Mining the Structure of Tag Spaces for User Modeling

Eric Schwarzkopf, Dominik Heckmann, Dietmar Dengler, and Alexander Kröner

German Research Center for AI (DFKI)
66123 Saarbruecken, Germany
`Firstname.Lastname@dfki.de`

Abstract. We propose an approach for using data from a social tagging application like `del.icio.us` as a basis for user adaptation. We discuss several algorithms for mining taxonomies of tags from tag spaces. The mined taxonomy can be used to define adaptation rules that determine how to adapt a system to a user given the user's personal tag space.
The contributions of this work are a description of an application scenario for taxonomy-mining algorithms, a discussion of algorithms by Mika[3], Heymann et al.[2], and Schmitz et al.[4], and the proposal of an extension to the algorithms that takes the contexts of tags into account when building a taxonomy. We look at the performances of the algorithms on a dataset retrieved from `del.icio.us` and give a tentative recommendation of what algorithm to use.

1 Introduction

Tag spaces are an obvious source of data for user modeling. The user of a social tagging tool could provide access to his personal tag space to an e-commerce site which could use the data to tailor its structure and presentation to the user. For example, a book store could determine that a customer who uses the tags *code*, *java*, and *mysql* frequently is most likely a programmer and recommend the most popular programming books.

How can we use a tag space and a user's tagging data to create a user model and adapt a system? The first step in the approach we are proposing is mining a taxonomy of tags from the tag space. The system engineer then creates a set of application-specific adaptation rules based on the mined taxonomy. Finally, a user's personal tags are mapped into the taxonomy to determine which adaptation rules apply to the user. This process is depecited in figure 1.

Not all tag spaces are suitable for this type of user modeling. Because we want to learn something about the user's interests, we require tagging data used by the user for himself (as in `del.ico.us`) and not for others (as in `flickr`).

In a taxonomy of tags, subtags of a tag are specializations of the tag (for example, *pop-music* should be a subtag of *music*). Given a taxonomy of tags, we can compute a value for the association between a user u and a tag t by computing the similarity between the set of tags used by u and the set containing t and all of t's subtags by using, for example, Jaccard's coefficient. For the designer of an adaptive system a taxonomy simplifies identifying the semantics of a tag (by using its predecessors and successors as context) and its generality (the higher it is in the taxonomy, the more users will be associated with it). Hence, we think a taxonomy is a good underlying structure for the task at hand.

Mining a taxonomy from a tag space is the main subject of this paper. We will look at several taxnonomy-mining algorithms proposed in the literature, evaluate their performance on a dataset retrieved from `del.icio.us`, and, based on our results, give a tentative recommendation of what algorithm to use when mining taxonomies.

We do not focus on privacy issues in this paper, but since they are of special relevance in this domain, we provide some ideas that can be implemented easily on top of existing tagging systems. To limit the possibility of misuse, the user should restrict access to his data. To that end, a user could maintain several profiles, where a profile is a subset of the user's tagged resources. For instance, there could be a job profile for data collected in the user's professional role and a personal profile for data related to the user's hobbies. There are a number of ways to create profiles easily — and it has to be easy because otherwise the user

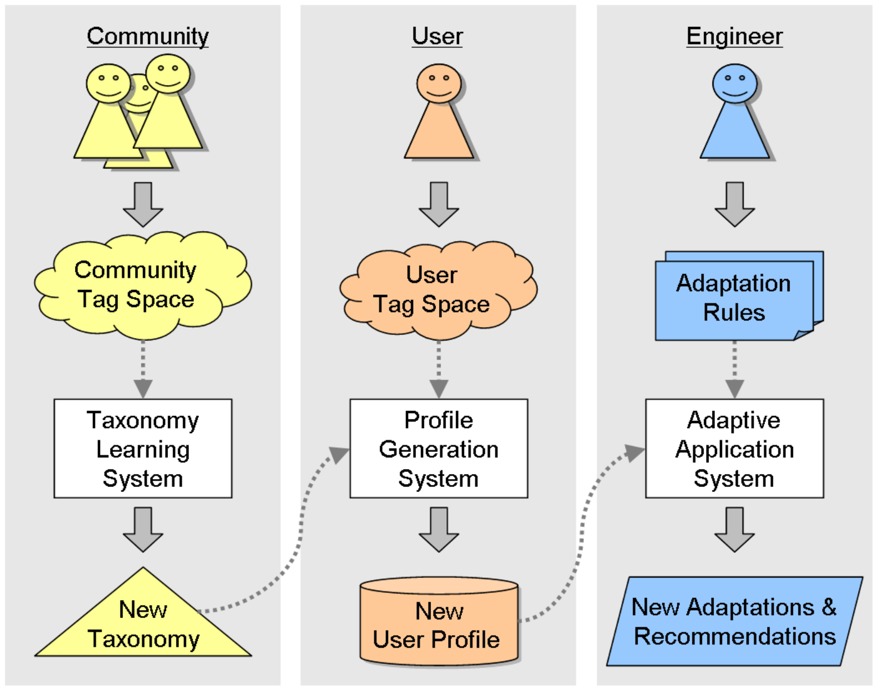

Fig. 1. Overview of the adaptation process.

could just as well fill in a questionnaire to create his profile. One is to associate a specific tag with a profile, so that all entries using the tag are automaticaly assigned to the profile. For each profile, the user should be able to create a snapshot in time, for example, resources tagged within the previous two weeks, and to provide only this snapshot to a third party system. Current tagging system only provide for an all or nothing decision: Once a third party knows the account name of the user, it can retrieve all current and future data in that account. But by using the APIs offered by most social tagging services, it is possible to create the described profile service even without modifying the existing services (for example, by building upon a user+tag RSS feed from `del.icio.us`).

