An integrated approach to discover tag semantics

Antonina Dattolo
University of Udine
Department of Mathematics and Computer Science
Via delle Scienze, 206
33100, Udine - Italy
antonina.dattolo@uniud.it

Davide Eynard
USI - University of Lugano
ITC - Institute for Communication Technologies
Via Buffi, 13
6900, Lugano - Switzerland
davide.eynard@usi.ch

Luca Mazzola
USI - University of Lugano
ITC - Institute for Communication Technologies
Via Buffi, 13
6900, Lugano - Switzerland
luca.mazzola@usi.ch

ABSTRACT

Tag-based systems have become very common for online classification thanks to their intrinsic advantages such as self-organization and rapid evolution. However, they are still affected by some issues that limit their utility, mainly due to the inherent ambiguity in the semantics of tags. Synonyms, homonyms, and polysemous words, while not harmful for the casual user, strongly affect the quality of search results and the performances of tag-based recommendation systems.

In this paper we rely on the concept of tag relatedness in order to study small groups of similar tags and detect relationships between them. This approach is grounded on a model that builds upon an edge-colored multigraph of users, tags, and resources. To put our thoughts in practice, we present a modular and extensible framework of analysis for discovering synonyms, homonyms and hierarchical relationships amongst sets of tags. Some initial results of its application to the delicious database are presented, showing that such an approach could be useful to solve some of the well known problems of folksonomies.

Categories and Subject Descriptors

H.3.5 [**On-line Information Services**]: Web-based services; H.3.3 [**Information Search and Retrieval**]: Clustering.

General Terms

Algorithms, Experimentation, Measurement.

Keywords

Folksonomies, tag clustering, tag disambiguation, similarity.

1. INTRODUCTION

Folksonomies are the result of collaboratively organizing information through user-chosen, free-form metadata called *tags*: they are democratic and bottom-up, flat (as opposed to hierarchical), inclusive (meaning that if some resource does not fit existing tags it is sufficient to create a new one), current, and extremely easy to use. However, despite all their good points, tag-based systems suffer of different problems which are due to the inherent ambiguity in the semantics of terms.

As described in [8, 11], main ambiguities are due to the presence of synonyms (i.e. `game` and `juego`, or `web2.0` and `web_2`), homonyms (some of which may be *polysemous*, i.e. `check` as in "to check" and as in chess, while others might be not, as in `sf` for "San Francisco" or "Science Fiction"), and basic level variations (such as `dog` and `poodle`). Moreover, even when the meaning of a single word is well defined, the purpose of a tag still might vary from one tagging action to another: as an example, the tag `blog` could be applied to a blog service such as blogger.com, to a blog page, to a blog software, or to a piece of news that we want to blog later.

The motivations to better define the semantics of these tags are different: synonym detection increases recall in search, and can be used to refine results in recommendation systems; finding homonyms also means finding different contexts of use of the same tag, increasing the precision of returned results; basic level variations identify a hierarchy within a tag set that can be exploited to improve search.

The *main contribution* of this work is to introduce a holistic approach to the problem of disambiguation in tag-based systems; the choice of a *holistic* approach is based on some considerations: we start from the evidence that different types of ambiguity might coexist within the same tag set (i.e. the tag `sf` has synonyms as `sanfrancisco` and `scifi`, and it can be considered as a homonym for these two different concepts). Thus, instead of focusing on just one type of ambiguity within folksonomies, we decide to provide a way to analyze tags for different kinds of relations at the same time. Building on the ideas presented in [18], we start from the complex but expressive edge-colored multigraph of users, tags, and resources, and propose some simplified definitions of that graph that maintain some of its basic properties. This approach simplifies the work of extracting information from the structure and builds a basis for a set of simplified analysis methodologies which rely on a simple graph. From the technical point of view, our approach relies on a modular and extensible framework of analysis, allowing us to plug in new metrics or change some parameters to test if different setups produce better results. This information can then be used to improve tag-based recommendation systems and to support automatic suggestion of new resources

SAC'11 March 21-25, 2011, TaiChung, Taiwan.

based on matching between profile and user's interests (aka personalization process).

The paper is organized as follows: in Section 2 we describe related projects and the position of this paper among other works at the state of the art. In Section 3 we introduce some basic definitions and show the process we follow to create a simple graph starting from an edge-colored multigraph. In Section 4 our approach is described, together with our framework and its modular architecture. Section 5 reports on some experimental results, starting from the dataset we gathered up to some interesting conclusions based on the output produced by our analysis tool. Finally Section 6 presents some conclusions and future work.

