Neighborhood-Based Tag Prediction*

Adriana Budura[1], Sebastian Michel[1], Philippe Cudré-Mauroux[2], and Karl Aberer[1]

[1] Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland
adriana.budura@epfl.ch, sebastian.michel@epfl.ch, karl.aberer@epfl.ch
[2] MIT, USA
pcm@csail.mit.edu

Abstract. We consider the problem of tag prediction in collaborative tagging systems where users share and annotate resources on the Web. We put forward HAMLET, a novel approach to automatically propagate tags along the edges of a graph which relates similar documents. We identify the core principles underlying tag propagation for which we derive suitable scoring models combined in one overall ranking formula. Leveraging these scores, we present an efficient top-k tag selection algorithm that infers additional tags by carefully inspecting neighbors in the document graph. Experiments using real-world data demonstrate the viability of our approach in large-scale environments where tags are scarce.

1 Introduction

The past few years have witnessed the rise of Web 2.0 applications promoting the sharing of documents through online communities. Documents shared by users in such applications vary largely, from scientific publications on *CiteULike* (*http://www.citeulike.org*) to images on *Flickr* (*http://www.flickr.com*) or bookmarks on *del.icio.us* (*http://del.icio.us*). One standard practice in all these scenarios is to rely on user-provided metadata to foster search capabilities. *Tags* are short textual annotations used to describe documents and represent such user-generated metadata in Web 2.0 applications. Tags are essential in resolving user queries targeting shared documents, yet require human attention to be generated. Users typically tag a small fraction of the shared documents only, leaving most of the other documents with incomplete metadata. This lack of metadata seriously impairs search, as documents without proper annotations are typically much harder to retrieve than correctly annotated documents.

Several attempts have already been made to foster the creation of additional tags in such a context. A number of successful projects, such as the E.S.P. game (*http://www.espgame.org*), are based on incentives compelling the end-users to create additional tags. Automatic tag creation techniques can produce good

* The work presented in this paper was partially supported by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

results for specific tag categories; they fail however in general, as automating the labeling of arbitrary documents is an inherently challenging research problem.

In the following, we present a tag inference technique to assign tags to previously un-tagged documents or to extend the set of tags for already tagged documents by *propagating* existing tags from one document to related documents. We put forward HAMLET (*Harvesting Adjacent Metadata in Large-ScalE Tagging Systems)*, a suite of principles, scoring models and algorithms for metadata propagation. HAMLET is based on the assumption that similar documents share similar tags, therefore we take advantage of a graph relating similar documents, and suppose that a tag attached to one particular document in the graph might be appropriate for the neighboring documents as well. Our approach is triggered by the observation that in our considered scenarios links very often stay in the topic of the source document. As this has been shown for Web pages, it is even more visible in citation of scientific papers (where authors refer to related work).

We apply our principles on real data extracted from two popular Web portals: CiteULike and del.icio.us. CiteULike is an application where users can create online libraries of academic papers, tag papers, and search for scientific articles while del.icio.us provides similar functionalities for organizing bookmarks.

1.1 Computational Model

Our model is based on three concepts: *documents*, *tags*, and *document neighborhoods*. A document $d \in \mathcal{D}$ is uniquely identifiable and represents in our scenarios scientific publications or Web pages. The documents are created and shared by users who interact with the tagging system. A *tag* $t \in \mathcal{T}$ represents a user-provided, unstructured piece of metadata attached to a document. We write $t \mapsto d$ to indicate that a particular tag t is attached to a document d and $T(d)$ to denote the bag of tags $\{t \mid t \mapsto d\}$ attached to a given document d.

The document graph is generated by the authors of the documents by references (citations) to other publications in the case of scientific publications or by linking via html links in the case of Web pages. To propagate tags from one document to the others, we take advantage of document graphs organizing related documents together. We write $d_i \to d_j$ to indicate that the author of a document has decided to relate document d_i to document d_j through a link in the graph. The *neighborhood* of a document d is denoted $N(d)$ and contains all documents which are reachable from d by traversing the edges of the graph. We will devise a top-k algorithm to dynamically explore the neighborhood of a particular document in order to infer new tags which are relevant for the considered document.

1.2 Contributions and Outline

Our contributions can be summarized as follows:

- We identify document graphs explicitly created by end-users in two real world scenarios (citation graph for scientific publications and Web graph) as an instrument for tag inference;
- We present a scoring model to assess the relevance of tags propagated along edges of the document graph;

- We show how to cast our tag propagation problem into a top-k selection problem and adapt state-of-the-art top-k algorithms to our context;
- We experimentally evaluate the performance of our approach by analyzing the propagation of tags for documents from two real world tagging portals.