2 Mining Taxonomies

2.1 Test Data

To gain some experience with the algorithms discussed below, we applied them to data retrieved from `del.icio.us`. We collected 2553 account names by periodically polling the RSS feed of recent additions and then downloaded for each account the 100 most recently added bookmarks with the associated tags, resulting in a set of 125801 distinct bookmarks, 158870 user-bookmark pairs (most users had collected less than 100 bookmarks), 23334 shared bookmarks (bookmarks collected by more than one user), and 37697 distinct tags.

The approach taken to sampling accounts resulted in a set of users with diverse interests but low overlap in annotated bookmarks. These characteristics have to be taken into account when interpreting the results of our experiments.

2.2 Estimating the Quality of a Taxonomy

In order to compare the performance of the taxonomy-mining algorithms, we need some kind of measure of the quality of the mined taxonomies. For this paper, we use the structure of a taxonomy to assess their quality: Given the characteristics of our test data, we assume that a taxonomy is of low quality if its maximum depth is larger than 5 or most of the tags are on the first level of the taxonomy. A depth larger than 5 indicates that a large number of users have a common, very specific interest, while a flat taxonomy indicates that there are only very few subsumption relationships between tags. Both conditions are very unlikely for our dataset, and manual inspection of some of the mined taxonomies confirmed that an average depth of 3 with most of the tags on levels 1 and 2 is to be expected of a high-quality taxonomy.

The appendix lists the structure of taxonomies generated by the discussed algorithms with different parameter settings.

2.3 Algorithms

Before talking about algorithms for learning taxonomies, we define formally what we mean by a tag space. The following definition is adopted from Mika [3] and is used throughout the subsequent discussion.

> A **tag space** is a hypergraph $H :=< V, E >$. The set of vertices V is the union of three disjoint sets A, C, and I representing the set of users (actors), the set of tags (concepts), and the set of annotated objects (items). E is the set of ternary edges $\{\{a, c, i\} \,|\, \text{user}\, a \,\text{labels object}\, i \,\text{with tag}\, c\}$.

Our intention is to discover subsumption relationships between tags as seen by the user community. We say a tag t subsumes a tag u if and only if the intension of t properly contains the intension of u. That is, t subsumes u if all imaginable objects that could be sensibly tagged with u can also be sensibly tagged with t.[1]

Mika [3] looks at the weighted graphs $O_{ac} :=< C, E_{ac}, w_{ac} >$, where C is the set of tags, $E_{ac} := \{(x, y) \,|\, x,\, y \in C, \exists a \in A\, \exists i,\, j \in I : \{a, x, i\},\, \{a, y, j\} \in E\}$, $w_{ac}((x, y)) := |\{a \in A \,|\, \exists i,\, j \in I : \{a, x, i\},\, \{a, y, j\} \in E\}|$, and $O_{ic} :=< C, E_{ic}, w_{ic} >$, where $E_{ic} := \{(x, y) \,|\, x,\, y \in C, \exists i \in I\, \exists a,\, b \in A : \{a, x, i\},\, \{b, y, i\} \in E\}$ and $w_{ic}((x, y)) := |\{i \in I \,|\, \exists a,\, b \in A : \{a, x, i\},\, \{b, y, i\} \in E\}|$. That is, O_{ac} is the graph of tags in which an edge between two tags is weighted by the number of people who have used both tags, while O_{ic} is the graph of tags in which an edge is weighted by the number of resources that have been tagged with both tags. Mika suggests using O_{ic} for concept mining by applying graph clustering; he reports mining cohesive groups of concepts from `del.icio.us` data using α-set analysis. Because O_{ic} does not reflect how popular tags are in the user community (local structure can be determined by very few users), he uses O_{ac} to discover taxonomic relationships between tags using a set-theoretic approach that corresponds to mining association rules as is described further below when we look at the approach proposed by Schmitz et al.[4]. The idea is that the community of users associated with a narrower tag is a sub-community of the community associated with the broader tag.

Heymann at al.[2] create for each tag t a vector representation $v_t := (w(t, i))_{i \in I}$, where $w((c, i)) := |\{a \,|\, (a, c, i) \in E\}|$, and then define a tag similarity graph $S :=< C, E_s >$, where $E_s := \{(a, b) \,|\, a,\, b \in C, \cos(v_a, v_b) > d\}$ with cos denoting the cosine similarity and d a predefined threshold. Note that in general, S does not correspond to Mika's O_{ic}, because the latter uses the overlap in tagged resources to determine the weight of an edge while the cosine similarity measures how similar the distribtions of tags are over all resources.