2. RELATED WORK

The efforts to overcome limitations in tag-based systems can be roughly categorized in two main groups: one which is more directed towards the development of theoretical models, such as the tripartite graph [13, 17], and another which is more directed towards "ad-hoc" solutions, and whose aim is to solve a single issue (or a family of issues) in the most effective and convenient way (see, for instance, the FLOR system [1]). The former approach has the advantage of allowing researchers to better understand how folksonomies work and provides a common ground to describe their inner details. The latter, instead, offers a pragmatic and practical way to overcome specific problems.

In this second category, some authors concentrate on the disambiguation of tags with mutual contextualization [15], clusterization [16], and semantic grounding [5]; other algorithms instead are aimed at automatically restructuring folksonomies [3, 4, 12]. A more extensive work [7] presents a list of common issues in tag based systems related to their intrinsic ambiguity. Other, more general approaches exist: in [2] the authors present a methodology for extracting new semantic tags for a resource, using a domain ontology; in [9], the authors use multiple dimensions to enrich the tag information for final user support.

Even if the practical level may appear more efficient in attacking specific problems, some issues (such as homonymy, polisemy, synonymy, term variations and spelling errors) are not independent from each other and a global solution is sometimes the only feasible way to achieve good and reliable results. Especially for the objective of supporting recommender systems through tags [8], and aware of how deep the impact of tag ambiguities in folksonomies is within these systems [10], it appears that a holistic approach assumes an important role.

For this reason, our work builds both on general theories and ad-hoc solutions, proposing their integration in a unifying framework which is grounded on a model that builds upon the edge-colored multigraph of users, tags, and resources and adapts it for specific purposes.

3. FOLKSONOMIES AS EDGE-COLORED MULTIGRAPHS

In this section we introduce a new formal definition of folksonomy, simplifying the notion of tripartite graph, proposed in [13, 17] in the definition of a collaborative community, and largely used for folksonomy.

For consistency, we introduce the preliminary definition of edge-colored multigraph.

DEFINITION 1. *An* edge-colored multigraph *is a triple*

$$ECMG = (MG, C, c)$$

where $MG = (V, E, f)$ *is a multigraph composed of a set of vertices* V*, a set of edges* E *and a function*

$$f : E \rightarrow \{\{u, v\} \,|u, v \in V, u \neq v\}.$$

C *is a set of colors, and* $c : E \rightarrow C$ *is an assignment of colors to edges of the multigraph.*

DEFINITION 2. *A* personomy *related to a user* u *is an indirect edge-colored graph of color* C_u*. It is a four-tuple* $P_u = \{T, R, E, C_u\}$ *such that:*

- T *is the finite set of tags used by* u*;*
- R *is the finite set of resources tagged by* u*;*
- T *and* R *are disjoint;*
- E *is the set of non-ordered couple of vertices* (x, y) *:* $x \in T \wedge y \in R$*;*
- C_u *is the color of each edge in* P_u*.*

DEFINITION 3. *Given a set of users* U *and the family of personomies* P_u $(u \in U)$*, a* folksonomy *is*

$$F = \bigcup_{u \in U} P_u$$

In this way, a folksonomy is the edge-colored multigraph described by: the multigraph $((T \cup R), E, f)$, the set of colors $C = \cup_{u \in U} C_u$ and the assignment c. T, R, E, C are respectively the union of tags, resources, edges and colors present in the personomies.

Figure 1 shows an example of folksonomy. The set C is composed of three users, identified by three colors (normal line, bold line, and dashed line). The set R is composed of three resources (r_1, r_2, r_3), identified by three solid, round discs, while the set T is composed of 17 tags identified by empty circles, and labelled as t_i, with $i = 1 \ldots 17$. Finally, E contains non-ordered couple of vertices in R and in T.

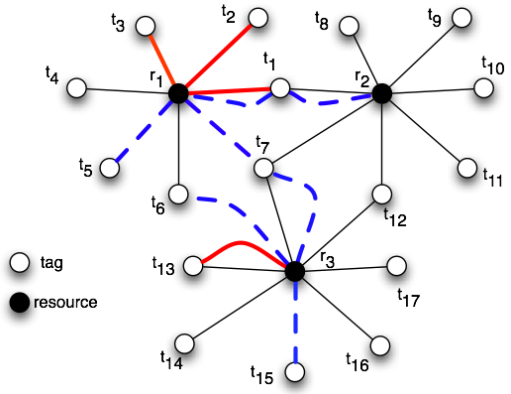

Figure 1: A folksonomy as a collection of personomies (each color represents a different user).