We start below by reviewing related work in Section 2. We highlight the peculiarities of our setting, present the principles underpinning tag propagation and introduce our new scoring model in Section 3. In Section 4 we describe how to cast our tag propagation problem into a top-k aggregation problem, and present a new algorithm tailored to our needs. We discuss experimental results in Section 5 before concluding in Section 6.

2 Related Work

Recently the sphere of social annotations has gained increasing attention. The term *folksonomy* has been widely used to denote the process of collaboratively generating unstructured annotations for documents. Efforts in this area have concentrated on formalizing generic collaborative tagging systems. The work by Mika [17] considers a tri-partite model of instances, users, and concepts and mines the relationships among these three entities. [8,16,7] have concentrated on understanding the tagging process and the resulting social annotations by examining tag distributions, constructing tag-tag correlation networks and extracting statistics of users' tagging behavior. [14] investigate the overlap between anchor-text and tags for a Web page. [9] present an in-depth analysis of the del.icio.us tagging portal and conclude that the data contained in tagging portals represents valuable information which could enhance the performance of search engines. A similar conclusion is drawn by Yanbe et al. [24] who also investigate the possibility of a hybrid search approach, based both on PageRank and on social annotations. Bao et al. [2] introduce two algorithms which integrate information extracted from the tags into the search process. In [19] Schenkel et al. propose a new method to leverage social tags for improved content search. While their method also incorporates users in the query processing, it does not consider tag propagation using document-to-document similarities. Their method is orthogonal to our method as we focus on tag retrieval which is an important ingredient for meaningful document retrieval in social networks.

A different body of work deals with the problem of extracting semantic information from unstructured annotations. A probabilistic model has been proposed in [23] in order to derive the emergent semantics which characterize the collaborative tagging process. The authors also describe a framework for discovering semantically related tags based on the relations between the three entities: tags, users and resources. Rattenbury et al. [18] extract events and place semantics from Flickr tags based on geographical and temporal metadata attached to images. Schmitz et al. [20] address a similar problem and propose a solution based on a subsumption model for deriving an ontology from social annotations. While the above mentioned approaches are similar to our work, they are actually orthogonal since the authors do not directly address the problem of inferring tags for resources.

Addressing the problem of inferring new annotations for resources, the approach in [3] generates personal annotations based on the content of the documents or on data from the personal desktop. Our method enhances the annotation set of documents by solely relying on their context and uses content-based information only to optimize the results.

Sigurbjörnsson et al. [21] address a similar problem of automatic tag suggestion in the Flickr portal by using a co-occurrence measure for assessing tag relatedness. As opposed to our approach, the authors do not consider the local context of a resource and rely exclusively on global statistics.

The work by Jäschke et al. [13] considers a tripartite model of users, resources, and tags and employ their Folkrank [11] algorithm, a PageRank like, iterative algorithm, to assign tags to resources. While the general idea is similar to ours, it requires knowledge of the global graph.

The same problem is tackled by Heymann et al. [10] where the authors predict tags for Web pages in del.icio.us based on the anchortext, Web page content and on the surrounding hosts. The authors predict tags from the set of 100 most frequent tags found in del.icio.us by training a classifier which runs a binary classification task for each tag. The classifier is trained on a set of very popular bookmarks. As opposed to this method our approach does not rely on classification. In addition, we do not restrict ourselves to a predefined set of tags which are predicted. By doing so we could expect a gain in precision; however, this would drastically restrict our general approach to only a small number of tags which can be inferred.

Tag prediction is treated as a classification problem by Song et al. [22] who propose a real-time automatic tag recommendation method. The authors start by building document clusters offline and then assign a new document to those clusters with some probability in an online process. Again, our method is different since we do not consider any learning algorithms.

Our idea of using the context of a document (defined by its neighborhood in the graph) is furthermore supported by the observations in, e.g., [4], showing that in a Web graph most outgoing hyperlinks stay in the same topic as the originating page.

Existing work in the area of graph based classification [15,1] is also potentially relevant to our approach. [1] is in particular related to our approach since it considers content based node similarities as edge weights in the citation graph which supports our initiative of using links to relate documents. However, these classification algorithms are inherently limited to only a few classes, usually rely on global computations, or require several iterations, which makes these approaches hardly applicable in our envisioned scenario.

3 Scoring Model

In the following we introduce the main characteristics of our setting, discuss generic principles underpinning our tag propagation framework and show how these principles can be applied to create a scoring model for assessing tag relevance w.r.t. a document.