To create a taxonomy of tags, they first sort the tags in non-increasing order of their closeness-centrality in the similarity graph. Closeness-centrality of a node n_i is defined as the inverse of the total distance that n_i is from all other nodes: $(\sum_{j=1}^{g} d(n_i,\, n_j))^{-1}$, where d is the geodesic distance between n_i and n_j.[2] They then start with an empty taxonomy,

[1] We assume here that there is a one-to-one correspondence between semantic concepts and tags, which is, as Mika points out, incorrect.

[2] Why a centrality measure for identifying general tags and not a more efficiently computed, local measure such as the degree of a node? A node far down in the taxonomy can have a large local connectivity (for example, a node pointing to a large number of leafs), but its centrality will be low because the distance to most of the nodes in the taxonomy will be high.

which contains only a root node not associated with a tag, and add each tag in turn starting from the most central. A tag is added as a child of either the tag it is most similar to if the similarity is above a threshold or the root node.

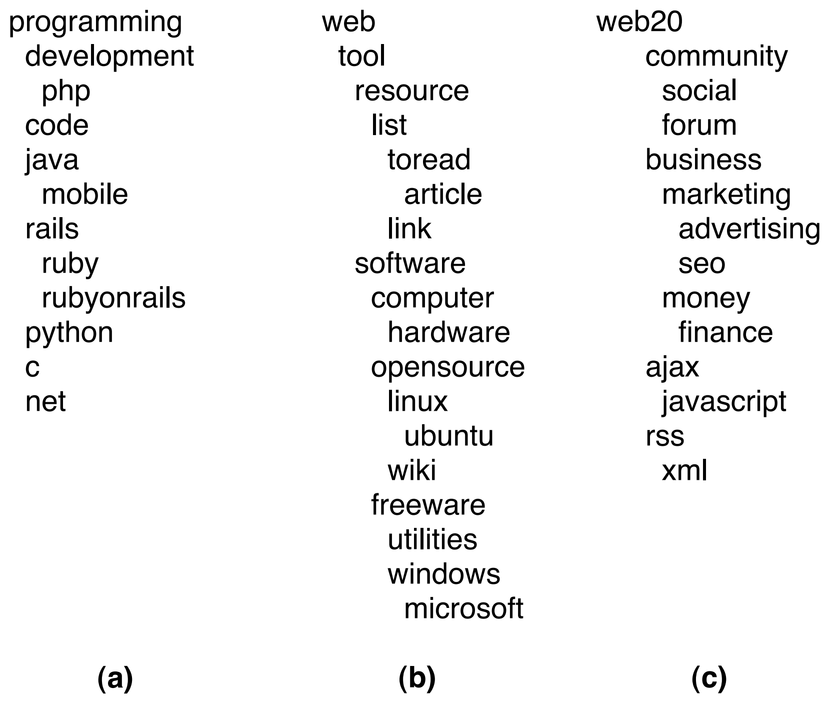

Fig. 2. Parts of a taxonomy created using the algorithm by Heymann et al. on the test data. Trees (b) and (c) suggest problems with contextual similarity: `ubuntu` is related to `linux`, but it is not obvious why both are subtags of `web`.

Heymann et al. base their approach on three assumptions: (1) the relationships in the taxonomy also exist in the similarity graph, (2) there are noisy connections between tags that have no matching connection in the taxonomy (hence, the edges in the similarity graph are a superset of the edges in the taxonomy), and (3) noisy connections are more common higher up in the hierarchy (that is, for more general tags).

Their algorithm further assumes that all tags are part of a taxonomy. This is a simplifying assumption because in general only a subset of tags is used for denoting the categories of resources [1]. Furthermore, the algorithm does not take the context of a parent tag into account when adding a child: The similarity between a tag and its potential parent in the taxonomy does not depend on the ancestors of the parent. This results in chains such as *design→web→howto→productivity→business* in which each link seems to make sense but the complete chain does not.

Applied to our test data, this context agnostic assignment results in a poorly structured taxonomy: Depending on how the similarity thresholds are chosen, the taxonomy is either too flat, with a large number of tags not having any children, or a single tag being the root of a deep tree containing most of the other tags, with a large number of tag chains not making sense (see figure 2).

A simple way to take the context into account is to require from a tag that is has a certain minimum average similarity to all predecessors of the parent. We add this test to the original algorithm after the tag p in the taxonomy that is most similar to the tag t to be inserted is determined. If the average similarity between t and the predecessors of p is below a given threshold, a copy p_c of p is added as a new top-level node to the taxonomy and t is made a child of p_c.

Applied to our test data, the extended algorithm leads to taxonomies without intuitively incorrect chains, but the overall structure is in general too flat (see figure 3).

