksonomy (§ 4.2).

7.4.2.3 DomFp

Let us now focus on the *content* of annotated resources, by observing that Web pages in social spam often tend to have a similar document structure, possibly due to the fact that many of them are automatically generated by tools that craft Web sites from predefined templates. We want to estimate the likelihood that the content of a tagged page is generated automatically, by measuring its structural similarity to other pages from a body of annotations manually labeled as spam. We first extract a *fingerprint* of a page's DOM structure (hence the feature name *DomFp*). We strip away all the content but the HTML 4.0 elements (`w3.org/TR/REC-html40/index/elements.html`) and then build a string by mapping HTML elements to symbols while preserving their order of appearance in the page.

Assuming a body of labeled social spam, we can build a set K of fingerprints associated with spam resources. Each fingerprint $k \in K$ will have an associated frequency denoting the number of times that fingerprint is encountered in spam. Based on this frequency we can estimate the probability $\Pr(k)$ that k is associated with spam resources. The likelihood that a resource r is spam is then estimated by measuring the similarity between its fingerprint $k(r)$ and the spam fingerprints. To measure the similarity between two fingerprints k_1 and k_2 we turn to the shingles method [BGMZ97]. For each fingerprint sequence we build a set of shingles (10 symbols each), and define the fingerprint similarity $\sigma(k_1, k_2) \in [0, 1]$ as the Jaccard coefficient between the sets of shingles corresponding to the two fingerprints. Let us finally define the *DomFp* feature for resource r as the normalized weighted average:

$$f_{DomFp}(r) = \frac{\sum_{k \in K} \sigma(k(r), k) \cdot \Pr(k)}{\sum_{k \in K} \sigma(k(r), k)}.$$

We interpret *DomFp* as the likelihood that resource r is spam based on its document structure. This feature requires that each resource be crawled to extract its fingerprint. We also

assume that a labeled spam collection is available and spam fingerprint probabilities are precomputed. For each resource it is necessary to compute its similarity to all spam fingerprints, with complexity that grows linearly with the size of the labeled spam collection.

7.4.2.4 NumAds

Another resource feature, *NumAds*, draws upon the idea that spammers often create pages for the sole purpose of serving ads (cf. § 7.4.1). Ads are typically served from external resources through javascript. Since Google AdSense is the most popular ad service, we focus on it to illustrate our idea, by simply counting the number of times the Google ad server `googlesyndication.com` appears in a Web page tagged by a user. Let $g(r)$ be the number of ads in page r. We compute the *NumAds* feature as

$$f_{NumAds}(r) = \frac{g(r)}{g_{max}},$$

where g_{max} is a normalization constant. For evaluation purposes we set $g_{max} = 113$, which is the maximum value of $g(r)$ over all resources in our dataset (cf. § 7.4.3). Computing *NumAds* requires the complete download of a Web page.

7.4.2.5 Plagiarism

Our last resource feature, *Plagiarism*, shares with *DomFp* the goal of detecting automatically generated pages. Spammers can easily copy original content from all over the Web, as discussed in § 7.4.1. To estimate the odds that a page's content is not genuine, we look for authoritative pages that are likely sources of plagiarized content. We first extract a random sequence of 10 words from the page's content. This sequence is submitted as a phrase query (using double quotes) to the Yahoo search service API (`developer.yahoo.com/download`). We can measure *Plagiarism* by the number of results returned by the search

engine, excluding the originating resource's URL. Let $y(r)$ be the number of search hits different from r matching the phrase query extracted from r. We then define

$$f_{Plagiarism}(r) = \frac{y(r)}{y_{max}},$$

where y_{max} is a normalization constant. For evaluation purposes we limited the results from Yahoo to 10, and set $y_{max} = 10$. So $f_{Plagiarism}(r) = 1$ if $y(r) \geq 10$.

Plagiarism is the most expensive of our features. The page must be downloaded to extract the random sequence of words, then the search engine must be queried for a total of two network requests per resource. While each of the resource features requires a download, the extra request and the query limits of the Yahoo search service impose a larger burden on the computation of *Plagiarism*.

7.4.2.6 ValidLinks

Our last feature, *ValidLinks*, is defined at the level of a user and focuses on the detection of user profiles created for spam purposes. Many of the resources posted by a spammer may have questionable content (for example copyright infringing material along with ads) and be taken offline when detected by the hosting service. Other examples include malware or phishing sites. Malicious users may temporarily set up a page to obtain sensitive information, and then post the malicious resource to a social site. Once enough data is collected or the site is taken down by the hosting company, the resource disappears. Finally, a spammer may become inactive for various reasons and leave broken links in the social site. To capture these situations, we define *ValidLinks* as the fraction of a user's posts with valid resources,

$$f_{ValidLinks}(u) = \frac{|V_u|}{|R_u|},$$

where $R_u = \{r : (\exists t : (u, r, t) \in F)\}$ is the set of resources tagged by user u and $V_u \subset R_u$ is the subset of these resources whose links are valid. To determine the validity of a link we send an HTTP HEAD request. This must be done for each of a user's posts, making this feature expensive to compute for users with many resources.

7.4.3 Evaluation

Dataset Description

To evaluate the proposed features, both individually based on their discrimination power and together in support of a classifier, we need labeled examples to build training and test sets. We turn to a public dataset released by `BibSonomy.org` as part of the ECML/PKDD 2008 Discovery Challenge on Spam Detection in Social Bookmarking Systems (`kde.cs.uni-kassel.de/ws/rsdc08`). The dataset contains all the annotations of 27,000 users, of which 25,000 are manually labeled as spammers and the remaining 2,000 as legitimate users. Manual classification by a trusted moderator is one of the methods described in prior work [KEG+07]. In this particular case, the criterion is that if any one resource tagged by a user is judged to be spam, then the user is labeled as a spammer.

To perform our evaluation we sampled a subset of users from the complete dataset. The sample is random, except for two sources of intentional bias. One bias is to select an equal number of spammers and legitimate users. While such a ratio of spammers is not reflective of the original dataset, we have two reasons for this balance. First, from an evaluation perspective, spammers are so predominant in the BibSonomy dataset that a baseline classifier labeling all users as spammers would achieve over 92% accuracy, making it difficult to compare different features and algorithms. Second, in a realistic setting we expect that a social bookmarking site would have a spam defense mechanism in place, so that the density of spam in the system should not be so high. We also apply a bias such that users

with more posts in their profiles have a higher probability of being sampled. The resulting dataset comprises of a total of 500 users, 250 of whom are labeled spammers, with all their annotations.

Three features — *TagSpam*, *TagBlur* and *DomFp* — rely on statistics from the folksonomy, that is, we need a training set to compute them. We could split the dataset into a training set for this purpose and a test set, however this would create an imbalance with respect to the test set size for the other features. To keep the same test set size of 500 users for all features, we drew another independent sample from the original BibSonomy dataset using the same procedure. The annotations of these separate 500 users (again half spammers) were used to compute the values of *TagSpam*, *TagBlur*, and *DomFp* for the 500 users in our dataset. More precisely, to compute *TagBlur* we need at least one post with more than two tags, leaving us with 486 users in the original dataset with usable statistics. There are other feature requirements as well. Users without any valid resource links have *NumAds*, *DomFp*, and *Plagiarism* undefined. This yields 453 users with the *NumAds* feature defined. To extract a DOM fingerprint we need at least one crawled resource containing at least one W3C-standard HTML element, leaving 450 users with usable *DomFp* statistics. Only 446 users have at least one resource with text content allowing us to compute *Plagiarism*. Finally, all 500 users have *TagSpam* and *ValidLinks* features defined.

7.4.3.1 Feature Analysis

The labels in our dataset are at the level of users, therefore we must apply user-level spam detection to be able to evaluate our features and algorithms. One of our proposed features, *ValidLinks*, is a user-level measure. The other five features defined in § 7.4.2, however, are computed at the level of posts or resources, and we must aggregate them across each user's posts for evaluation. We considered various aggregation methods: average, minimum, maximum, product (sum-log), and variance. Among these approaches, the simple average

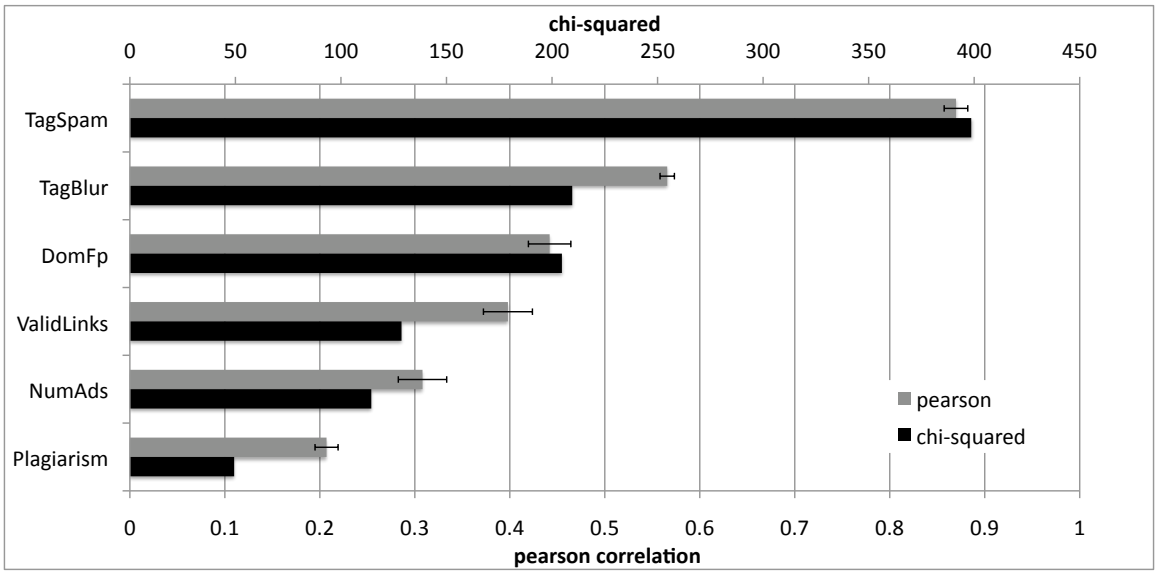


Figure 7.16: Discrimination power of our features with respect to spammers, as measured by Pearson's correlation and χ^2. For the former, error bars are computed by shuffling the contingency matrices.

of feature values across the posts/resources of a user provides the best results in terms of user discrimination power, which we report below. In some individual feature cases other aggregation schemes performed as well as averaging, but none was as effective across all features as the simple average:

$$f(u) = \frac{1}{|P(u)|} \sum_{(u,r) \in P(u)} f(u,r)$$

where $f(u,r)$ is the value of feature f for post (u,r) (or just for resource r) and the set $P(u)$ of posts by user u is defined as $\{(u,r) : (\exists t : (u,r,t) \in F)\}$ (and similarly for resources).

To analyze the discriminating power of each feature in separating spammers from legitimate users, we first normalized each feature such that $f(u) \in [0,1]$. We then divided the unit interval into 20 equal-size bins, building a contingency matrix $n(l,f)$ for the number of users with feature value $f(u) = f$ and spam label $l(u) = l$ ($l = 1$ for spammers, $l = 0$ otherwise). Finally we applied two standard discrimination functions, the χ^2 statistics and

Pearson's correlation. Figure 7.16 shows that both measures yield a consistent ranking of our six features by discrimination power. Not surprisingly, the three features that rely on statistics from the training set are the most effective; *TagSpam* is the best predictor — spammers do tend to use certain "suspect" tags more than legitimate users. All features have a statistically significant correlation with the spam label, therefore all features are predictive. These results are robust; we repeated the analysis on a separate, independent sample of users and found the same correlations.

For each feature we also show in Figure 7.17 and Figure 7.18 the histograms of spammers and legitimate users from the contingency matrices. These distributions give a visual intuition for how well each feature discriminates spammers. While some features, such as *TagSpam*, clearly lend themselves to being used in linear discrimination classifiers, others such as *ValidLinks* may also be good discriminators but would require the use of nonlinear classifiers; the distributions of spammers and legitimate users are not so easily separated by a simple threshold. This is evident if we use each feature alone in conjunction with a threshold to detect spammers. When we rank users by the values of each feature we obtain the ROC curves and AUC values shown in Figure 7.19. The ranking of the features by AUC is roughly consistent with the discrimination power of the features (Figure 7.16) except for those features that require a nonlinear classifier, which underperform in this simple linear detection setting.

We note that our top feature, *TagSpam*, achieves alone an AUC of 0.99, which compares favorably with the best classifier from the ECML/PKDD 2008 Challenge on Spam Detection in Social Bookmarking Systems: the winner of the challenge scored an AUC of 0.98 [GK08]. This is very encouraging, especially when considering that we use a smaller and more balanced sample of the dataset.