This view of a folksonomy as a collection of personomies includes all the information normally available in a social tagging system. Depending on the analyses that have to be

performed on the folksonomy, however, less expressive models might be sufficient. A first step towards simplification can be done with the injective transformation from Figure 1 to Figure 2, that is performed by grouping edges which connect the same vertices, and consequently losing information about whom applied a tag to a resource. The final weights to be assigned to the aggregated edges are calculated according to the following general formula:

$$w(r_i, t_j) = \sum_{u \in U} w_u(r_i, t_j) \qquad (r_i, t_j) \in E$$

This allows us the assignment of different weights according to which users have performed the tagging (i.e. the weight could be proportional to the user rank within a system, set higher for a community which is object of study, or lower for users identified as spammers). Our current choice is to assign the uniform weight 1 to all the users, reducing the calculation to the number of times a tag has been used on a resource (and, as a user can only use the same tag once for each resource, this also totals to the number of users applying a specific tag to a resource).

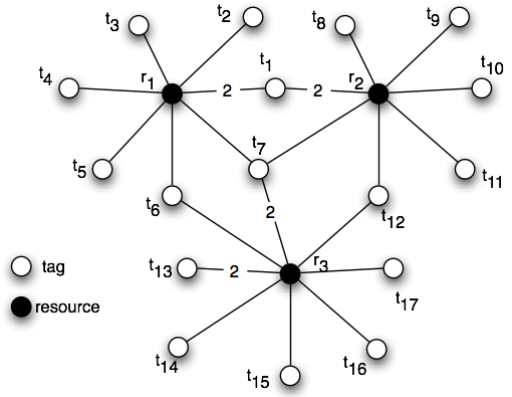

Figure 2: A bigraph as a weighted folksonomy: weights on edges represent the number of users using a specific tag for a resource.

This new bi-graph is useful to have a compact view of the relationships between resources and tags. As it can be noted by looking at Figure 2, it is also easy to identify which node $t \in T$ is more central and acts as a bridge between the different resources $r \in R$ (in this example, the tags t_6, t_7 and t_{12}). To further simplify the graph we can also collapse resources, creating a new link between tag t_a and t_b if and only if they are connected to the same resource r_i. The resulting graph is shown in Figure 3.

This kind of representation shows edges between tags if they have been used on the same resource, but it does not provide any information about the strength of these bonds. In this case, the weights that have to be assigned to the new edges are not a function of the previous weights anymore (or, at least, not a function of them alone). Different approaches have been taken trying to provide meaningful values for these weights. For instance:

- the number of triples (t_i, r, t_j) where (t_i, r), $(r, t_j) \in E$ represents the number of co-occurrences across all the resources (that is, the number of resources on which the tags co-occur): this is also what is commonly defined as co-occurrence. The main advantage of this metric is that it is simple to calculate and the results are meaningful. Its main limit, instead, is that it is biased towards very popular tags, as they frequently co-occur with many others;

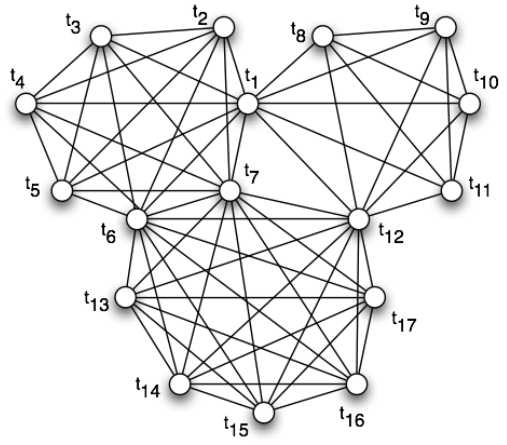

Figure 3: A graph of tagging relationships.

- a *normalized co-occurrence* takes into account not only the number of resources on which two tags co-occur, but also how common these tags are within the system. An example is the *Jaccard index*, defined as follows:

$$NCO = \frac{|A \cap B|}{|A \cup B|}$$

where A is the set of documents tagged with t_a and B is the set of documents tagged with t_b. This metric is more complex but it also returns much better results: tags which are less popular but still shared within subcommunities tend to be ranked higher, exposing the vocabularies typical of their domains of interest;

- more advanced metrics [5] take into account other parameters: for instance, distributional measures are based on vector space representations of tags, requiring information related to other tags, resources, or users. The limits of this approach are the same ones of many vector-based representations (i.e. high dimensionality, small distances between vectors, higher computational complexity). The results, however, are interesting as they tend to represent more "semantically similar" terms, that are related not because they appear *together* with a given tag but rather in the same *contexts*.