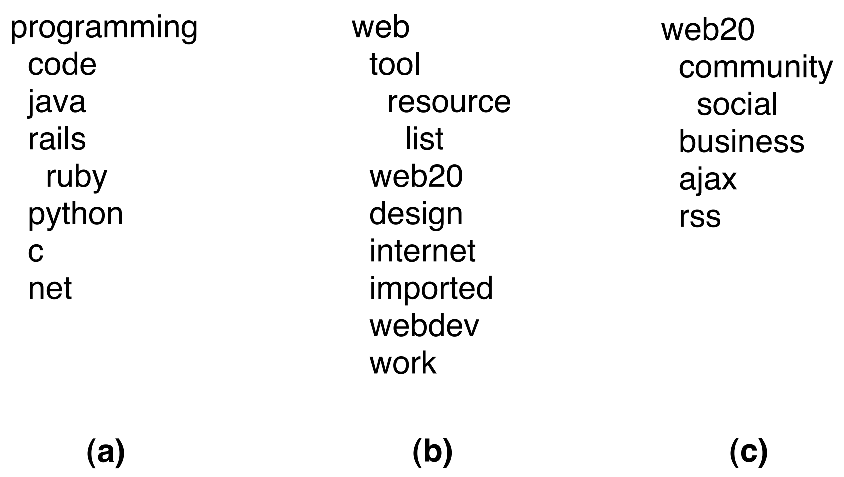

Fig. 3. Parts of a taxonomy created using Heymann's algorithm modified to take the context of tags into account.

Another approach is proposed by Schmitz et al.[4]. They mine from a tag space association rules of the form *If users assign the tags from X to some resource, they often also assign the tags from Y to them.* If resources tagged with t_0 are often also tagged with t_1 but a large number of resources tagged with t_1 are not tagged with t_0, t_1 can be considered to subsume t_0.

Formally, Schmitz et al. learn association rules over the set $T := \{\{i \mid \{a, c, i\} \in E\} \mid a \in A, c \in C\}$. Here, an association rule is a tuple in $2^I \setminus \emptyset \times 2^I \setminus \emptyset$. Whether an association rule (X, Y) is of interest or not can be determined by thresholds on its support $\text{supp}(X, Y) := |\{U \in T \mid X \subset U, Y \subset U\}| \,/\, |T|$ and its confidence $\text{conf}(X, Y) := |\{U \in T \mid X \subset U, Y \subset U\}| \,/\, |\{U \in T \mid X \subset U\}|$.

The cosine similarity measure used by Heymann et al. does not take into account the total count of occurences of a tag because tag vectors are being normalized. For instance, the vector $(1, 2, 3)$ is more similar to $(100, 180, 250)$ than $(100, 180, 250)$ is to $(50, 150, 200)$, although intuitively the latter pair should be more similar in respect to the corresponding tags' positions in the taxonomy.

In contrast, association rules reflect the frequencies of subsets. Assume we have got tags t_1, t_2, t_3 with the same distributions as used in the discussion of cosine similarity $v_{t_1} = (1, 2, 3)$, $v_{t_2} = (100, 180, 250)$, and $v_{t_3} = (50, 150, 200)$, and that the resources tagged with t_1 and t_3 are proper subsets of the resources tagged with t_2. Then $\text{conf}((t_2, t_1)) = 6 \,/\, 530$ and $\text{conf}((t_2, t_3)) = 400 \,/\, 530$. Hence, there is a stronger relationship between t_2 and t_3 than t_2 and t_1.

Schmitz et al. do not describe how a taxonomy can be created from the mined rules. One possible approach is the following: Given the set R of interesting association rules in $I \times I$ (that is, associations between single tags), we can define a graph $\text{AR} :=< Var, E_{ar}, w_{ar} >$, where $V_{ar} := \{i \in I \mid \exists j \in I : (i, j) \in R \,\text{or}\, (j, i) \in R\}$, $E_{ar} := \{(x, y) \mid (y, x) \in R\}$, $w_{ar}((x, y)) := \text{conf}((y, x))$. We then create the graph AR' by keeping for each node y only the incoming edge (x, y) with the strongest weight $w_{ar}((x, y))$. AR' is a forest, and a single tree (the taxonomy) can be created by introducing a new node r (the root of the taxonomy) and connecting it to all existing root nodes in the forest.

This algorithm, like the algorithm by Heymann et al., assumes that there is a one-to-one correspondence between tags and concepts, does not attempt to distinguish the different uses of tags, and ignores the context of a tag in the taxonomy.

Overall, we get better results on our test data with association rules than with Heymann's algorithm, but we observe similar issues in respect to the context of tags (see 4).

- blog
  - blogging
  - technology
    - tech
  - new
    - politic
  - business
    - startup
    - management
    - marketing
      - advertising
      - seo
  - wordpress
    - plugin
    - themes
  - culture
  - daily
  - rss
    - feed

(a)

- programming
  - tutorial
    - howto
      - hack
      - diy
      - tip
    - photoshop
  - ruby
    - rubyonrails
    - rails
  - php
    - mysql
  - net
  - api
  - framework
  - c
  - python

(b)

- web
  - browser
  - directory
  - internet
  - link
  - accessibility
  - web20
    - social
    - community
    - collaboration
  - website

(c)

Fig. 4. Parts of a taxonomy created using the assocition-rule algorithm.