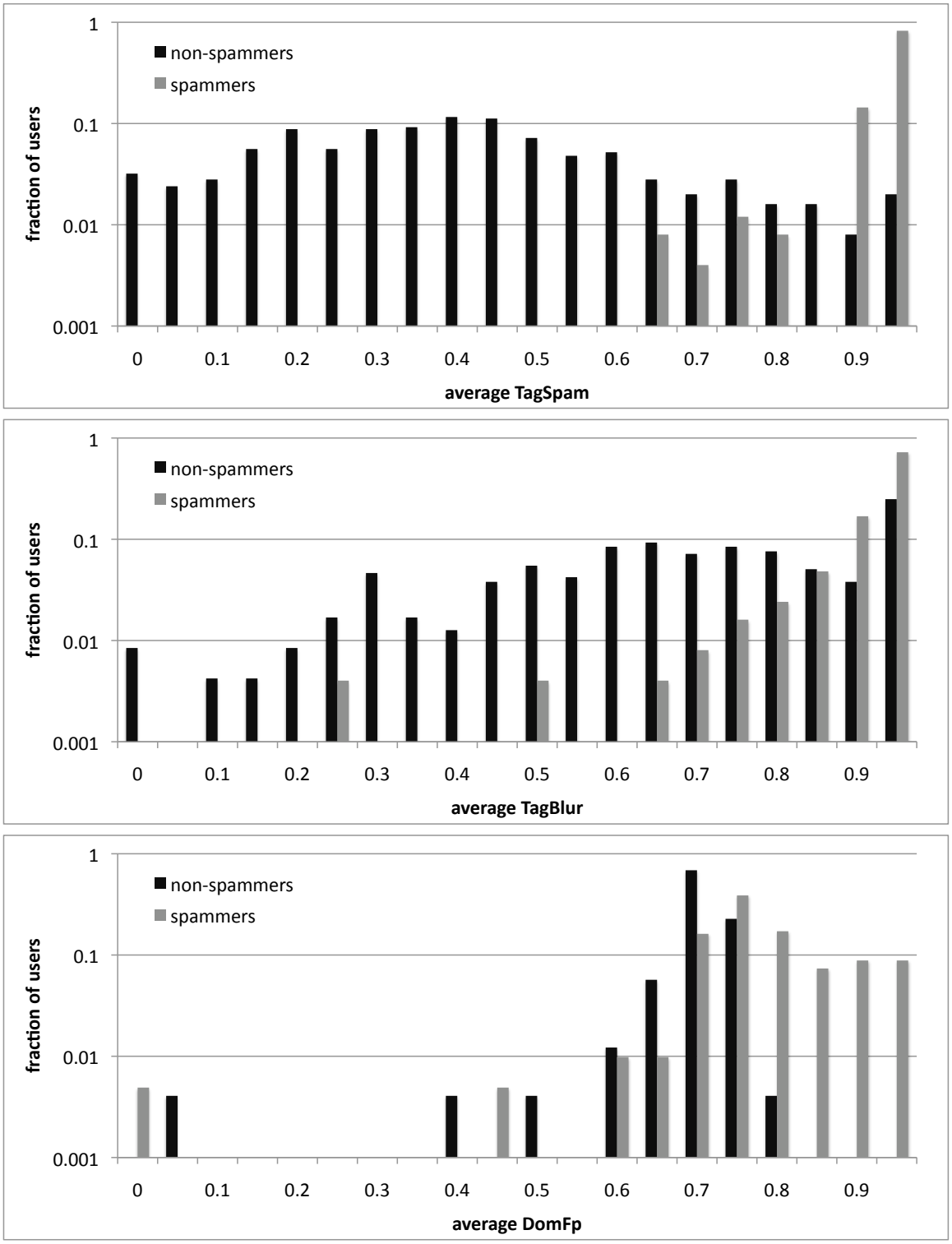


Figure 7.17: Distributions of the values of the three proposed features TagSpam, TagBlur, and DomFp for spammers versus legitimate users. Each feature's distribution is based on the subset of users for which the feature is defined (cf. text).

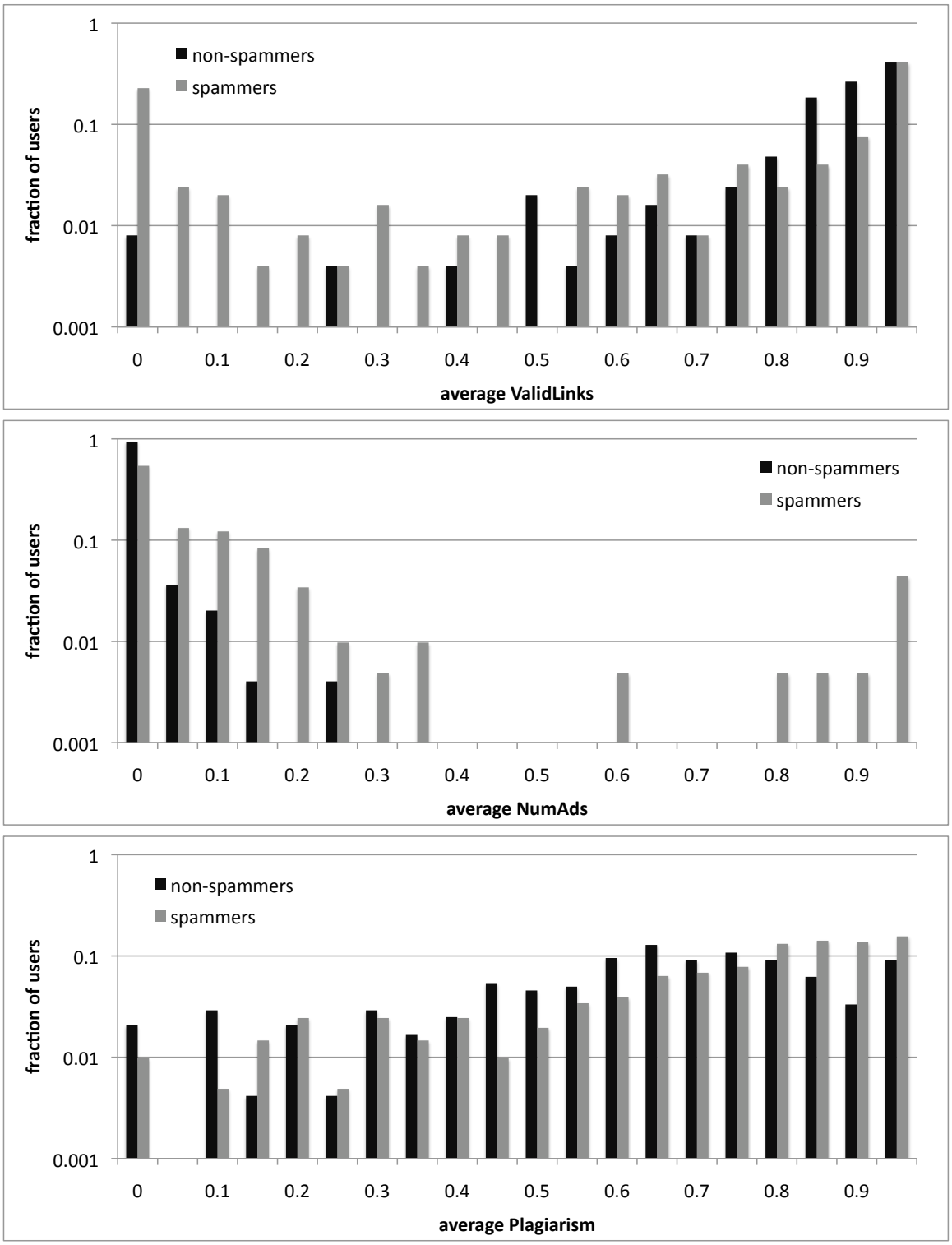


Figure 7.18: Distributions of the values of the three proposed features ValidLinks, NumAds, and Plagiarism for spammers versus legitimate users. Each feature's distribution is based on the subset of users for which the feature is defined (cf. text).

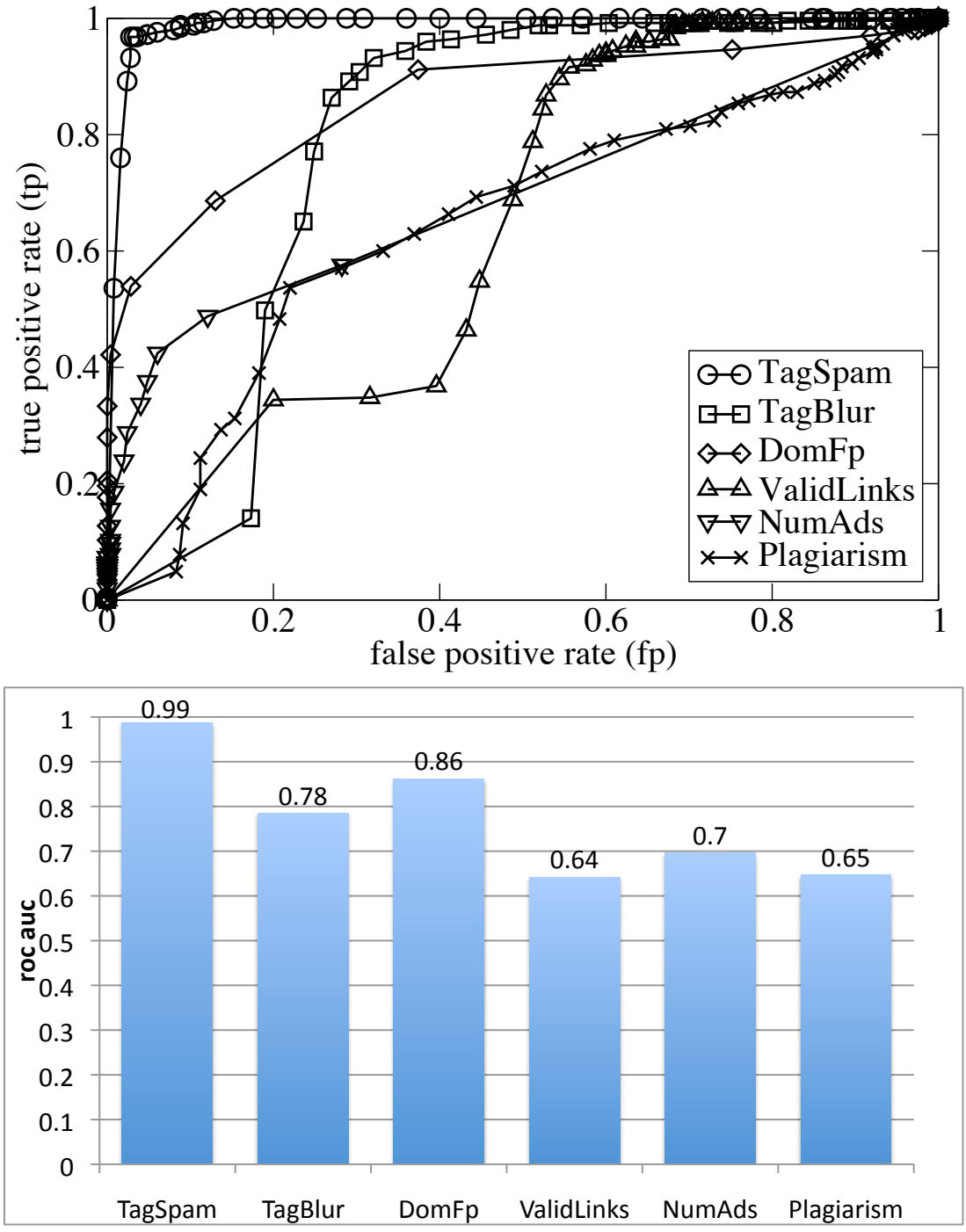


Figure 7.19: Top: ROC curves for each of the proposed features to detect spammers. A true positive is a correctly classified spammer, a false positive is a legitimate user incorrectly classified as a spammer. Bottom: Areas under the ROC curves (AUC) for the six features. Larger AUC denotes better detection trade-off between true and false positives for the linear classifier based on the corresponding feature.

7.4.3.2 Classification

Having found that all of the proposed six features have predictive power, we used various supervised learning algorithms to construct social spam detectors based on these features. We turned to the Weka software library ([WF05], `cs.waikato.ac.nz/ml/weka`) for off-the-shelf implementations of many established machine learning algorithms. To use all features we focused on the subset of 431 users (203 spammers) in our dataset for whom all six features are defined. We evaluated many of the classifiers in the Weka suite using default values for all parameters and 10-fold cross-validation. The top 30 classifiers and their performance are shown in Table 7.1. The best algorithm, additive logistic regression (LogitBoost), even without any tuning reaches an accuracy of almost 98% with a false positive rate below 2%. All classifiers perform very well, with accuracy over 96% and false positive rate below 5%. This attests to the effectiveness of our proposed features.

To explore the issue of feature selection and see if the accuracy can be further improved by tuning a detector's parameters, let us consider AdaBoost, a well-known and popular ensemble classifier whose performance is very close to the best (AdaBoostM1 in Table 7.1). For sake of comparison we also consider the linear support vector machine (SVM), another widely popular algorithm (SMO in Table 7.1). After tuning the parameters for both we could not improve on the default SVM, while AdaBoost's performance was enhanced by simply extending the number of iterations to 1000. AdaBoost thus achieved an accuracy of 98.4% (better than LogitBoost with default parameters), with a false positive rate of 2% and $F_1 = 0.98$ (comparable to LogitBoost). This is our best result.