In this work, in order to express the strength of relationships between tags, we calculate the weight of each edge between tags in the following way:

$$\overline{w}(t_i, t_j) = \sum_{k|(t_i, r_k) \in E \wedge (r_k, t_j) \in E} w(r_k, t_i) * w(r_k, t_j)$$

We apply the product of weights for each couple of tags sharing a common resource; then we apply the sum of pre-

vious weights for all the couples of tags sharing more than one resource. The result, applied on Figure 2, is shown in Figure 4, where the different weights (1, 2, 3, and 4) are

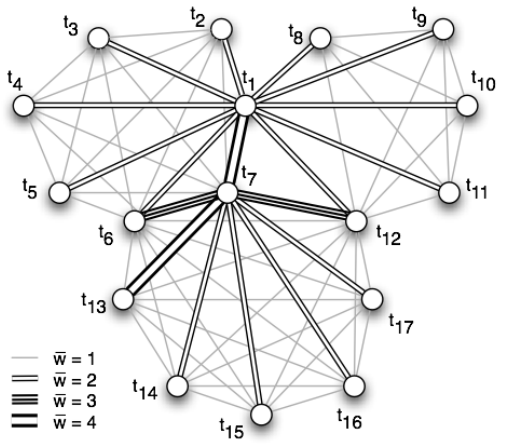

Figure 4: An alternative view of the tagging relationship, with the $\overline{w}$ measure applied on the edges.

visualized using different line styles (see the legend). In particular, we note, for example, that t_6 and t_7 (see Figure 2) share two resources (r_1 and r_3); so, the weight associated to (t_6, t_7) and related to the only resource r_1 (resp. r_3) is $w(r_1, t_6) * w(r_1, t_7) = 1$ (resp. $w(r_1, t_6) * w(r_1, t_7) = 2$). Consequently, $\overline{w}(t_6, t_7) = 3$.

4. OUR APPROACH

As discussed in the last Section 3, folksonomies can be represented at different levels of expressiveness. Our steps have been directed towards graph simplification for reducing the complexity (in terms of tags) for finding synonyms, homonyms, or hierarchies between them. In fact, also considering the fact that tags are positively correlated with users [20], typical comparison algorithms do not scale well on the original graph. Starting from the assumption that ambiguous tags should at least be related (either by cooccurrence or by presence in the same contexts), we restrict our analyses to the top n tags which are related to a given one.

The architecture of our framework is shown in Figure 5. Inspired by the approach described in [19], we decided to make the system modular in order to perform different kinds of analyses on the tag corpus: new algorithms should be easily pluggable into the system, but at the same time it should be possible to group common operations together in the same modules.

The three main components are the Tag Analysis Tool, the Disambiguation Tool, and the Front-End. The *Tag Analysis Tool* accesses a folksonomy (that we assume stored inside a database), calculates relatedness between tags according to chosen metrics, and finally outputs its results in different matrices. These matrices can be saved as tables in a DB or as CSV files ready to be imported by other tools. The *Disambiguation Tool* is the core component of our system: it takes as inputs a tag t, the number of top-related tags n to take into account, and the similarity matrices generated by the Tag Analysis tool, and provides new information about t and the set of its related tags. This information is generated by the disambiguation plugins that are installed into the system: some possible results are synonymy relationships between tags, partitions of a tag set according to the different acceptations of a term, or a hierarchy between tags. Details about how this information can be generated are presented in the following paragraphs.

As the output of disambiguation plugins is heterogeneous, the system also needs a component which is able to manage it, showing data to users in a meaningful way or exporting it in a format which is suitable for other applications. This is what the *Front-End* component is in charge of, that is basically managing all the system inputs and outputs.

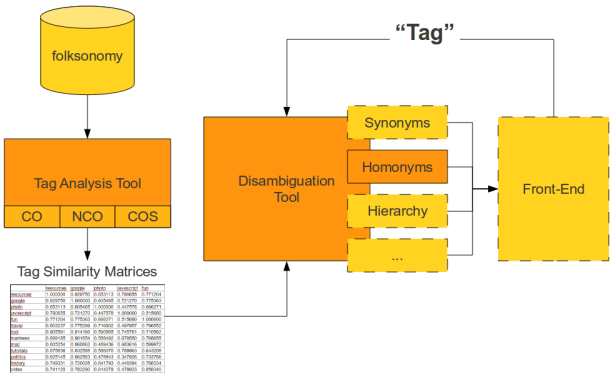

Figure 5: The architecture of our system.