We can extend the algorithm to take the context into account by requiring that there is an edge from a tag to all of its subtags in AR. For each root in AR$'$, we traverse the corresponding tree. If we reach a tag that is not adjacent to the root in AR, we make a copy of its parent in AR$'$ a new root and continue traversal. This corresponds to the modification we made to Heymann's algorithm.

2.4 A Tentative Recommendation

For our set of test data and implementations, the association-rule algorithm and its extension generated better taxonomies than Heymann's algorithm for a relatively wide range of parameters. The appendix lists structural features of taxonomies mined by the algorithms using several sets of parameter values. The 'better' structural features (see above) of the association-rule algorithms further support the impression we got from manual inspection of a sample of taxonomies.

- blog
  - blogging
  - technology
    - tech
  - news
    - politic
  - business
    - marketing
  - wordpress
  - culture
  - daily
  - rss

(a)

- programming
  - tutorial
  - ruby
    - rubyonrails
    - rails
  - php
  - net
  - api
  - framework
  - c
  - python
  - reference
    - database
  - language
  - java
  - code
  - c#
  - xml

(b)

- web
  - browser
  - directory
  - internet
  - link
  - accessibility
  - web20

(c)

Fig. 5. Parts of a taxonomy created using the assocition-rule algorithm modified to take the context of tags into account.

3 Mapping User Interests

Given a taxonomy of tags, we need to identify which of those tags best describe the interests of a user so we can apply the appropriate adaptation rules. We do this by computing the similarity between sets of tags: We represent a tag of the taxonomy by the set containing the tag and all of its subtags. Jaccard's similarity coefficient, defined as the ratio between the size of the intersection of two sets and the union of those sets, $|A \cap B| \, / \, |A \cup B|$, is used to determine how strongly the user, represented by the intersection of his tags with the set of all tags of the taxonomy, is associated with a specific tag of the taxonomy.

For example, one user in our dataset uses the following tags:

wiki code firefox cc blogger mysql hardware webstv wordpress own search mp3 linux css nba hacker zooomr bloglines java network service bittorrent bookmark vbscript lyrics perl blog teacher book crack teaching net irc homepage album asp assembly dictionary clubbox web20 tool javascript notepad yahoo wargames diagram xml game lifelype proxy regexp translation php ruby security foobar2000 decompiler p2p audio embedded forum database mobile eclipse html server bbs fju freebsd encryption movie sniffer ide maplebbs portalsite pda software

If we map this set to a taxonomy mined using the association-rule algorithm, we learn that the user is strongly associated with the concepts *programming*, *security*, and *software*. Note that an interest for *programming* is not explicit in the user's tags, but inferred using the mined taxonomy.

4 Things We Ignored

The presented approaches to mining taxonomies from tag spaces ignore a number of issues relevant to our application domain:

dynamics: How tags are used changes in time. For example, a tag specialization might be introduced that describes a subset of the tagged resources better and thus replaces a more broader tag, or a tag for an entirely new and popular concept might be introduced. This dynamics will affect adversly the quality of the mapping of user interests to the then out-of-date tag taxonomy. An ideal system would adapt the taxonomy to any changes so manual maintenance of the taxonomy or the adaptation rules would not be necessary.

not a 1-to-1 mapping between tags and concepts: As Mika points out, a concept might be represented not by a single but by a set of several tags. This suggests that the quality of the learned structure of the tag space can be improved if the simplifying assumption of a 1-to-1 mapping is dropped.

different uses of tags: Tags can be used in different functions. In addition to denoting categories and subcategories, they can describe the content, type, and owner of a resource, the opinion of the tagger, what to do with a resource ("toread"), or the relation to the tagger ("mycomments") (see Golder et al.[1]). Distinguishing between the different functions should improve the learned structure of the tag space.

polysemy, synonymy: All presented approaches ignore polysemy and synonymy in the tag space. This leads to a reduction in quality of the learned structure because the quantitative relationships between concepts are misrepresented.

5 Conclusion

We have begun exploring an application scenario in which data from a tag space is used to adapt a system to an individual user. Algorithms and approaches in this domain are still in their infancy, and with lots of relevant data available on the web and its potential usefulness (not only in the mentioned e-commerce scenario), we see it as a promising area of research.

References

1. Scott A. Golder and Bernardo A. Huberman. Usage patterns of collaborative tagging systems. *J. Inf. Sci.*, 32(2):198–208, April 2006.
2. Paul Heymann and Hector Garcia-Molina. Collaborative creation of communal hierarchical taxonomies in social tagging systems. Technical Report 2006-10, Stanford University, April 2006.
3. Peter Mika. Ontologies are us: A unified model of social networks and semantics. In *International Semantic Web Conference*, volume 3729 of *Lecture Notes in Computer Science*, pages 522–536. International Semantic Web Conference 2005, Springer, November 2005.
4. Christoph Schmitz, Andreas Hotho, Robert Jäschke, and Gerd Stumme. Mining association rules in folksonomies. In V. Batagelj, H.-H. Bock, A. Ferligoj, and A. iberna, editors, *Data Science and Classification*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 261–270, Berlin, Heidelberg, 2006. Springer.