The effect of feature selection is reported in Table 7.2 and displayed in Figure 7.20. In the case of SVM we see a modest improvement in accuracy and decrease in false positive rate by using both *TagSpam* and *TagBlur*, but additional features do not help. Performance is actually hindered by the addition of the *ValidLinks* feature. This is understandable because this feature does not give a clean linear separation of spammers from legitimate

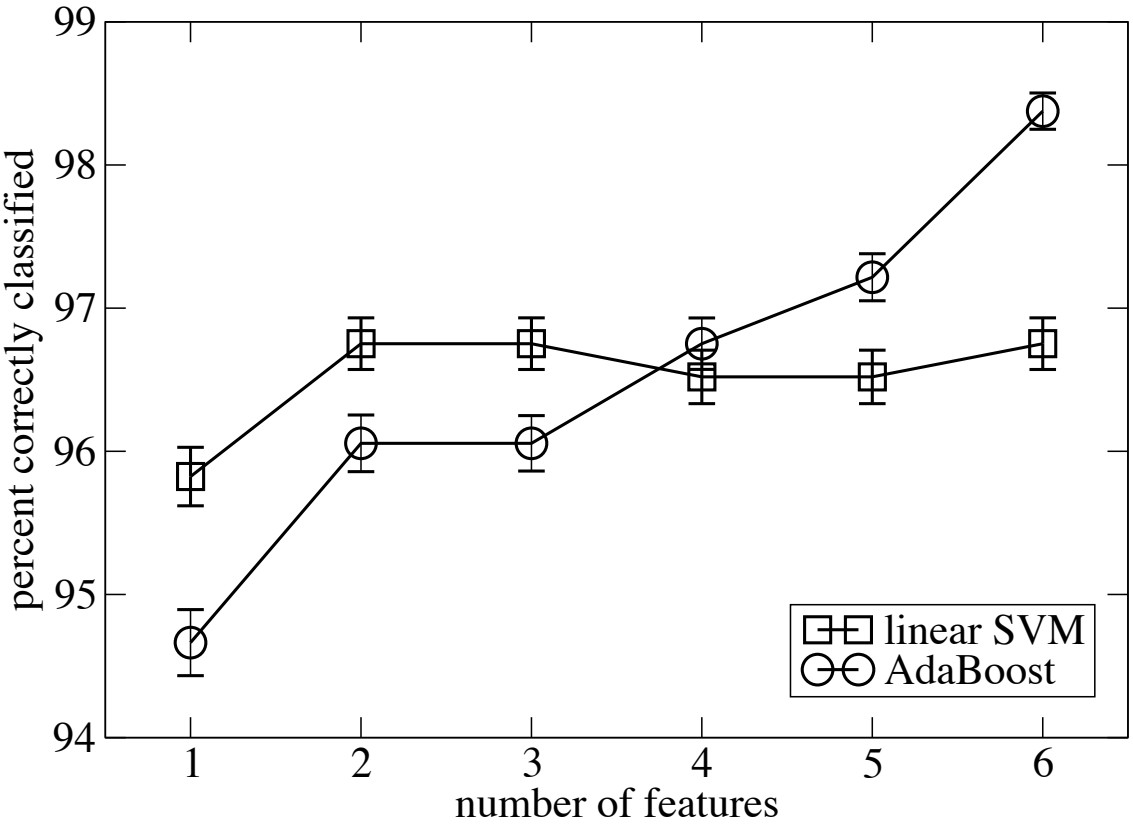


Figure 7.20: Accuracy of linear SVM and AdaBoost social spam detectors with features select in order of discrimination power (cf. Figure 7.16). Error bars are based on root mean squared error reported by Weka.

users and the linear SVM is therefore unable to exploit it. AdaBoost, on the other hand, is able to combine evidence from all features and thus improve both accuracy and false positive rate by learning from as many features as are available.

7.4.4 Discussion

Our discussion of the incentives of social spam has motivated the design of a number of novel features to detect spammers who abuse social bookmarking systems. These features all have strong discriminating power in detecting spammers; using just one of them allows

to correctly classify about 96% of users with a linear discriminant function, while combining all six features boosts the accuracy to over 98% with an ensemble classifier. At the same time the false positive rate can be kept below 5% with a single feature and pushed down to 2% combining all features. These promising results provide a new baseline for future efforts on social spam. To facilitate such efforts we have made our dataset freely available to the research community at `GiveALink.org/socialspam`.

From an efficiency/feasibility perspective, the *TagBlur* feature looks promising. While not quite as predictive as *TagSpam*, it relies on tag-tag similarity measures, which can be maintained up-to-date with incremental techniques [MRM08, MCM+09b] and therefore are always available (cf. 5.4.2). Other features rely on the availability of infrastructure to enable access to resource content (e.g. *DomFp*, *NumAds*) or the cooperation of a search engine (e.g. *Plagiarism*) and thus their feasibility depends on the circumstances of a particular social annotation system. Another efficiency issue, not explored here, is feature selection on the tags. The computation of features such as *TagSpam* and *TagBlur* might be greatly accelerated by focusing on a subset of the tags in the folksonomy, for example the most frequently used.

Given the current financial incentives for social spam, we have no doubt that this is but one early chapter in an escalating arms race to combat the emerging phenomena of Web 2.0 abuse.

Table 7.1: Top Weka classifiers, ranked by accuracy (fraction of users correctly classified). Also shown is the false positive rate (FP), related to precision and defined as the fraction of legitimate users who are wrongly classified as spammers. In a deployed social spam detection system it is more important that the false positive rate be kept low compared to the miss rate, because misclassification of a legitimate user is a more consequential mistake than missing a spammer. The F_1 measure is the harmonic mean of precision and recall. Recall is related to misses (undetected spammers). Each classifier uses default parameter values, is trained using all six features, and is validated with 10-fold cross-validation. Best scores are highlighted.

Weka Classifier	Accuracy	FP	F_1
LogitBoost	**97.91%**	.018	**.978**
LWL	97.68%	**.013**	.975
AdaBoostM1	97.68%	.018	.975
ConjunctiveRule	97.68%	**.013**	.975
DecisionTable	97.68%	.018	.975
DecisionStump	97.68%	**.013**	.975
RandomCommittee	97.45%	.018	.973
RandomForest	97.45%	.018	.973
Bagging	97.22%	.022	.970
NNge	97.22%	.022	.970
ADTree	97.22%	.026	.970
ClassificationViaRegression	96.98%	.031	.968
Decorate	96.98%	.018	.968
MultiBoostAB	96.98%	.026	.968
LMT	96.98%	.044	.969
REPTree	96.98%	.026	.968
RBFNetwork	96.75%	.031	.966
SMO	96.75%	.048	.966
JRip	96.75%	.039	.966
OneR	96.75%	.035	.966
PART	96.75%	.039	.966
J48	96.75%	.039	.966
BayesNet	96.52%	.039	.963
Logistic	96.29%	.044	.961
VotedPerceptron	96.29%	.053	.961
NaiveBayes	96.06%	.035	.958
NaiveBayesSimple	96.06%	.035	.958
NaiveBayesUpdateable	96.06%	.035	.958
MultilayerPerceptron	96.06%	.035	.958
SimpleLogistic	96.06%	.048	.959

Table 7.2: Performance of linear SVM and AdaBoost social spam detectors as we select features in the order of their discrimination power (Figure 7.16). Best performance (highlighted) is when all features are used.

	SVM			AdaBoost		
Features	Accuracy	FP	F_1	Accuracy	FP	F_1
TagSpam	95.82%	.061	.957	94.66%	.048	.943
+ TagBlur	96.75%	.048	.966	96.06%	.044	.958
+ DomFp	96.75%	.048	.966	96.06%	.044	.958
+ ValidLinks	96.52%	.048	.964	96.75%	.026	.965
+ NumAds	96.52%	.048	.964	97.22%	.026	.970
+ Plagiarism	96.75%	.048	.966	**98.38%**	**.022**	**.983**

8

Conclusion

8.1 Summary

Social annotation systems allow users to freely markup any online resource. This research exploits two representations of user annotations: hierarchical and tripartite (cf. § 4). From these representations, relationships among tags and resources are extracted. These relationships lead to novel applications in tag recommendation, resource recommendation, query result interfaces, and spam detection.

The hierarchical representation can be found in all major Web browsers. This representation is more strict in comparison to the tripartite representation. From the hierarchical representation, we apply Lin (cf. § 4.1.1) and incremental Lin (cf. § 4.1.2) for extracting relationships among links in browser bookmarks. An evaluation that involves the approximation of reference similarities shows that similarities induced by Lin and incremental Lin outperform a randomly generated set of similarities (cf. § 6.1). Furthermore, a user study found that Lin outperformed Google for finding similar online resources [MSM06c]. Unfortunately, Lin has a global dependency that leads to inefficient updates to the global semantic similarity network. In order to address this, we introduced incremental Lin (cf. § 4.1.2). Incremental Lin is able to recover the performance of Lin efficiently (cf. § 6.1).

The tripartite representation is the underlying structure of many collaborative annotation systems. These annotations can be in the form of tags, votes, ratings, and direct user feedback. These systems have achieved widespread use with applications in music (e.g. `last.fm`), movies (e.g. `imdb.com`), images (e.g. `flickr.com`), and bookmarks (e.g. `delicious.com`). In § 4.2, a framework for extracting relationships among objects (i.e. tags, resources, users) from the tripartite representation is introduced. We have demonstrated that our maximum information path outperforms all other measures in terms of an efficiency/effectiveness trade-off.

Navigation on the Web can be improved with the relationships induced from social annotation systems. These relationships are much different than the hyperlinks embedded in Web pages. We explore using these relationships to aid users when viewing results from a search engine query. A hybrid approach combining the ranked list and visualizing the relationships among the results enables users to find information with fewer queries (cf. § 7.3.3).

The ease of use and popularity of social bookmarking tools have made them a prime target for spammers. For the purpose of detecting social spam, six features of spammers were identified (cf. § 7.4.2). One of the features examines the focus of the tags used to annotate a resource. Tags that are semantically dissimilar from one another are more likely to signal spam. Another feature for spam detection is plagiarized content. Pages with content taken from another site, such as Wikipedia, are more inclined to be from spammers. Other suspicious features include ads, broken links, and automatically generated HTML. Combining these features lead to a spam detection accuracy above 98% (cf. § 7.4.3.2).

The socially induced semantic network and the applications explored here can lead to better online tools. In the following section, I will discuss some of our ongoing research.

8.2 Future Work: Towards the Web 3.0

The Web 3.0 refers to the integration of Semantic Web and Artificial Intelligence technologies for information understanding. Social participation has become widespread, and it is now the duty of future applications to mine the user supplied data. We exploit this data for inducing semantic networks to achieve a level of understanding. There are many avenues one may pursue based on these networks beside the applications already explored in prior work. For example, the similarity measures currently do not take into account information such as favoring more active users, the temporal dimension, and alternative collaborative filtering methods. Also, the evaluation framework introduced in § 6 can be further studied. For example, should the analysis include the entire collection of similarities or just the top similarity values? Below is a brief description of other research questions currently under development.

8.2.1 Retrieval and Ranking

The most natural application of a similarity network is recommendation, i.e., given a resource/tag return the most similar objects (resource/tag). We will explore techniques for mining the similarity network for studying different ranking methods. We call these alternative ranking methods *popularity*, *novelty*, and *personalization* [SHM+05, MSM06b, MSM06c]. We then present open questions in regards to search based on these ranking methods. For illustration, let us refer to any similarity measure from § 4 as $\sigma(x, y)$. All similarity measures $\sigma(x, y)$ must be normalized to $[0, 1]$, such as collaborative aggregation with maximum information path. Because the hierarchical representation is easily converted to a tripartite (cf. Figure 4.3), we will refer to any *item* as a tag, resource, or user.

8.2.1.1 Popularity

The first measure we will look at is *Popularity*. This measure provides a total order of the objects independent of any user query. *Popularity* is the term we use to describe to non-technical users the *centrality* of an object in the similarity matrix. The centrality of an item is the average of the shortest-path similarities between this node and every other node. An item with high centrality is one that is very similar to all other items in the collection.

Calculating centrality requires the computation of the similarity between all pairs of nodes according to all of the paths between them. There are many ways in which this can be done. One possible approach is to compute the similarity on a given path as the product of the similarity values along all edges in the path. For example, if item x and y are connected by a path $x \rightsquigarrow y$ that goes through z, where $\sigma(x, z) = 0.5$ and $\sigma(z, y) = 0.4$, then the similarity between x and y on that path is $\sigma(x \rightsquigarrow y) = 0.5 \times 0.4 = 0.2$. Although this approach is rather intuitive, it is too aggressive for computing similarities [Roc05].

One method of computing shortest path is to convert similarity values to distances, then compute shortest-path distances using Floyd-Warshall's algorithm [CLRS01], and finally convert these values back into shortest-path similarity values. To convert between similarity and distance values, one may use the following formula:

$$dist(x, y) = \frac{1}{\sigma(x, y)} - 1. \tag{8.1}$$

Note that when similarity is equal to 1, distance is equal to 0, and when similarity is equal to 0, distance is infinity. Thus the closer two resources are, the higher their similarity is. The distance along a given path is the sum of the distances along all edges in the path. The

shortest-path similarity between two items is thus defined as

$$\sigma_{max}(x,y) = \left[1 + \min_{x \rightsquigarrow y} \sum_{(u,v) \in x \rightsquigarrow y} \left(\frac{1}{\sigma(u,v)} - 1\right)\right]^{-1}. \tag{8.2}$$

Computing popularity is time consuming. If the number of items in our database is U, then calculating all-pairs-shortest-path has complexity $O(U^3)$.