As discussed in Section 2, literature provides algorithms that can be used to find synonyms, disambiguate tag contexts, and categorize them in hierarchies. In the following sections we show some approaches that build upon these algorithms and that could be easily integrated into our architecture.

4.1 Synonym Detection

Two words in tag-based systems are considered synonyms if -similarly to natural text- they can be replaced by each other without affecting the meaning of a "sentence" (in our case a tagging action). What is different from natural language is that the different words can be variations of the same one (as in `blog`, `blogs`, and `blogging`), translations into other languages, sets of terms joined by non-alphabetic characters, and so on. The consequence of this heterogeneity is that there is no "one size fits all" solution to the problem of detecting whether two words are synonyms.

In our system we have chosen different heuristics for synonym detection. Each of them returns the likelihood of two tags to be synonyms, then the results are weighted to obtain an overall likelihood. The heuristics we have decided to use are:

- an **edit distance** such as Levenshtein distance, which calculates differences between two strings in terms of insertion, deletion, and substitution of characters. This would return high similarities for variations of the same word obtained by inserting extra characters to simulate spacing (as in `web20`, `web2.0`, and `web_2_0`), but might return wrong results especially when short strings are

involved (to solve this problem a Levenshtein distance normalized with respect to string length could be used);

- synonym search within **WordNet**. Results are good in terms of precision, however the high-quality, top-down thesaurus provided by WordNet only covers a small part of the bottom-up vocabulary built with a folksonomy, which often contains typos, slang, acronyms, and so on;

- **online translation** tools. Online services can be used to obtain the translation of a tag in different languages. As fixed vocabularies suffer the same problem of WordNet, collaboratively grown corpora can be chosen instead to get translations for recent terms. As an example, Wikipedia articles usually contain links to their translated versions. For every link, both the translated word and the language code for that translation are published in a semi-structured way;

- **stemming**. NLP (Natural Language Processing) tools can be used to detect if two tags share the same stem, defined as the part of a word that is common to all its inflected variants. If so, then there is a higher probability that these tags are synonyms.

It is worth to be noted that one more heuristic is also somehow "embedded" into the system, whenever top related tags are calculated using Tag Context Similarity. In fact, by using co-occurring tags to define the context, this metric tends to consider related those tags which have been attributed a similar meaning.

4.2 Homonym Detection

Checking for homonyms basically means verifying if the same tag has been used in different contexts. Typically, usage contexts for a tag t are computed by clustering tags related to t in groups, according to a chosen measure of similarity. Then, the tags most frequently occurring in these groups can be used as a way to name and disambiguate the different contexts of use of t (see [15]).

Clustering algorithms vary in performance and in the type of discovered clusters. What we need in our case is an algorithm which only relies on relatedness measures between tags and which is able to find overlapping clusters: this is particularly useful as other tags in the context could be homonyms too and we do not want to lose the information they might bring to the semantics of a context.

The algorithm we have chosen for our first prototype has been described in [14]. The basic idea is that a community is a subgraph G identified by the maximization of a property of fitness of its nodes. The fitness of G is calculated as follows:

$$f_G = \frac{s_{in}^G}{(s_{in}^G + s_{out}^G)^\alpha}$$

where s_{in}^G and s_{out}^G are, respectively, the *strength* of the internal links (i.e. double the sum of the weights of all the edges that link the nodes in G with each other) and the strength of the external ones (i.e. the sum of the weights of all the edges linking nodes in the group with nodes not belonging to it). The parameter α can be used to tweak the groups' sizes: large values yield very small groups, while small ones bring larger clusters.

Item	Count
Users	30,948
Tags	482,465
Resources	3,744,679
Assignments	21,698,526

Table 1: Dataset details.

4.3 Hierarchy Detection

Hierarchy is a specific case of basic level variation. To detect the $is-a$ relationships that allow us to build a hierarchy from a flat set of tags we have chosen to apply the "Hearst Patterns on the Web" method [6]. Hearst Patterns are a set of patterns against which we can test a pair of terms to find instance-of and subconcept relationships. If we use I to refer to an instance and (C_1, C_2) to refer to two distinct classes, some examples of these patterns are *"C_1 (and|or) other C_2"* (as in "poodles and other dogs") and *"C_1 such as I"* (as in "cities such as San Francisco").