A Structural Features of Mined Taxonomies

edge sim	parent sim	#taxa	max depth	#lvl1	#lvl2	#lvl3	#lvl4	#lvl5
0.05	0.02	489	13	2	16	41	76	98
0.05	0.04	489	13	2	16	41	76	98
0.05	0.06	489	13	16	22	44	76	95
0.05	0.08	489	11	40	25	44	75	91
0.05	0.10	489	11	85	33	40	71	83
0.05	0.12	489	10	134	59	51	72	62
0.05	0.14	489	10	178	67	51	59	48
0.05	0.16	489	9	209	72	62	57	35
0.05	0.18	489	8	238	79	61	46	29
0.05	0.20	489	8	265	90	57	39	18
0.05	0.22	489	8	293	89	50	31	15
0.05	0.24	489	5	324	88	48	20	9
0.05	0.26	489	5	349	75	39	19	7
0.05	0.28	489	5	367	69	33	16	4
0.05	0.30	489	5	386	67	29	6	1
0.05	0.32	489	4	402	60	23	4	0
0.05	0.34	489	4	414	55	18	2	0
0.05	0.36	489	3	424	48	17	0	0
0.05	0.38	489	3	430	46	13	0	0
0.05	0.40	489	3	441	38	10	0	0
0.1	0.02	427	15	8	11	10	11	23
0.1	0.04	427	13	13	21	29	30	47
0.1	0.06	427	13	13	21	29	30	47
0.1	0.08	427	13	13	21	29	30	47
0.1	0.10	427	13	13	21	29	30	47
0.1	0.12	427	11	65	43	42	41	53
0.1	0.14	427	11	115	62	38	32	44
0.1	0.16	427	11	147	63	50	28	29
0.1	0.18	427	11	175	65	47	24	27
0.1	0.20	427	11	201	82	45	19	20
0.1	0.22	427	8	232	86	46	17	16
0.1	0.24	427	8	261	88	38	15	10
0.1	0.26	427	8	285	77	31	11	9
0.1	0.28	427	8	306	71	24	8	8
0.1	0.30	427	7	326	70	20	5	3
0.1	0.32	427	7	339	62	17	4	2
0.1	0.34	427	6	351	55	14	4	2
0.1	0.36	427	5	362	47	15	2	1
0.1	0.38	427	5	368	44	12	2	1
0.1	0.40	427	4	378	38	10	1	0

Table 1. Results of Heymann's algorithm applied to the test data. *edge sim* is the minimum similarity required for an edge to be created between two tags in the similarity graph. *parent sim* is the minimum similarity required for a tag to become the child of a taxon.

edge sim	parent sim	context sim	#taxa	max depth	#lvl1	#lvl2	#lvl3	#lvl4	#lvl5
0.05	0.02	0.02	601	10	114	186	73	63	63
0.05	0.02	0.04	655	8	168	301	71	49	33
0.05	0.02	0.06	685	7	198	377	61	34	10
0.05	0.02	0.08	700	6	213	412	47	21	6
0.05	0.02	0.10	709	4	222	439	38	10	0
0.05	0.02	0.12	718	4	231	457	25	5	0
0.05	0.02	0.14	719	4	232	469	15	3	0
0.05	0.02	0.16	720	3	233	476	11	0	0
0.05	0.02	0.18	721	3	234	481	6	0	0
0.05	0.02	0.20	722	3	235	483	4	0	0
0.05	0.04	0.02	601	10	114	186	73	63	63
0.05	0.04	0.04	655	8	168	301	71	49	33
0.05	0.04	0.06	685	7	198	377	61	34	10
0.05	0.04	0.08	700	6	213	412	47	21	6
0.05	0.04	0.10	709	4	222	439	38	10	0
0.05	0.04	0.12	718	4	231	457	25	5	0
0.05	0.04	0.14	719	4	232	469	15	3	0
0.05	0.04	0.16	720	3	233	476	11	0	0
0.05	0.04	0.18	721	3	234	481	6	0	0
0.05	0.04	0.20	722	3	235	483	4	0	0
0.05	0.08	0.02	577	10	128	154	70	62	62
0.05	0.08	0.04	629	8	180	265	70	49	32
0.05	0.08	0.06	661	7	212	339	61	34	10
0.05	0.08	0.08	676	6	227	374	47	21	6
0.05	0.08	0.10	685	4	236	401	38	10	0
0.05	0.08	0.12	694	4	245	419	25	5	0
0.05	0.08	0.14	696	4	247	431	15	3	0
0.05	0.08	0.16	698	3	249	438	11	0	0
0.05	0.08	0.18	699	3	250	443	6	0	0
0.05	0.08	0.20	700	3	251	445	4	0	0
0.05	0.16	0.02	513	9	233	101	59	48	32
0.05	0.16	0.04	540	8	260	139	58	42	17
0.05	0.16	0.06	558	6	278	178	57	35	6
0.05	0.16	0.08	573	6	293	204	48	20	5
0.05	0.16	0.10	587	5	307	229	36	13	2
0.05	0.16	0.12	597	4	317	249	24	7	0
0.05	0.16	0.14	599	4	319	257	17	6	0
0.05	0.16	0.16	605	3	325	269	11	0	0
0.05	0.16	0.18	607	3	327	274	6	0	0
0.05	0.16	0.20	608	3	328	276	4	0	0