Because of the complexity of all-pairs-shortest-path, we consider an approximation we call *prestige*. Prestige is a recursive measure inspired by PageRank [BP98] — the prestige of an item is tied to the prestige of its neighbors in the similarity graph. The difference between our prestige and PageRank is that PageRank is computed on a directed, unweighted graph where edges represent hyperlinks. Prestige is computed on an undirected, weighted graph in which the weights of edges represent social similarity σ as defined by a similarity matrix. In the undirected, weighted case, the solution to prestige is a system of homogeneous linear equations:

$$p(i) = (1-\alpha) + \alpha \times \sum_j \frac{\sigma(i,j) \times p(j)}{\sum_k \sigma(j,k)} \tag{8.3}$$

where the denominator is the *strength* of node j:

$$\kappa_j = \sum_k \sigma(j,k). \tag{8.4}$$

When $\alpha = 1$, the solution to prestige is proportional to strength: $p(i) \propto \kappa_i$. This has been proven analytically in the undirected, unweighted case for node degree [Bol98]. The result can be generalized to the undirected, weighted case for strength. Setting $\alpha = 1$, we simplify equation 8.3 to

$$p(i) = \sum_j \frac{\sigma(i,j) \times p(j)}{\sum_k \sigma(j,k)}. \tag{8.5}$$

To see that strength is a possible solution, we can substitute κ into equation 8.5 yielding

$$p(i) = \sum_j \frac{\sigma(i,j) \times \kappa_j}{\sum_k \sigma(j,k)}. \tag{8.6}$$

Expanding κ_j gives us

$$p(i) = \sum_j \left(\frac{\sigma(i,j)}{\sum_k \sigma(j,k)} \times \sum_l \sigma(j,l) \right). \tag{8.7}$$

Both terms $\sum_k \sigma(j,k)$ and $\sum_l \sigma(j,l)$ correspond to κ_j, and cancel each other:

$$p(i) = \sum_j \sigma(i,j) = \kappa_i. \tag{8.8}$$

Therefore, prestige coincides with Equation 6.1 and with κ. Thus we use the strength of a node as a proxy for prestige; strength is computed very efficiently.

8.2.1.2 Novelty

Next, we introduce a measure we call *novelty*. If we had the all-pairs-shortest-path similarities, pairs of items may have an indirect shortest-path similarity higher than the direct edge similarity. There are pairs of items (x, y) where $\sigma(x, y)$ is relatively low, but if both x and y are very similar to a third item z, then their shortest-path similarity could be much higher. This phenomenon, known as semi-metric behavior [Roc02], is valuable for a recommendation system because it reveals similarity that is implied by the global associative semantics but has not yet been discovered by individual users. If used in addition to a direct similarity measure, it may empower the recommendation system to not only generate natural and obvious suggestions, but also unexpected ones that could inspire users to broaden and deepen their interests.

From an implementation perspective, the runtime complexity of all-pairs-shortest-path

is $O(|V|^3)$ where $|V|$ is the number of nodes in the graph. For feasibility, we will explore an approximation where only paths of length at most two edges are evaluated. This removes an order of magnitude from the calculation because each object pair now depends on the average number of edges per node. This reduces the complexity to $O(|V|^2 \times \frac{|E|}{|V|}) = O(|V| \times |E|)$ where $|E|$ is the number of edges.

We use the all-pairs-shortest-path approximation σ'_{max} to exploit semi-metric behaviors by a novelty measure defined as

$$novelty(x,y) = \begin{cases} \frac{\sigma'_{max}(x,y)}{\sigma(x,y)} & \text{if } \sigma(x,y) > 0 \\ \frac{\sigma'_{max}(x,y)}{\sigma_{min}} & \text{if } \sigma(x,y) = 0 \end{cases} \tag{8.9}$$

where σ_{min} is the smallest non-zero similarity value, $\sigma_{min} = \min_{\sigma(x',y')>0} \sigma(x',y')$. This measure is similar to one of the semi-metric ratios introduced by Rocha [Roc02]. For purposes of recommendation we are only interested in pairs of items where $novelty(x,y) > 1$, i.e., the indirect similarity is higher than the direct similarity. As the gap grows, the novelty value grows as well.

We call this measure novelty because, when a user submits query x and our search engine returns answer y, where $\sigma(x,y)$ is low but $\sigma'_{max}(x,y)$ is high, then y is a valid recommendation and is novel with respect to the measured associations of any one user.

A natural extension of ranking search results by novelty is a semi-metric recommendation system that is roughly equivalent to searching by novelty. In the standard recommendation system evaluated in § 7.2, results are generated by mining the database for objects that have high similarity to the user query. Thus the standard search system is essentially "search by similarity." On the other hand, in the semi-metric recommendation system, the results will be objects that have high *novelty* to the user query. Results are generated from different sets of objects in the two applications. The recommendation system considers all objects in the database and picks the ones most similar to the query. The semi-metric

recommendation system will only consider the items that have higher shortest-path similarity than direct similarity to the query, and pick the ones with highest novelty. This system will provide different information that relates to the query in a non-trivial way. Rather than presenting similar information, semi-metric results will provide pages that address the same questions from a different perspective: a different domain of knowledge, perhaps a different time period or geographical location. Thus the semi-metric recommendation system could inspire collaboration between groups of users who do not yet realize they have similar interests.

8.2.1.3 Personalization

User profiles can be used to personalize search results. With the help of a semantic similarity network, we may calculate a personal similarity score between every item and the profiles of each registered user. This new similarity will be based on how similar the item is to the user's item set. There are many possible ways to quantify this similarity measure. One option is to calculate the average similarity between the given item and the user's profile. Based on preliminary analysis, we believe this approach is not appropriate because a user may have many interests that are very different in nature. For example, if somebody reads sites about both politics and Linux, a URL about Gentoo Linux will have low average similarity to the user's bookmarks but will nonetheless be interesting to the user because it coincides with one of her interests.

The alternative measure that we will explore is the maximum similarity between the given item and an item in the user's collection. If a user profile contains a set of items (e.g., tags or resources) B, then the similarity between some other item x and B will be:

$$\sigma_p(x, B) = \begin{cases} \max_{y \in B} \sigma(x, y) & \text{if } x \notin B \\ 1 & \text{if } x \in B. \end{cases} \tag{8.10}$$

In addition, when personalizing search results, we would like to pay particular attention to the interests of the user that are unique and stand out in respect to the other users. For example, the default bookmarks and other bookmarks that most people have should not overly affect the personalization. Thus we wish to weigh the similarity between a item and a user's bookmark by how unlikely it is that the user has that bookmark, in a way analogous to the inverse document frequency in the TFIDF weighting scheme [SJWR00]. If the number of users is N (i.e., number of registered users) and the number of those who own item y is $N(y)$, we can modify the measure as follows:

$$\sigma_p(x, B) = \begin{cases} \max_{y \in B} \left[log \frac{N}{N(y)} \times \sigma(x, y) \right] & \text{if } x \notin B \\ 1 & \text{if } x \in B. \end{cases} \tag{8.11}$$

The personalized similarity measure is computationally expensive: if U is the number of items in our database, B is the number of items in the largest bookmark collection donated by a user, and N is the number of users, then the algorithm has complexity $O(U \cdot B \cdot N)$. We will precompute the personalized similarity scores for all registered users and store them in the database.

In our system at `GiveALink.org`, once a user is logged in, we make personalized recommendations according to our measure. Figure 8.1 is a screen shot of personalized resource recommendations given a user's profile. The order of the Web links are ranked according to the resource's personal similarity.

8.2.1.4 Search

Search is similar to recommendation except that it start from a keyword query. The user may wish to obtain relevant resources, or related tags. This can be particularly useful in an annotation or bookmark management setting. Such applications can be provided

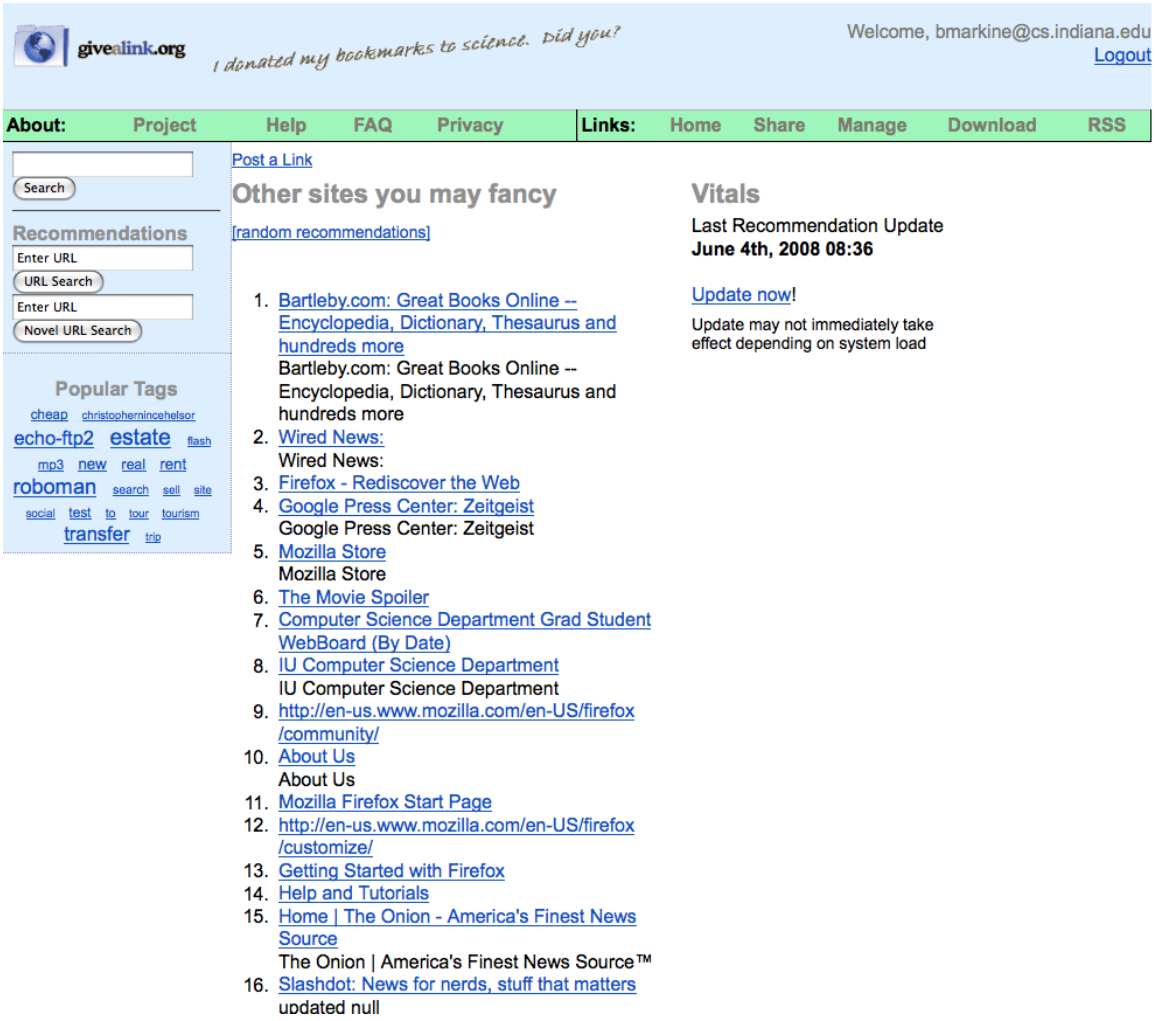


Figure 8.1: Screenshots of personalized recommendations based on the most similar URLs to a profile.

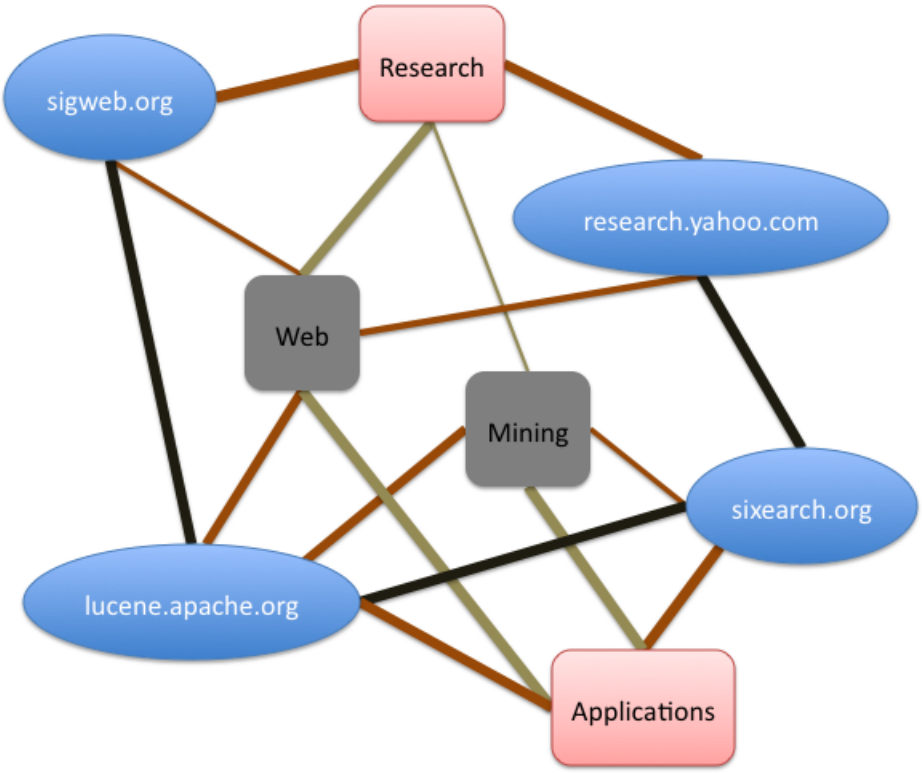


Figure 8.2: An example of a graph for search. Given the query "Web Mining," we can mine a network of tags and a network of resources for ranking query results. The resource network is connected to the tag network by counting the number of times a resource is annotated by a tag.

as Web services for third-party applications. Existing social bookmarking systems such as `delicious.com` provide these functionalities, based on simple relations such as frequency of co-occurrence or chronological ordering [WZM06, HJSS06b]. We will explore more sophisticated measures based on our similarity network.