The Hearst Patterns on the Web technique uses the Web itself as a source of knowledge, looking for patterns on a search engine and taking the number of web pages in which a certain pattern appears as an indicator for the strength of the pattern. The main advantage of this approach is that the Web is an always growing and up-to-date knowledge base, just as the the tag vocabularies we want to study. Its main drawback, instead, is that the $O(n^2)$ complexity of the comparison task means a large amount of queries to search engines, making the approach not scalable to large sets of tags. Even if we work with limited sets (the top n related tags), we have designed a caching module to make the system perform better.

5. EXPERIMENTAL RESULTS

In order to perform our experiments, our first step has been to build a dataset by downloading contents from delicious[1]. This has been done by creating a Web scraper able to parse user pages and running it over a set of more than 30,000 users. The resulting data have been saved inside a database which is then accessed by our Tag Analysis tool. As second step, we have implemented the prototype. Albeit the system is not complete yet, the modules that have currently been implemented are already capable of providing some interesting results. So, we have tested the tool with some tags which are known to be ambiguous and have gathered the results to show our system's performance. The details of this whole process are described in the following paragraphs.

5.1 Preliminar Data Analysis

Before the real analysis, we describe the coverage of our delicious DB dump in terms of number of tags, resources, and tagging events. Data has been gathered by downloading contents from more than 30,000 user accounts and resulted in more than 480,000 tags, 3.7 million resources, and 21 million tag assignments (see Table 1). We decided to exclude from our dataset the tag `system:unfiled` that is generally used for bookmarks that have been added to the system but have not been tagged[2]. On this dataset, we ran the Tag Analysis tool, pre-calculating similarities between tags.

[1] www.delicious.com

[2] Actually, `system:unfiled` is the most frequent tag in deli-

In order to calculate Tag Context Similarity, we represented tags as vectors of co-occurring tags. The size of the vector is one more parameter to the system: the bigger the size, the better the results. Of course, a bigger size also means a slower system and the need for more storage space. Our choice fell on the top 10,000 tags, which provide experimentally good results but still can be analyzed in a reasonable time.

5.2 Our Prototype

The components of our system that have already been implemented in the prototype are the ones shown in full lines in Figure 5. For the Tag Analysis tool the metrics that we have currently implemented are co-occurrence (CO), normalized co-occurrence (NCO) according to the Jaccard index, and Tag Context Similarity as defined in [5]. Usually calculating these similarity matrices takes a long time, so the Tag Analysis component has been built to run as a batch job. The tool also allows one to split the tag space in slices that can be analyzed independently, making it possible to perform the actual computation with different processors (or even computers, provided they can all access the DB).

The Disambiguation tool has been developed with the homonyms plugin, implementing the algorithm described in [14] and showing related tags as a set of overlapping clusters matching the different tag contexts. As a part of the synonyms plugin, Wikipedia synonym discovery downloads translations for a tag from its matching Wikipedia article (if it exists). The front-end is currently a text-only application that runs on a UNIX shell, allowing users to specify different parameters (i.e tag to analyze, size of the related tag set, number of top tags to take into account) and returning results in different formats.

The prototype has been written using scripting languages (mostly Perl, plus few parts in PHP) and makes use of available libraries for external data access (DBI for local and remote databases, LWP to retrieve Web contents, and a customized version of WWW::Wikipedia to access Wikipedia pages).

5.3 Results of tag analysis

We tested the system against different sets of tags: the top 20 tags in delicious; a group of tags known to be ambiguous (`apple`, `cambridge`, `sf`, `stream`, `turkey`, `tube`), and a set of subjective tags –chosen between the most popular ones in delicious– that are used transversally, across different domains (`cool`, `fun`, `funny`, `interesting`, `toread`).