Table 2. Results of the extension of Heymann's algorithm applied to the test data. *edge sim* is the minimum similarity required for an edge to be created between two tags in the similarity graph. *parent sim* is the minimum similarity required for a tag to become the child of a taxon.

edge sim	parent sim	context sim	#taxa	max depth	#lvl1	#lvl2	#lvl3	#lvl4	#lvl5
0.1	0.02	0.02	506	13	87	129	61	26	34
0.1	0.02	0.04	554	9	135	217	61	27	28
0.1	0.02	0.06	583	9	164	284	63	21	13
0.1	0.02	0.08	603	8	184	343	41	15	5
0.1	0.02	0.10	613	8	194	370	26	10	5
0.1	0.02	0.12	618	8	199	390	19	3	2
0.1	0.02	0.14	622	6	203	400	12	5	1
0.1	0.02	0.16	625	5	206	409	6	3	1
0.1	0.02	0.18	625	5	206	411	5	2	1
0.1	0.02	0.20	625	5	206	412	4	2	1
0.1	0.04	0.02	500	13	86	123	61	26	34
0.1	0.04	0.04	548	9	134	211	60	27	28
0.1	0.04	0.06	577	9	163	279	63	21	13
0.1	0.04	0.08	597	8	183	339	42	13	5
0.1	0.04	0.10	607	8	193	366	27	8	5
0.1	0.04	0.12	612	8	198	385	19	3	2
0.1	0.04	0.14	616	6	202	395	12	5	1
0.1	0.04	0.16	619	5	205	404	6	3	1
0.1	0.04	0.18	619	5	205	406	5	2	1
0.1	0.04	0.20	619	5	205	407	4	2	1
0.1	0.08	0.02	500	13	86	123	61	26	34
0.1	0.08	0.04	548	9	134	211	60	27	28
0.1	0.08	0.06	577	9	163	279	63	21	13
0.1	0.08	0.08	597	8	183	339	42	13	5
0.1	0.08	0.10	607	8	193	366	27	8	5
0.1	0.08	0.12	612	8	198	385	19	3	2
0.1	0.08	0.14	616	6	202	395	12	5	1
0.1	0.08	0.16	619	5	205	404	6	3	1
0.1	0.08	0.18	619	5	205	406	5	2	1
0.1	0.08	0.20	619	5	205	407	4	2	1
0.1	0.16	0.02	453	11	173	89	61	29	25
0.1	0.16	0.04	474	9	194	122	62	26	20
0.1	0.16	0.06	496	8	216	159	61	20	10
0.1	0.16	0.08	512	8	232	205	46	10	5
0.1	0.16	0.10	526	8	246	233	27	8	4
0.1	0.16	0.12	533	8	253	251	19	3	2
0.1	0.16	0.14	538	6	258	261	12	5	1
0.1	0.16	0.16	543	5	263	270	6	3	1
0.1	0.16	0.18	544	5	264	272	5	2	1
0.1	0.16	0.20	544	5	264	273	4	2	1

Table 3. Results of the extension of Heymann's algorithm applied to the test data. *edge sim* is the minimum similarity required for an edge to be created between two tags in the similarity graph. *parent sim* is the minimum similarity required for a tag to become the child of a taxon.