In search applications it is important to have a query-independent function capable of ranking resources by their authority, prestige, or popularity. Also, a viable search system based on social annotation needs to address several issues. First, does one apply stemming or other morphological normalization techniques to query keywords and tags in an effort

to maximize the match probability and this recall? Much research on such issues exist in information retrieval, however it is not clear if all the results are directly applicable to social annotation systems. For example, TFIDF is usually helpful when representing content based on the assumption that very common terms are noisy. Such assumption may not carry through to tags. Second, for multi-term queries, how to combine ranking based on tag matches with other measures? For instance, if the query contains two terms, one would want to show resources tagged with both terms before those tagged with only one. At the same time, one may want to rank according to some query independent centrality measure. It is not clear which of these criteria should take precedence or how they should be combined. Furthermore, one could use tag similarity to expand the query; would we prefer a resource matching none of the query terms but several related tags, or a resource matching just one of the original query terms? And finally, suppose a highly relevant resource is identified based on many matched tags. Social similarity (or other network measures, such as novelty) would identify related resources. How to rank strongly related resources that do not match the query, vs. resources that are weakly associated with query terms? And what about combining multiple network measures such as similarity, novelty, centrality, and personalization? Figure 8.2 is an example of the structure we can exploit for retrieval of search results. All of these are open questions requiring careful empirical evaluation to experiment with appropriate retrieval and ranking functions that integrate and weigh many factors.

8.2.2 User Prediction

Even though we introduced similarities among tags and resources, one can easily extend these measures to compute user similarity. Recalling from § 4.2, an open question is to aggregate over tags or resources to induce user similarity. The site `last.fm` presents an opportunity for evaluating our 'social similarity' measure to induce relationships among

users. An application based on these mined relationships is friend prediction similar to the 'People You May Know' application in `facebook.com`. One evaluation is to test whether there is a correlation between the extracted similarities, and user's friends lists. Another evaluation is to plot ROC curves as the threshold for the similarities change.

8.2.3 Social Spam Detection

In an effort to bring the findings from § 7.4 to bear on contemporary tagging systems, making them more robust with respect to social spam, we are currently working on the integration of a spam detection system into `GiveALink.org`. Of course spam detection is just one of the measures that must be put in place to defend a social bookmarking system from spammers. Other measures include prevention (e.g., through captchas) and mitigation (e.g., through ranking algorithms that penalize tags used by spammers) [HKGM07].

When deploying a spam detection system in a live social tagging site it is not necessary to aggregate post-level features into user-level features, as was done due to evaluation dataset constraints (cf. § 7.4.3). We plan to experiment with the option of detecting individual spam posts rather than classifying users. The detection system can be used in many ways: to filter posts, to flag posts to a moderator, or to flag users, perhaps when a significant portion of their posts is deemed to be spam. It remains to be seen whether the finer resolution of posts will lead to increased effectiveness (by decreasing false positives) or to an encouragement of anti-social behavior.

Bootstrapping is an open issue. In the absence of spam labels, or until these may become available through a user feedback mechanism, we need spam assessments to compute the features that rely on tag spam statistics, such as *TagSpam* and *DomFp* (defined in § 7.4.2). One approach we are exploring is to bootstrap the probabilities necessary to compute these features using the other features, which do not require supervision, on a sample of annotations. There are also indirect dependencies on labeled data, however.

TagBlur (defined in § 7.4.2) relies on tag similarity, which is assumed to be computed on legitimate annotations. An abundance of spam in the folksonomy would bias the tag similarity values making *TagBlur* less effective. For example if "software" and "sex" co-occur often (due to spam posts), the system could wrongly conclude that these tags are related, missing posts such as those in Figure 7.15. Therefore the similarity computations must exclude posts labeled as spam. In general, a deployed social spam detection system must be incrementally trained. As similarities are updated and spam labels collected in response to newly received annotations, the feature values of incoming posts that depend on these assessments need to be computed with the latest statistics to keep the detector fresh. A question for future research is whether after the initial bootstrap phase, unsupervised features such as *ValidLinks* or *Plagiarism* (defined in § 7.4.2) should continue to be used — along with user input — to update the supervised features, in semi-supervised fashion.

8.2.4 Bookmark Management

Many browser interfaces for bookmarking operate on a hierarchical file and folder structure that can be easily navigated [RBMM09a, RBMM09b]. Hierarchies in this situation make it difficult to deal with non-hierarchical structure, such as intersections between topics. Labeling a bookmark with two dissimilar labels means bookmarking it twice, an inefficient organizational method. For example, a user may want to organize a link as a *movie* and a *blog*. In the user's bookmark hierarchy, it is possible that the folders *movie* and *blog* only share the top most folder. Selecting an appropriate location in the tree that works for the user would require her to bookmark the link twice or reorganize her bookmarks.

Collaborative social bookmarking has, for many people, eclipsed the use of traditional client-side bookmark managers, including those built into browsers. These social bookmarking services traditionally operate server-side, but there are a number of browser

plugins to bring the application to the client. The most popular of these browser plugins is the one for delicious (`addons.mozilla.org/en-US/firefox/browse/type:1/cat:22`).

As each user's bookmark collection grows, new management interfaces are necessary, and no established system seems to incorporate the strengths of both bottom up (tagging) and top-down (hierarchical) approaches. A semantic similarity network may induce a more efficient method of managing bookmarks. The user should be able to synchronize online bookmarks with those on the browser, using an interface that combines the most appealing aspects of traditional, hierarchical bookmark managers — as in Internet Explorer — with the tagging organization found on social sites. We see a critical synergy between the bookmark manager and the emergent social semantic structure: the manager will increase the amount of annotation and facilitate its sharing by making it convenient to organize resources according to the users preferred interface; in turn, the similarity networks can enhance the user experience by making suggestions, mapping resources and tags in meaningful ways, and allowing the user to navigate through these spaces. For example, we will explore spanning tree and clustering algorithms over the similarity graphs to build semantic structures that help the user cope with their growing collections. `GiveALink.org` will support client-side organization (through browser extensions).

The dual, integrated view of both personal and social bookmarks will lead to new methods for both organization and recommendation. The application will employ a tag cloud comprised of a user's most popular tags, helping to encourage consistency in tag use. Our approach will also leverage the use of both general (more popular) and specific tags, helping to preserve a taxonomic system approximating that of browsers hierarchical bookmark managers.

The GiveALink application will offer the user a visualization of semantic tag networks induced by socially-aggregated tag similarity. Global visualizations of the social network

of tags centered around the users topic will function as a recommendation for tags, and then for new content. Visualizations of a users personal network of tags can, again, reinforce common, higher-level tags, as well as offer content-oriented recommendations from his or her own network.

Another function of the bookmark manager is the visualization of a pages semantic context beyond its content. An interface that provides a semantic map will allow the user to see where the current page is in terms of related tags and pages. Users can use the map as an alternative way to understand the content and context of the page, as well as an alternative way to navigate the Web, by visiting or exploring related pages and tags.

Figure 8.3 illustrates a very early version of the proposed bookmark manager. The extension displays a sidebar taking up a portion of the browser space, in an effort to allow the user to continue browsing comfortably without alerts or popups to bookmark data. As the figure illustrates, a user is able to view a tag cloud of his or her most popular tags, visualize the semantic relationships between tags, and view the resources annotated by the tags.

8.2.5 Tagging Game

As an incentive for users to annotate resources and relationships, we propose an online game [vA06, vAD08, RBMM09a, RBMM09b]. Users will tag pages to find a path from a given starting resource to a target resource. We will pair resources selected from the users collection with (possibly weakly related) randomly chosen ones. Each time a user provides a tag, `GiveALink` will store the tagging metadata. When enough participants have confirmed a tag resource pair, the similarity networks are updated accordingly. Players will be rewarded for the amount of information they provide, and top scores will be posted. The game will entertain users while simultaneously addressing an important problem,

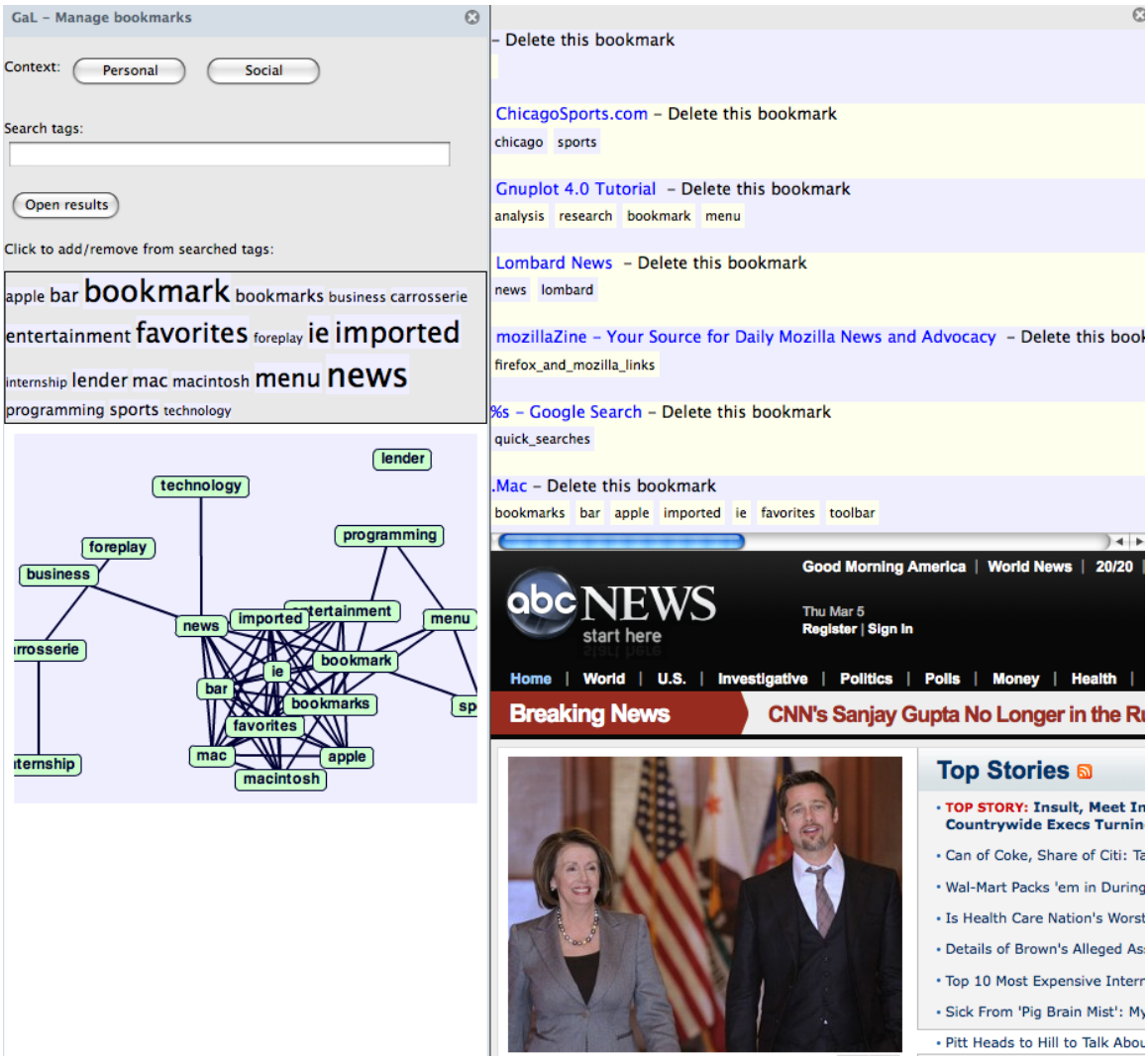


Figure 8.3: A screenshot of an early `GiveALink` bookmark manager plugin prototype.