For each tag, we calculated the top n (with $n = 50$) related tags with the three metrics we described previously and performed synonym and homonym analyses. A first result is that, even when used alone, Tag Context Similarity already tends to provide synonyms as top-related tags. For instance, looking at the top 50 tags related to `toread`, we see that it can also be spelled as `read`, `read_later`, and `to_read`. Repeating the analysis with a less popular term with the same meaning (`@readit`), the list of synonyms grows covering 9 out of the top 10 tags, and 17 out of the top 50. We attribute this different behavior to the fact that, as these tags are used transversally, being less popular also means covering less contexts, increasing the similarity with other less popular synonyms.

cious: almost 36 million occurrences, that is about the triple of `design`, which is the second most frequent tag

Context	Tags
Science Fiction	(scifi, Sci-Fi, starwars, sciencefiction, Science-Fiction, cyberpunk, Trek, Fantasy, fandom, genre, horror, movies, sf)
Literature	(writing, literature, ebooks, ebook, book, books, reading, culture, science)
San Francisco	(SanFrancisco, California, bayarea, san_francisco, san-francisco)

Table 2: Clustering results for the tag `sf`.

The Wikipedia plugin returns translations for tags whose name directly match an English Wikipedia article (we currently skip disambiguation pages to avoid adding more ambiguity). Experimentation with this tool is still at a preliminary phase. From the results returned by Wikipedia, we select only the ones which actually match existing tags so they can be used to test related tags for synonymy. Analyzing the 31 tags in our three sets, we were able to get 215 new words from Wikipedia. Of those 215, only 83 are valid tags in our delicious dataset and 20 of them belong to the 10,000 most used tags. Out of these 20, there are only two translations that belong to the set of top-related tags of their English synonym (`musica` and `musik` for the tag `music`). However, we found that the same translations (together with `musique`) are also present in the set related to the tag `stream` (see Table 3). This is compatible with the hypothesis that synonyms belong to similar contexts, and the fact that contexts can be identified in different ways.

A way to identify contexts in our system is by clustering related tags. To test our clustering algorithm, we executed it on the set of ambiguous tags with different values of α. As a positive note, we were able to find meaningful clusters and, in cases like the ones shown in Table 2 and 3, to easily classify them according to different contexts of use. In Table 2 we show the results of clustering tags related to `sf` with $\alpha = 1.4$. Out of the 11 groups that the algorithm was able to detect, we identified three main contexts of usage (science fiction, literature, and San Francisco). The main (biggest) group for each context is shown.

In Table 3 we show the results for the `stream`, with $\alpha = 1.74$. Here, the main contexts we were able to spot are

Context	Tags
Video	(streaming, multimedia, television, broadcast, iptv, player, broadcasting, podcasts, webtv, webcast, videos, content, watch, sharing, clips, sites, share, firefox:bookmarks, shows, audio, firefox:toolbar, en, live, channel, recommendations)
Audio	(Musik, mp3, radio, audio, sound, mp3s, podcasts, musica, Musique), (stream, streams, webradio, internetradio, radios, tunes, broadcasting, music, muziek, Jukebox, station, mediaplayer, winamp, discovery, channel, onlinemusic, playlist, webtv, listen)

Table 3: Clustering results for the tag `stream`.

video and audio. Note that the tag `audio` is present in both groups, as of course the concept is present also in video streaming. From the performance point of view, as the clustering algorithm worked only on the top n related tags, the

tool was able to return results in few seconds and so we could test it in realtime. As a negative note, we found that sometimes the values of α still need to be trimmed manually. This is due to the fact that we are not analyzing the network as a whole, but we are only focusing on the subgraph involving the top n related tags, which changes from one tag to another and whose characteristics might differ from the others.

After both the experiments, results confirmed us that synonyms and homonyms are strongly related: disambiguating a tag allows us to define its different contexts of use, and within each context we have more chances to find synonyms.

6. CONCLUSION AND FUTURE WORK

Trying to overcome some intrinsic limits of folksonomies, we have developed a model which simplifies the tripartite graph by focusing only on relations between tags. Then, we have designed a system that relies on this model to analyze sets of related tags and detect synonyms, homonyms, and hierarchical relationships amongst them.

The infrastructure we implemented is highly modular and allows users to plug-in new metrics and modify some working parameters to foster the power of different algorithms in discovering relationships, creating meaningful clusters and suggest hierarchies between tags.

The prototype used for the present work is yet partial, but allowed us to perform some tests, whose results seem to be interesting and promising. In our future work we are going to proceed following two main directions: on the one hand, we are planning to implement all the modules shown in Figure 5 to provide a fully functional system; on the other hand, we want to extend our tests to a more extensive dataset [20] and, relying on the modularity of the tool, compare our results with the ones provided by already existing solutions.

7. REFERENCES

[1] S. Angeletou. Semantic enrichment of folksonomy tagspaces. In *ISWC 2008*, volume 5318 of *LNCS*, pages 889–894. Springer, 2008.