confidence	support	#taxa	max depth	#lvl1	#lvl2	#lvl3	#lvl4	#lvl5
0.05	0.0001	1071	8	57	129	283	285	182
0.05	0.0002	502	8	19	59	172	132	86
0.05	0.0003	344	8	13	44	125	97	47
0.05	0.0004	258	6	11	42	96	70	30
0.05	0.0005	211	6	12	37	77	58	21
0.05	0.0006	162	6	7	29	63	42	16
0.05	0.0007	133	6	7	27	50	32	12
0.05	0.0008	114	6	5	23	44	26	11
0.05	0.0009	106	6	7	26	41	20	9
0.05	0.0010	93	6	9	26	31	17	8
0.10	0.0001	1067	7	80	323	331	199	105
0.10	0.0002	499	7	33	160	173	86	40
0.10	0.0003	339	7	24	112	120	58	22
0.10	0.0004	255	6	21	89	89	41	14
0.10	0.0005	209	5	20	75	72	32	10
0.10	0.0006	157	5	12	58	53	26	8
0.10	0.0007	130	5	12	52	40	18	8
0.10	0.0008	113	5	10	48	31	17	7
0.10	0.0009	105	5	10	47	27	15	6
0.10	0.0010	92	5	12	41	22	13	4
0.15	0.0001	1037	6	115	482	304	98	34
0.15	0.0002	482	5	54	245	134	34	15
0.15	0.0003	327	5	36	170	94	18	9
0.15	0.0004	246	5	31	129	69	13	4
0.15	0.0005	201	5	29	103	57	10	2
0.15	0.0006	154	5	20	81	42	9	2
0.15	0.0007	127	5	18	67	32	8	2
0.15	0.0008	108	5	14	57	27	8	2
0.15	0.0009	101	5	14	56	22	8	1
0.15	0.0010	89	5	16	50	17	5	1
0.20	0.0001	988	6	149	576	196	56	10
0.20	0.0002	457	5	75	274	85	20	3
0.20	0.0003	309	5	52	186	58	10	3
0.20	0.0004	227	4	41	134	45	7	0
0.20	0.0005	185	4	38	108	33	6	0
0.20	0.0006	140	4	28	82	25	5	0
0.20	0.0007	114	4	23	66	21	4	0
0.20	0.0008	97	4	19	56	18	4	0
0.20	0.0009	88	4	16	51	17	4	0
0.20	0.0010	79	4	18	46	13	2	0
0.25	0.0001	903	4	170	564	131	34	4
0.25	0.0002	403	4	83	256	52	12	0
0.25	0.0003	267	4	57	168	37	5	0
0.25	0.0004	196	4	44	121	27	4	0
0.25	0.0005	159	4	40	97	19	3	0
0.25	0.0006	120	4	30	74	13	3	0
0.25	0.0007	94	4	24	58	10	2	0
0.25	0.0008	78	4	19	48	9	2	0
0.25	0.0009	71	4	17	44	8	2	0
0.25	0.0010	65	4	17	38	8	2	0

Table 4. Results of the association-rule algorithm applied to the test data.

confidence	support	#taxa	max depth	#lvl1	#lvl2	#lvl3	#lvl4	#lvl5
0.05	0.0001	1355	6	228	756	259	81	30
0.05	0.0002	646	5	107	362	134	34	9
0.05	0.0003	435	5	73	247	89	21	5
0.05	0.0004	331	5	62	199	54	12	4
0.05	0.0005	269	5	51	162	42	10	4
0.05	0.0006	211	5	41	127	31	9	3
0.05	0.0007	170	5	35	101	24	7	3
0.05	0.0008	145	5	30	88	18	7	2
0.05	0.0009	135	5	28	83	17	5	2
0.05	0.0010	117	5	26	70	15	4	2
0.10	0.0001	1258	5	233	794	193	34	4
0.10	0.0002	594	5	111	382	88	12	1
0.10	0.0003	399	5	74	254	63	7	1
0.10	0.0004	305	5	62	201	36	5	1
0.10	0.0005	246	5	51	162	28	4	1
0.10	0.0006	189	5	40	125	19	4	1
0.10	0.0007	154	5	35	100	14	4	1
0.10	0.0008	135	5	32	88	11	3	1
0.10	0.0009	123	5	28	81	11	2	1
0.10	0.0010	106	5	26	68	9	2	1
0.15	0.0001	1181	4	242	802	121	16	0
0.15	0.0002	548	4	113	379	52	4	0
0.15	0.0003	372	4	76	256	37	3	0
0.15	0.0004	280	4	62	192	24	2	0
0.15	0.0005	225	4	51	151	21	2	0
0.15	0.0006	175	4	39	119	15	2	0
0.15	0.0007	146	4	36	97	11	2	0
0.15	0.0008	126	4	32	84	8	2	0
0.15	0.0009	116	4	29	78	8	1	0
0.15	0.0010	100	4	27	65	7	1	0
0.20	0.0001	1087	4	240	756	79	12	0
0.20	0.0002	496	4	112	353	28	3	0
0.20	0.0003	334	4	75	238	18	3	0
0.20	0.0004	248	4	61	174	11	2	0
0.20	0.0005	199	4	51	136	10	2	0
0.20	0.0006	151	4	38	103	8	2	0
0.20	0.0007	124	4	33	82	7	2	0
0.20	0.0008	107	4	29	71	5	2	0
0.20	0.0009	99	4	27	66	5	1	0
0.20	0.0010	86	4	25	55	5	1	0
0.25	0.0001	969	4	229	668	61	11	0
0.25	0.0002	424	4	103	296	22	3	0
0.25	0.0003	281	4	70	193	15	3	0
0.25	0.0004	210	4	57	141	10	2	0
0.25	0.0005	167	4	47	109	9	2	0
0.25	0.0006	125	4	34	81	8	2	0
0.25	0.0007	97	4	27	61	7	2	0
0.25	0.0008	81	4	22	52	5	2	0
0.25	0.0009	75	4	21	48	5	1	0
0.25	0.0010	69	4	21	42	5	1	0

Table 5. Results of the extended association-rule algorithm applied to the test data.