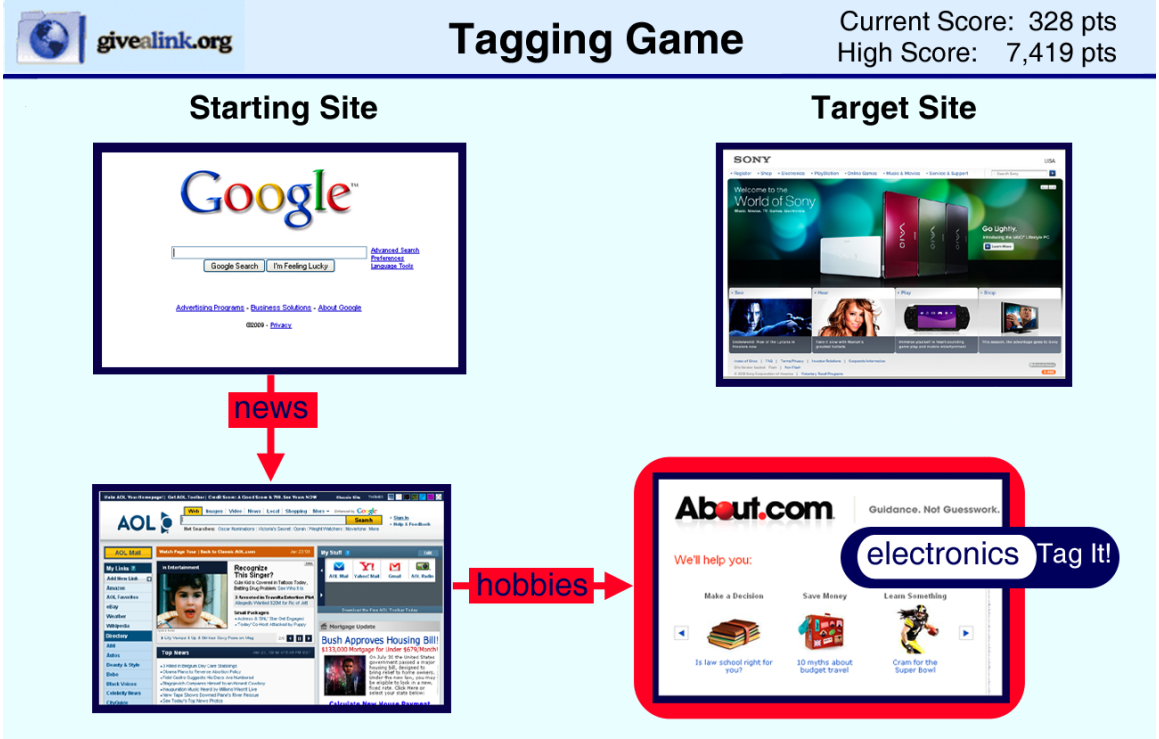


Figure 8.4: A screenshot of an early version of the `GiveALink` tagging game.

namely the sparseness of the semantic networks. Figure 8.4 shows an early prototype of the tagging game.

Bibliography

[ADWM04] Alex T. Adai, Shailesh V. Date, Shannon Wieland1, and Edward M. Marcotte. Lgl: Creating a map of protein function with an algorithm for visualizing very large biological networks. *Journal of Molecular Biology*, 2004.

[AJK05] Anne Aula, Natalie Jhaveri, and Mika Käki. Information search and re-access strategies of experienced web users. In *Proc. of the 14th Intl. Conference on World Wide Web*, pages 583–592, New York, NY, USA, 2005. ACM.

[AJR+01] Ping An, Alin Jula, Silvius Rus, Steven Saunders, Tim Smith, Gabriel Tanase, Nathan Thomas, Nancy Amato, and Lawrence Rauchwerger. Stapl: A standard template adaptive parallel c++ library. In *Proc. Intl. Workshop on Advanced Compiler Technology for High Performance and Embedded Processors*, 2001.

[ALSB01] James Allan, Anton Leuski, Russell Swan, and Donald Byrd. Evaluating combinations of ranked lists and visualizations of inter-document similarity. *Information Processing and Management*, 37:435–458, 2001.

[APH06] Melanie Aurnhammer and Luc Steels Peter Hanappe. Integrating collaborative tagging and emergent semantics for image retrieval. In *Proc. WWW2006, Collaborative Web Tagging Workshop*, May 2006.

[AS08] Josh Attenberg and Torsten Suel. Cleaning search results using term distance features. In *Proc. 4th Intl. Workshop on Adversarial Information Retrieval on the Web*, pages 21–24, 2008.

[BCB03] Katy Börner, Chaomei Chen, and Kevin Boyack. *Annual Review of Information*

Science and Technology, chapter Visualizing Knowledge Domains, pages 179–255. Inc./American Society for Information Science and Technology, 2003.

[BGMZ97] Andrei Z. Broder, Steven C. Glassman, Mark S. Manasse, and Geoffrey Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8–13):1157–1166, 1997. Proc 6th Intl. WWW Conf.

[BH06] Alexander Budanitsky and Graeme Hirst. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47, 2006.

[BISI+06] Judit Bar-Ilan, Snunith Shoham, Asher Idan, Yitzchak Miller, and Aviv Shachak. Structured vs. unstructured tagging — a case study. In *Proc. WWW2006, Collaborative Web Tagging Workshop*, 2006.

[BKS06] Grigory Begelman, Philipp Keller, and Frank Smadja. Automated tag clustering: Improving search and exploration in the tag space. In *Proc. WWW2006, Collaborative Web Tagging Workshop*, 2006.

[BLAZ08] Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. A few bad votes too many?: Towards robust ranking in social media. In *Proc. 4th Intl. Workshop on Adversarial Information Retrieval on the Web*, pages 53–60, 2008.

[BM98] Vladimir Batagelj and Andrej Mrvar. Pajek – program for large network analysis. In *Connections*, pages 47–57, 1998.

[BM06] Christopher H. Brooks and Nancy Montanez. Improved snnotation of the blogosphere via autotagging and hierarchical clustering. In *Proc. of the 15th Intl. Conference on World Wide Web*, pages 625–632, New York, NY, USA, 2006. ACM Press.

[BMS97] Satish Balay, Lois Curfman Mcinnes, and Barry F. Smith. Efficient management of parallelism in object-oriented numerical software libraries. In *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhauser Press, 1997.

[BMW97] Russell Beale, Rodger J. McNab, and Ian H. Witten. Visualising sequences of queries: A new tool for information retrieval. In *Proc. IEEE Conf. on Information Visualisation*. IEEE Computer Society, 1997.

[Bol98] Bela Bollobas. *Modern Graph Theory*. Springer-Verlag, 1998.

[Bou03] Maged Boulos. The use of interactive graphical maps for browsing medical/health internet information resources. *Intl. Journal of Health Geographics*, 2:1, 2003.

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, April 1998.

[BRA+08] Fabricio Benevenuto, Tiago Rodrigues, Virgilio Almeida, Jussara Almeida, Chao Zhang, and Keith Ross. Identifying video spammers in online social networks. In *Proc. 4th Intl. Workshop on Adversarial Information Retrieval on the Web*, pages 45–52, 2008.

[BS97] Marko Balabanovic and Yoav Shoham. Combining content-based and collaborative recommendation. *Communications of the ACM*, 40(3), March 1997.

[BS04] Richard Boardman and M. Angela Sasse. "stuff goes into the computer and doesn't come out": A cross-tool study of personal information management. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems*, pages 583–590, New York, NY, USA, 2004. ACM.

[BSV05] Paolo Boldi, Massimo Santini, and Sebastiano Vigna. Do your worst to make the best: Paradoxical effects in pagerank incremental computations. *Internet Mathematics*, 2(3):387–404, 2005.

[BTST06] Dominik Benz, Karen H. L. Tso, and Lars Schmidt-Thieme. Automatic bookmark classification: A collaborative approach. In *Proc. of the Second Workshop on Innovations in Web Infrastructure*, Edinburgh, Scotland, 2006.

[BXW+07] Shenghua Bao, Guirong Xue, Xiaoyuan Wu, Yong Yu, Ben Fei, and Zhong Su. Optimizing web search using social annotations. In *Proc. of the 16th Intl. Conference on World Wide Web*, pages 501–510, Banff, Canada, 2007.

[CBHS08] Ciro Cattuto, Dominik Benz, Andreas Hotho, and Gerd Stumme. Semantic grounding of tag relatedness in social bookmarking systems. In *Proc. Intl. Semantic Web Conference*, LNCS, Karlsruhe, Germany, 2008.

[CBSL07] Ciro Cattuto, Andrea Baldassarri, Vito D. P. Servedio, and Vittorio Loreto. Emergent community structure in social tagging systems. In *Proc. of the European Conference on Complex Systems*, Dresden, Germany, October 2007.

[CDT03] Albert Chan, Frank Dehne, and Ryan Taylor. Cgmgraph/cgmlib: Implementing and testing cgm graph algorithms on pc clusters. In *Intl. Journal of High Performance Computing Applications*. Springer, 2003.

[CG08] J.F. Chevalier and P. Gramme. RANK for spam detection ECML - Discovery Challenge. In *Proc. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2008.

[CH97] Julian C. Cummings and William F. Humphrey. Parallel particle simulations using the pooma framework. In *Proc. 8th SIAM Conference on Parallel Processing for Scientific Computing*, 1997.

[Chr06] Stijn Christiaens. Metadata mechanisms: From ontology to folksonomy ... and back. In *Proc. On the Move to Meaningful Internet Systems Workshop*, LNCS. Springer, 2006.

[CK97] Jeromy Carriere and Rick Kazman. Webquery: Searching and visualizing the Web through connectivity. *Computer Networks and ISDN Systems*, 29:1257–1267, 1997.

[CKS04] Karen Church, Mark T. Keane, and Barry Smyth. The First Click is the Deepest: Assessing Information Scent Predictions for a Personalized Search Engine. In *Proc. 3rd Workshop on Empirical Evaluation of Adaptive Systems*, pages 173–182, 2004.

[CLRS01] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, 2nd ed.* MIT Press, 2001.

[CLW08] James Caverlee, Ling Liu, and Steve Webb. Socialtrust: Tamper-resilient trust establishment in online communities. In *Proc. 8th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 104–114, 2008.

[DCM+08] Justin Donaldson, Michael Conover, Ben Markines, Heather Roinestad, and Filippo Menczer. Visualizing social links in exploratory search. In *Proc. 19th ACM Conference on Hypertext and Hypermedia*, 2008.

[DER86] Iain S. Duff, Albert M. Erisman, and John K. Reid. *Direct Methods for Sparse Matrices*. Oxford University Press, Inc., New York, NY, USA, 1986.

[DG04] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. In *Proc. of the Sixth Symposium on Operating Systems Design and Implementation*, 2004.

[DI06] Jorg Diederich and Tereza Iofciu. Finding communities of practice from user profiles based on folksonomies. *Proc. of the 1st Intl. Workshop on Building Technology Enhanced Learning solutions for Communities of Practice*, 2006.

[DNFY05] Fernando Das-Neves, Edward A. Fox, and Xiaoyan Yu. Connecting topics in document collections with stepping stones and pathways. In *Proc. 14th ACM Intl. Conference on Information and Knowledge Management*, pages 91–98, 2005.

[Edd96] Dirk Eddelbüttel. Object-oriented econometrics: Matrix programming in c++ using gcc and newmat. *Journal of Applied Econometrics*, 1996.

[FBFM09] Santo Fortunato, Marián Boguñá, Alessandro Flammini, and Filippo Menczer. On local estimations of pagerank: A mean field approach. *Internet Mathematics*, 4(2-3), May 2009.

[FFvA+06] George W. Furnas, Caterina Fake, Luis von Ahn, Joshua Schachter, Scott Golder, Kevin Fox, Marc Davis, Cameron Marlow, and Mor Naaman. Why do tagging systems work? In *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, pages 36–39, New York, NY, USA, 2006. ACM Press.

[FR91] Thomas M. J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Software- Practice and Experience*, 21:1129–1164, 1991.

[GC07] Zhiwei Guan and Edward Cutrell. An eye-tracking study of the effect of target rank on web search. In *Proc. of Computer Human Interaction Human Factors in Computing Systems*, pages 417–420. ACM Press, 2007.

[GGMP04] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. Combating web spam with trustrank. In *Proc. 13th Intl. Conference Very Large Data Bases*, pages 576–587, 2004.

[GGMW03] Prasanna Ganesan, Hector Garcia-Molina, and Jennifer Widom. Exploiting Hierarchical Domain Structure to Compute Similarity. In *ACM Transactions on Information Systems 21(1)*, pages 64–93, 2003.

[GH06] Scott Golder and Bernardo A. Huberman. The structure of collaborative tagging systems. *Journal of Information Science*, 32(2):198–208, April 2006.

[GK08] Anestis Gkanogiannis and Theodore Kalamboukis. A novel supervised learning algorithm and its use for spam detection in social bookmarking systems. In *Proc. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2008.

[GL05] Douglas Gregor and Andrew Lumsdaine. The parallel bgl: A generic library for distributed graph computations. In *Parallel Object-Oriented Scientific Computing*, 2005.

[GN99] Emden R. Gansner and Stephen C. North. An open graph visualization system and its applications. *Software - Practice and Experience*, 30:1203–1233, 1999.

[GS93] William Gropp and Barry Smith. Scalable, extensible, and portable numerical libraries. In *Proc. of the Scalable Parallel Libraries Conference*, 1993.

[HCL05] Jeffrey Heer, Stuart K. Card, and James A. Landay. Prefuse: A toolkit for interactive information visualization. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems*, pages 421–430, New York, NY, USA, 2005. ACM.