[2] A. Baruzzo, A. Dattolo, N. Pudota, and C. Tasso. Recommending new tags using domain-ontologies. In *Web Intelligence/IAT Workshops*, pages 409–412. IEEE, 2009.

[3] G. Begelman, P. Keller, and F. Smadja. Automated tag clustering: Improving search and exploration in the tag space. In *Proc. of Collaborative Web Tagging Workshop at WWW2006*, Edinburgh, Scotland, 2006.

[4] C. H. Brooks and N. Montanez. Improved annotation of the blogosphere via autotagging and hierarchical clustering. In *Proc. of the 15th International Conference on World Wide Web (WWW2006)*, pages 625–632, Edinburgh, Scotland, 2006.

[5] C. Cattuto, D. Benz, A. Hotho, and G. Stumme. Semantic grounding of tag relatedness in social bookmarking systems. In *ISWC 2008*, volume 5318 of *LNCS*, pages 615–631. Springer, 2008.

[6] P. Cimiano and S. Staab. Learning by googling. *SIGKDD Explor. Newsl.*, 6(2):24–33, 2004.

[7] A. Dattolo, F. Ferrara, and C. Tasso. The role of tags for recommendation: a survey. In *Proc. of the 3rd International Conference on Human System Interaction - HSI'2010*, pages 548–555, Rzeszow, Poland, May 2010. IEEE press.

[8] A. Dattolo, F. Tomasi, and F. Vitali. Towards disambiguating social tagging systems. In S. Murugesan, editor, *Handbook of Research on Web 2.0, 3.0 and X.0: Technologies, Business, and Social Applications*, chapter 20, pages 349–369. IGI Global, Hershey, PA, 2010.

[9] A. Dix, S. Levialdi, and A. Malizia. Semantic halo for collaboration tagging systems. In *Proc. of the Workshop on the Social Navigation and Community based Adaptation Technologies*, 2006.

[10] J. Gemmell, M. Ramezani, T. Schimoler, L. Christiansen, and B. Mobasher. The impact of ambiguity and redundancy on tag recommendation in folksonomies. In *Proc. of the third ACM conference on Recommender systems (RecSys '09)*, pages 45–52, New York, NY, USA, 2009. ACM.

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

[12] M. Grahl, A. Hotho, and G. Stumme. Conceptual clustering of social bookmarking sites. In *Proc. of 7th International Conference on Knowledge Management (I-KNOW '07)*, pages 356–364, Graz, Austria, Sept. 2007. Know-Center.

[13] R. Lambiotte and M. Ausloos. Collaborative tagging as a tripartite network. In *Computational Science ICCS 2006*, volume 3993 of *LNCS*, pages 1114–1117. Springer Berlin / Heidelberg, 2006.

[14] A. Lancichinetti, S. Fortunato, and J. Kertesz. Detecting the overlapping and hierarchical community structure of complex networks. *ArXiv E-prints*, Feb. 2008.

[15] C. man Au Yeung, N. Gibbins, and N. Shadbolt. Understanding the semantics of ambiguous tags in folksonomies. In *Proc. of the International Workshop on Emergent Semantics and Ontology Evolution (ESOE2007) at ISWC/ASWC2007, Busan, South Korea, November*, 2007.

[16] C. man Au Yeung, N. Gibbins, and N. Shadbolt. Contextualising tags in collaborative tagging systems. In *Proc. of the Twentieth ACM Conference on Hypertext and Hypermedia (HT '09)*, New York, NY, USA, July 2009. ACM.

[17] P. Mika. Ontologies are us: A unified model of social networks and semantics. In *ISWC 2005*, LNCS, pages 522–536. Springer, 2005.

[18] C. Schmitz, A. Hotho, R. Jäschke, and G. Stumme. Mining association rules in folksonomies. In *Proc. IFCS 2006*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 261–270, Berlin/Heidelberg, July 2006. Springer. Ljubljana.

[19] L. Specia and E. Motta. Integrating folksonomies with the semantic web. In *Proc. of the European Semantic Web Conference (ESWC2007)*, volume 4519 of *LNCS*, pages 624–639, Berlin Heidelberg, Germany, July 2007. Springer-Verlag.

[20] R. Wetzker, C. Zimmermann, and C. Bauckhage. Analyzing social bookmarking systems: A del.icio.us cookbook. In *Proc. of Mining Social Data (MSoDa) Workshop at ECAI 2008*, pages 26–30, July 2008.