[HGM06] Paul Heymann and Hector Garcia-Molina. Collaborative creation of communal hierarchical taxonomies in social tagging systems. Technical Report 2006-10, Computer Science Department, 2006.

[HHLS05] Tony Hammond, Timo Hannay, Ben Lund, and Joanna Scott. Social Bookmarking Tools (I): A General Review. *D-Lib Magazine*, 11(4), April 2005.

[HJSS06a] Andreas Hotho, Robert Jäschke, Christoph Schmitz, and Gerd Stumme. BibSonomy: A social bookmark and publication sharing system. In *Proc. Conceptual Structures Tool Interoperability Workshop at the 14th Intl. Conference on Conceptual Structures*, pages 87–102, 2006.

[HJSS06b] Andreas Hotho, Robert Jäschke, Christoph Schmitz, and Gerd Stumme. Information retrieval in folksonomies: Search and ranking. In York Sure and John Domingue, editors, *The Semantic Web: Research and Applications*, volume 4011 of *LNAI*, pages 411–426, Heidelberg, 2006. Springer.

[HKGM07] Paul Heymann, Georgia Koutrika, and Hector Garcia-Molina. Fighting spam on social web sites: A survey of approaches and future challenges. *IEEE Internet Computing*, 11(6):36–45, 2007.

[HKGM08] Paul Heymann, Georgia Koutrika, and Hector Garcia-Molina. Can social bookmarking improve web search? In *Proc. Intl. Conference on Web Search and Data Mining*, pages 195–206, New York, NY, USA, 2008. ACM.

[HMC99] Paul Jen-Hwa Hu, Pai-Chun Ma, and Patrick Y. K. Chau. Evaluation of user interface designs for information retrieval systems: A computer-based experiment. *Decision Support Systems*, 27:125–143, November 1999.

[HMHS06] Yusef Hassan-Montero and Victor Herrero-Solana. Improving tag-clouds as visual information retrieval interfaces. In *Intl. Conference on Multidisciplinary Information Sciences and Technologies*, 2006.

[HMM00] Ivan Herman, Guy Melancon, and M. Scott Marshall. Graph visualization

and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6:24–43, 2000.

[HR08] Marti A. Hearst and Daniela Rosner. Tag clouds: Data analysis tool or social signaller? In *Proc. of the 41st Annual Hawaii Intl. Conference on System Sciences*, page 160, Washington, DC, USA, 2008. IEEE Computer Society.

[HRGM08] Paul Heymann, Daniel Ramage, and Hector Garcia-Molina. Social tag prediction. In *Proc. of the 31st Annual Intl. ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 531–538, New York, NY, USA, 2008. ACM.

[HRS06] Harry Halpin, Valentin Robu, and Hana Shepherd. The dynamics and semantics of collaborative tagging. In *Proc. of the 1st Semantic Authoring and Annotation Workshop*, 2006.

[JC97] Jay J. Jiang and David W. Conrath. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In *Proc. of the Intl. Conference on Research in Computational Linguistics*. Taiwan, 1997.

[JMH+07] Robert Jäschke, Leandro Balby Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. Tag recommendations in folksonomies. In *Proc. Knowledge Discovery in Databases: PKDD 2007, 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 4702 of *Lecture Notes in Computer Science*, pages 506–514, Berlin, Heidelberg, 2007. Springer.

[KEG+07] Georgia Koutrika, Frans Adjie Effendi, Zoltán Gyöngyi, Paul Heymann, and Hector Garcia-Molina. Combating spam in tagging systems. In *Proc. of the 3rd Intl. Workshop on Adversarial Information Retrieval on the Web*, pages 57–64, New York, NY, USA, 2007. ACM.

[KH08] Chanju Kim and Kyu-Baek Hwang. Naive bayes classifier learning with feature selection for spam detection in social bookmarking. In *Proc. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2008.

[KL07] Owen Kaser and Daniel Lemire. Tag-Cloud Drawing: Algorithms for Cloud Visualization. *ArXiv Computer Science e-prints*, March 2007.

[Kle99] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[Knu93] Donald Knuth. *The Stanford GraphBase: A Platform for Combinatorial Computing*. New York: ACM Press and Addison-Wesley Publishing Company, 1993.

[KSHS08] Beate Krause, Christoph Schmitz, Andreas Hotho, and Gerd Stumme. The anti-social tagger: Detecting spam in social bookmarking systems. In *Proc. 4th Intl. Workshop on Adversarial Information Retrieval on the Web*, pages 61–68, 2008.

[KWSS08] Bill Kules, Max L. Wilson, M. C. Schraefel, and Ben Shneiderman. From keyword search to exploration: How result visualization aids discovery on the web. Human-Computer Interaction Lab Technical Report HCIL-2008-06, University of Maryland, 2008. http://www.cs.umd.edu/localphp/hcil/tech-reports-search.php?number=2008-06.

[LA00] Anton Leuski and James Allan. Lighthouse: Showing the way to relevant information. *Proc. of the IEEE Symposium on Information Visualization (InfoVis)*, pages 125–130, 2000.

[LA04] Anton Leuski and James Allan. Interactive information retrieval using clustering and spatial proximity. *User Modeling and User-Adapted Interaction*, 14:259–288, 2004.

[Lag00] Krista Lagus. *Text Mining with the WEBSOM*. PhD thesis, Helsinki University of Technology, 2000.

[Lee99] Lie-Quan Lee. The high performance generic graph component library. Master's thesis, University of Notre Dame, 1999.

[LG98] Steve Lawrence and C. Lee Giles. Context and page analysis for improved web search. *Internet Computing, IEEE*, 2:38–46, 1998.

[LGZ08] Xin Li, Lei Guo, and Yihong Eric Zhao. Tag-based social interest discovery. In *Proc. of the 17th Intl. Conference on World Wide Web*, pages 675–684, New York, NY, USA, 2008. ACM.

[LHFH05] Ben Lund, Tony Hammond, Martin Flack, and Timo Hannay. Social Bookmarking Tools (II): A Case Study - Connotea. *D-Lib Magazine*, 11(4), April 2005.

[LHKK79] C. L. Lawson, R. J. Hanson, D. R. Kincaid, and F. T. Krogh. Basic Linear Algebra Subprograms for Fortran usage. *ACM Transactions on Mathematical Software*, 5(3):308–323, 1979.

[Lin98] Dekang Lin. An information-theoretic definition of similarity. In *Proc. 15th Intl. Conference on Machine Learning*, pages 296–304. Morgan Kaufmann, 1998.

[LKK04] Krista Lagus, Samuel Kaski, and Teuvo Kohonen. Mining massive document collections by the websom method. *Inf. Sci.*, 163(1-3):135–156, 2004.

[LKL93] Joon Ho Lee, Myoung Ho Kim, and Yoon Joon Lee. Information retrieval based on conceptual distance in is-a hierarchies. *Journal of Documentation*, 49(2):188–207, 1993.

[LNK03] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *Proc. 12th Intl. Conference on Information and Knowledge Management*, pages 556–559, New York, NY, USA, 2003. ACM.

[LW95] Fritz Lehmann and Rudolf Wille. A triadic approach to formal concept analysis. In G. Ellis, R. Levinson, W. Rich, and J. F. Sowa, editors, *Conceptual Structures: Applications, Implementation and Theory*, volume 954 of *Lecture Notes in Computer Science*. Springer, 1995.

[Mar06] Gary Marchionini. Exploratory search: from finding to understanding. *Commun. ACM*, 49:41–46, 2006.

[MCM09a] Benjamin Markines, Ciro Cattuto, and Filippo Menczer. Social spam detection. In *Proc. of the 5th Intl. Workshop on Adversarial Information Retrieval on the Web*, 2009.

[MCM+09b] Benjamin Markines, Ciro Cattuto, Filippo Menczer, Dominik Benz, Andreas Hotho, and Gerd Stumme. Evaluating similarity measures for emergent semantics of social tagging. In *Proc. of the 18th Intl. World Wide Web Conference*, 2009.

[MD06] Krishna P. Gummadi Alan Mislove and Peter Druschel. Exploiting social networks for internet search. In *Proc. 5th Workshop on Hot Topics in Networks*, Irvine, CA, 2006.

[Men05] F Menczer. Mapping the semantics of web text and links. *IEEE Internet Computing*, 9(3):27–36, May/June 2005.

[MFK06] David R. Millen, Jonathan Feinberg, and Bernard Kerr. Dogear: Social bookmarking in the enterprise. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems*, pages 111–120, New York, NY, USA, 2006. ACM.

[Mik05] Peter Mika. Ontologies are us: A unified model of social networks and semantics. In Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors, *Intl. Semantic Web Conference*, volume 3729 of *LNCS*, pages 522–536. Springer, 2005.

[Mis06] Gilad Mishne. Autotag: A collaborative approach to automated tag assignment for weblog posts. In *Proc. of the 15th Intl. Conference on World Wide Web*, pages 953–954, New York, NY, USA, 2006. ACM Press.

[MM09] Benjamin Markines and Filippo Menczer. A scalable, collaborative similarity measure for social annotation systems. In *To appear in Proc. Hypertext*, 2009.

[MME+06] Ana G. Maguitman, Filippo Menczer, Fulya Erdinc, Heather Roinestad, and Alessandro Vespignani. Algorithmic computation and approximation of semantic similarity. *World Wide Web*, 9(4):431–456, 2006.

[MMRV05] Ana Gabriela Maguitman, Filippo Menczer, Heather Roinestad, and Alessandro Vespignani. Algorithmic detection of semantic similarity. In *Proc. 14th Intl. Conference on World Wide Web*, pages 107–116, 2005.

[MN99] K. Mehlhorn and St. Näher. *The LEDA Platform of Combinatorial and Geometric Computing*. Cambridge University Press, 1999.

[MNBD06] Cameron Marlow, Mor Naaman, Danah Boyd, and Marc Davis. Ht06, tagging paper, taxonomy, flickr, academic article, to read. In *Proc. of the 17th Conference on Hypertext and Hypermedia*, pages 31–40, New York, NY, USA, 2006. ACM Press.

[MRM08] Ben Markines, Heather Roinestad, and Filippo Menczer. Efficient assembly of social semantic networks. In *Proc. 19th ACM Conference on Hypertext and Hypermedia*, 2008.

[MSM06a] Ben Markines, Lubomira Stoilova, and Filippo Menczer. Givealink: A social recommendation system. In *Proc. NetSci*, 2006.

[MSM06b] Ben Markines, Lubomira Stoilova, and Filippo Menczer. Implicit tagging using donated bookmarks. In *Proc. WWW2006, Collaborative Web Tagging Workshop*, 2006.

[MSM06c] Ben Markines, Lubomira Stoilova, and Filippo Menczer. Social bookmarks for collaborative search and recommendation. In *Proc. 21st National Conference on Artificial Intelligence*, 2006.

[MST08] Leandro Balby Marinho and Lars Schmidt-Thieme. Collaborative tag recommendations. *Data Analysis, Machine Learning and Applications*, 2008.

[Mun00] Tamara Munzner. *Interactive Visualization of Large Graphs and Networks*. PhD thesis, Stanford University, 2000.

[OFWB03] Joshua O'Madadhain, Danyel Fisher, Scott White, and Yan-Biao Boey. The jung (java universal network/graph) framework. Technical report, University of California, Irvine, 2003.

[PP07] John C. Paolillo and Shashikant Penumarthy. The social structure of tagging internet video on del.icio.us. In *Proc. of the 40th Hawaii Intl. Conference on System Sciences*, Losa Alamitos, CA, 2007. IEEE Press.

[RB89] Roy Rada and Ellen Bicknell. Ranking documents with a thesaurus. *Journal of the American Society for Information Science*, 40(5):304–310, September 1989.

[RBMM09a] Heather Roinestad, John Burgoon, Benjamin Markines, and Filippo Menczer. Incentives for social annotation. In *To appear in Proc. SIGIR2009 Demonstrations*, 2009.

[RBMM09b] Heather Roinestad, John Burgoon, Benjamin Markines, and Filippo Menczer. Incentives for social annotation. In *To appear in Proc. Hypertext*, 2009.

[RGMM07] A. W. Rivadeneira, Daniel M. Gruen, Michael J. Muller, and David R. Millen. Getting our head in the clouds: Toward evaluation studies of tagclouds. In *Proc. of the SIGCHI Conference on Human Factors in Computing Systems*, pages 995–998, New York, NY, USA, 2007. ACM.

[RIS+94] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstorm, and John Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proc. of ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186, Chapel Hill, North Carolina, 1994. ACM.

[RMBB89] Roy Rada, Hafedh Mili, Ellen Bicknell, and Maria Blettner. Development and application of a metric on semantic nets. *Systems, Man and Cybernetics, IEEE Transactions on*, 19(1):17–30, 1989.

[Roc02] Luis M. Rocha. Semi-metric behavior in document networks and its application to recommendation systems. In V. Loia, editor, *Soft Computing Agents: A New Perspective for Dynamic Information Systems*, pages 137–163. IOS Press, 2002.

[Roc05] Luis M. Rocha. Personal communication, 2005.

[RPM03] Andreas Rauber, Elias Pampalk, and Dieter Merkl. The som-enhanced jukebox: Organization and visualization of music collections based on perceptual models. *Journal of New Music Research*, 32:193–210, 2003.

[Sch06] Patrick Schmitz. Inducing ontology from Flickr tags. In *Proc. WWW2006, Collaborative Web Tagging Workshop*, May 2006.

[SCH08] James Sinclair and Michael Cardew-Hall. The folksonomy tag cloud: When is it useful? *J. Inf. Sci.*, 34(1):15–29, 2008.

[SHJS06a] Christoph Schmitz, Andreas Hotho, Robert Jäschke, and Gerd Stumme. Kollaboratives wissensmanagement. In Tassilo Pellegrini and Andreas Blumauer, editors, *Semantic Web - Wege zur vernetzten Wissensgesellschaft*, pages 273–290. Springer, 2006.

[SHJS06b] Christoph Schmitz, Andreas Hotho, Robert Jäschke, and Gerd Stumme. Mining association rules in folksonomies. In V. Batagelj, H.-H. Bock, A. Ferligoj, and A. Žiberna, editors, *Data Science and Classification: Proc. of the 10th IFCS Conference*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 261–270, Berlin, Heidelberg, 2006. Springer.

[SHM+05] Lubomira Stoilova, Todd Holloway, Ben Markines, Ana Maguitman, and Filippo Menczer. GiveALink: Mining a Semantic Network of Bookmarks for Web Search and Recommendation. In *Proc. KDD Workshop on Link Discovery: Issues, Approaches and Applications*, 2005.

[Shn03] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. *The Craft of Information Visualization: Readings and Reflections*, 2003.

[SJWR00] K. Sparck-Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: Development and comparative experiment. In *Information Processing and Management*, volume 2, pages 809–840, 2000.

[SKKR00] Badrul M. Sarwar, George Karypis, Joseph Konstan, and John Riedl. Analysis

of recommender algorithms for e-commerce. In *Proc. 2nd ACM E-Commerce Conference*, 2000.

[SKKR01] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Reidl. Item-based collaborative filtering recommendation algorithms. In *Proc. of the 10th Intl. Conference on World Wide Web*, pages 285–295, 2001.

[SL94] David E Stewart and Zbigniew Leyk. *Meschach : Matrix Computations in C*. Canberra : Centre for Mathematics and its Applications, Australian National University, 1994.

[SL99] Jeremy G. Siek and Andrew Lumsdaine. The matrix template library: Generic components for high-performance scientific computing. *Computing in Science and Engineering*, 1(6):70–78, Nov 1999.

[SLL02] Jeremy Siek, Lie-Quan Lee, and Andrew Lumsdaine. *The Boost Graph Library : User Guide and Reference Manual*. Addison-Wesley, 2002.

[SLY03] Brent Smith, Greg Linden, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. In *Internet Computing, IEEE*, 2003.

[SM95] Upendra Shardanand and Patti Maes. Social Information Filtering: Algorithms for Automating "Word of Mouth". In *Proc. ACM CHI'95 Conf. on Human Factors in Computing Systems*, pages 210–217, 1995.

[Sta08] Chris Staff. Bookmark category web page classification using four indexing and clustering approaches. *Lecture Notes in Computer Science*, 2008.

[Sur04] James Surowiecki. *The Wisdom of Crowds*. Doubleday, 2004.

[SvZ08] Börkur Sigurbjörnsson and Roelof van Zwol. Flickr tag recommendation based on collective knowledge. In *Proc. 17th Intl. Conference on World Wide Web*, pages 327–336, New York, NY, USA, 2008. ACM.

[SZL+08] Yang Song, Ziming Zhuang, Huajing Li, Qiankun Zhao, Jia Li, Wang-Chien Lee, and C. Lee Giles. Real-time automatic tag recommendation. In *Proc. of the 31st Annual Intl. ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 515–522, New York, NY, USA, 2008. ACM.

[THA99] Loren Terveen, Will Hill, and Brian Amento. Constructing, organizing, and visualizing collections of topically related web resources. *ACM Transactions on Computer-Human Interaction*, 6:67–94, 1999.

[TSMST08] Karen H. L. Tso-Sutter, Leandro Balby Marinho, and Lars Schmidt-Thieme. Tag-aware recommender systems by fusion of collaborative filtering algorithms. In *Proc. of the 2008 ACM Symposium on Applied Computing*, pages 1995–1999, New York, NY, USA, 2008. ACM.

[vA06] Luis von Ahn. Games with a purpose. *Computer*, 39(6):92–94, 2006.

[vAD08] Luis von Ahn and Laura Dabbish. Designing games with a purpose. *Communications of the ACM*, 51(8):58–67, 2008.

[Vel95] Todd Veldhuizen. Using C++ template metaprograms. *C++ Report*, 7(4):36–43, May 1995. Reprinted in C++ Gems, ed. Stanley Lippman.

[Vos07] Jakob Voss. Tagging, folksonomy & co - renaissance of manual indexing?, 2007. http://www.citebase.org/abstract?id=oai:arXiv.org:cs/0701072.

[Wal02] Jörg A. Walter. On interactive visualization of high-dimensional data using the hyperbolic plane. In *Proc. ACM SIGKDD Intl. Conference Knowledge Discovery and Data Mining*, pages 123–131, 2002.

[WF05] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2 edition, 2005.

[WZM06] Harris Wu, Mohammad Zubair, and Kurt Maly. Harvesting social knowledge from folksonomies. In *Proc. of the 17th Conference on Hypertext and Hypermedia*, pages 111–114, New York, NY, USA, 2006. ACM Press.

[XFMS06] Zhichen Xu, Yun Fu, Jianchang Mao, and Difu Su. Towards the semantic web: Collaborative tag suggestions. In *Proc. WWW2006, Collaborative Web Tagging Workshop*, 2006.

[ZWY06] Lei Zhang, Xian Wu, and Yong Yu. Emergent semantics from folksonomies: A quantitative study. *Journal on Data Semantics VI*, 2006.

Curriculum Vitae

Contact Information

Benjamin C. Markines
bmarkine@cs.indiana.edu
cs.indiana.edu/~bmarkine

Department of Computer Science
School of Informatics
Indiana University
Bloomington, IN 47401 USA

Research Interests

My primary research interests are in social Web applications, tagging systems, Web mining, network science, and social networks. Other interests include computer vision, high performance computing, and anything concerning the Web.

Education

- **Indiana University**, Bloomington, Indiana USA

  Ph.D. Computer Science, May 2009
  - Thesis: "Socially Induced Semantic Networks and Applications"
  - Advisor: Filippo Menczer

- **Northern Illinois University**, DeKalb, Illinois USA

  M.S. Computer Science, May 2001
  - Concentration in system design and implementation

  B.S. Computer Science, Cum Laude, December 1999

Professional Experience

- **International Business Machines**, San Jose, California USA
  Software Engineer **August 2001 to present (currently on educational leave)**
  Developed an application to help debug components of the z/OS. Software developer of robust network storage application solutions, i.e., disaster recovery scenarios. Lead developer on an application for performance analysis of enterprise storage servers.

- **Institute for Scientific Interchange Foundation**, Torino, Italy
  Researcher **August 2007 to August 2008**
  Developer of online tools for ranking authors based on the PACS dataset. Continue work on GiveALink.org and other Ph.D. student activities.

- **SBC Global Network, Ameritech**, Hoffman Estates, Illinois USA
  Software Engineer **May 2000 to August 2000**
  Designed and implemented a database for tracking tests of large billing applications.

Publications

- Benjamin Markines, Ciro Cattuto, Filippo Menczer, Dominik Benz, Andreas Hotho, Gerd Stumme. Evaluating Similarity Measures for Emergent Semantics of Social Tagging.
  Final version in the Proceedings WWW 2009 [MCM+09b]

- Benjamin Markines, Ciro Cattuto, Filippo Menczer. Social Spam Detection.
  Final version in the Proceedings AIRWEB 2009 [MCM09a]

- Benjamin Markines, Filippo Menczer. A Scalable, Collaborative Similarity Measure for Social Annotation Systems.
  To appear in Poster Proceedings HT 2009 [RBMM09b]

- Heather Roinestad, John Burgoon, Benjamin Markines, Filippo Menczer. Incentives for Social Annotation.
  To appear in Proceedings of SIGIR 2009 Demonstrations [RBMM09a]

- Heather Roinestad, John Burgoon, Benjamin Markines, Filippo Menczer. Incentives for Social Annotation.
  To appear in Poster Proceedings HT 2009 [MM09]

- Benjamin Markines, Heather Roinestad, Filippo Menczer. Efficient Assembly of Social Semantic Networks.
  Final version in the Proceedings HT 2008 (Nominated for Best Paper) [MRM08]

- Justin Donaldson, Michael Conover, Ben Markines, Heather Roinestad, Filippo Menczer. Visualizing Social Links in Exploratory Search.
  Final version in the Proceedings HT 2008 [DCM+08]

- Weixia Huang, Katy Börner, Bruce Herr, and Ben Markines. Cyberinfrastructure Shell (CIShell): An OSGi-Based Framework for the Integration of Datasets and Algorithms. EclipseCON 2007, Santa Clara, California
  `http://www.eclipsecon.org/2007/index.php?page=sub/&id=3931`

- Ben Markines, Lubomira Stoilova, and Filippo Menczer. Social Bookmarks for Collaborative Search and Recommendation.
  Final version in the Proceedings AAAI 2006 [MSM06c]

- Ben Markines, Lubomira Stoilova, and Filippo Menczer. Implicit Tagging using Donated Bookmarks.

Final version in the Proceedings WWW 2006 Workshop on Collaborative Web Tagging [MSM06b]

- Ben Markines, Lubomira Stoilova, and Filippo Menczer. GiveALink: A Social Recommendation System.
  Final version in the Proceedings NetSci Conference 2006, Bloomington, Indiana [MSM06a]

- Lubomira Stoilova, Todd Holloway, Ben Markines, Ana Maguitman, and Filippo Menczer. GiveALink: Mining a Semantic Network of Bookmarks for Web Search and Recommendation.
  Final version in the Proceedings KDD Workshop on Link Discovery: Issues, Approaches and Applications, 2005 [SHM+05]

Invited Talks

- Center for Computing Services (CCS) Colloquium, Bowie, Maryland.
- Theoretical Aspects and Models of Large, Complex and Open Information Networks, ISI Foundation, Torino, Italy.
- Satellite workshop: "Enhancing Social Interaction: Recommendation Systems, Reputation, P2P and Trust," European Conference on Complex Systems (ECCS07), Dresden, Germany.

Academic Experience

- **Indiana University**, Bloomington, Indiana USA
  Research Assistant **August 2005 to present**
  Researcher and lead developer of the GiveALink project. GiveALink is a public site that accepts bookmark donations from the Web community. The bookmarks are analyzed for recommendation, personalization, and visualization.
  `givealink.org`

  Research Assistant **January 2006 to December 2006**
  Researcher and software developer for the Network Workbench. A tool designed for network scientists to manage, analyze, and visualize networks.
  `nwb.slis.indiana.edu`

  Graduate Teaching Assistant **September 2003 to May 2005**
  Instructor of programming lab sections as well as holding office hours and grading for the second class in the IU Computer Science undergraduate curriculum.

- **Northern Illinois University**, DeKalb, Illinois USA
  Graduate Teaching Assistant **January 2000 to May 2001**
  Instructor for an introduction to computer science class using the C programming

language. Responsibility included proctoring exams, developing machine projects, and grading.

Undergraduate Fellowship **August 1999 to December 1999**
Responsibilities included grading quizzes, exams, and homework in addition to holding office hours.

Professional Service

- *Program Committee*
  International ACM Conference on Hypertext and Hypermedia (HT 2008, HT 2009)

- *Paper Reviews*
  National Conference on Artificial Intelligence (AAAI 2007)
  International Annual ACM SIGIR Conference (SIGIR 2006)

Major Projects

- *GiveALink*
  A social bookmarking site where users may donate bookmarks and find recommendations. The recommendations are created from a collection of bookmarks aggregated together to form a semantic similarity network.
  `www.givealink.org`

- *Parallel Graph Visualization*
  Design parallel graph algorithms for visualization.
  `www.boost.org/libs/graph/doc/index.html`

- *Augmented SimCity*
  Development of an augmented reality application based on SimCity. Evaluated technologies such as ARToolkit, jARToolkit, and GL4Java.

- *Fluency*
  Enable non-programmers to develop their own applications without knowledge of a computer programming language.
  `fluency.knownspace.org`

- *Topical Crawlers*
  Develop and evaluate topical crawlers using support vector machines, naïve bayes, and cosine similarity to classify Webpages.

- *Three dimensional Wavelet Compression*
  Applying Wavelet compression to a three dimensional dataset, i.e., a movie.

Honors and Activities

- Demos on sixearch.org and givealink.org at CSI Piemonte Complexity Conference, Torino, Italy.
- Networks and Complex Systems Open House 2006, Bloomington, Indiana. Presented current research to prospective graduate students and faculty.
- Intel International Science and Engineering Fair 2006. Indiana Convention Center, Indianapolis, Indiana. Introduce informatics and Web technologies to high school students from around the United States.
- Computer Science Graduate Student Association whip for the 2005/2006 school year.
- Red Hat Certified Engineer for Red Hat Linux 9 2003.
- Graduated Cum Laude 1999.
- Dean's List in Spring 1998, Fall 1999, Spring 2000, and Fall 2000.