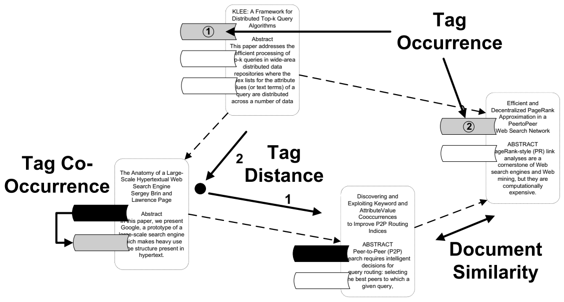
Tag Occurrence
Tag Co-Occurrence
Tag Distance
Document Similarity
1
2
1
2
KLEE: A Framework for Distributed Top-k Query Algorithms
Efficient and Decentralized PageRank Approximation in a PeertoPeer Web Search Network
The Anatomy of a Large-Scale Hypertextual Web Search Engine
Discovering and Exploiting Keyword and AttributeValue Cooccurrences to Improve P2P Routing Indices

Fig. 1. The four principles underpinning our notion of tag propagation

Our setting has the following characteristics:

Scarce Information: tagging documents requires human attention, which is one of the scarcest resources today. Realistically, only a small subset of users will make the effort of tagging a given shared document and most probably the number of tags attached will be small too.

Document Organization: the document collections we analyze are not flat, as documents are organized through user-defined links which are in our case html links or citations. Hence, in addition to the notion of document collection classically defined in information retrieval, we always take into consideration document neighborhoods defined by following series of links from one given document.

3.1 Principles

Our tag propagation algorithm relies on four key principles (cf. Figure 1) which help us determine whether a tag should be propagated towards a given document or not, i.e., to which extent the tag is relevant to the given document.

i. the frequency with which a tag appears in a given document neighborhood (*tag occurrence*);
ii. the frequency with which a pair of tags is simultaneously attached to a document (*tag co-occurrence*);
iii. the distance between a potential tag and a given document in the resource graph (*tag distance*);
iv. the similarity between the documents the tags are attached to (*document similarity*).

In the following we will present the different measures which we use to capture these principles and which will be later combined into an overall scoring function.

3.2 Measures of Tag Relevance

Tag Occurrence. The importance of a term in a document is usually considered as increasing proportionally with the term frequency, i.e., with the number of times the term appears in the document. In our setting where most tags are missing, however, tags rarely occur more than once for a given document, proving the measure pointless most of the time. Instead, we define the notion of tag occurrence as the number of times a tag t' appears in a certain neighborhood of a document d, $N(d)$: $occ_N(t') = \sum_{d' \in N(d)} \mathbb{1}_{d'}(t')$, where $\mathbb{1}_{d'}(t')$ is an indicator function equal to one if $t' \mapsto d'$, to zero otherwise. Hence, the tag occurrence can be seen as capturing the popularity of a tag in a neighborhood. Note that we consider only a boolean decision, i.e., a tag is attached to a document or not. We do not consider how often a tag is assigned to a document as we aim at searching for support of tags in the entire neighborhood and do not want to push the influence of tags that occur often in some documents but rarely in the neighborhood.

Tag Co-Occurrence. The tags attached to a document are usually not independent. Some tags regularly appear together as they describe correlated facets of a given document (e.g., *Macintosh* and *Apple*). Therefore we consider measures for tag co-occurrence like the Dice or Jaccard coefficients or conditional probabilities as one ingredient of assessing tag relatedness. Using such measures we can assess the potential relevance $co\_occ(t', t)$ of a tag t' for a document that already contains a tag t. As documents typically contain more than one tag, we extend this notion for sets of tags $T(d)$ attached to a document d: $co\_occ(t', T(d)) = \sum_{t \in T(d)} co\_occ(t', t)$.

As one particular instantiation, we consider in our work the conditional probability that a document contains tag t' given that it already contains tag t: $P(t' \mapsto d \mid t \mapsto d)$. We treat all initial tags individually, i.e., we do not consider an aggregated score for an observed tag $t' \in N(d)$ to all possible subsets of the initial set of tags attached to d. In the case when the set of initial tags is empty we synthetically introduce a tag t_s for the considered document that co-occurs once with t' in the whole collection.

Document Similarity. Another ingredient of our scoring model is the degree of similarity between the documents to which the tags are attached. We employ a standard vector space model and use the following cosine similarity metric:

$$sim(d, d') = \frac{\sum_i w_d[i] * w_{d'}[i]}{\sqrt{\sum_i w_d[i]^2} * \sqrt{\sum_i w_{d'}[i]^2}},$$

where w_d is the vector representation of document d and $w_d[i]$ its i-th entry, i.e. reflecting the importance of term $term_i$ for d. There are multiple ways to quantify this importance, such as the relative term frequency $rtf = tf/tf_{max}$ which we have chosen in this work, where tf is the term frequency which is defined as the number of occurrences for a particular term in a particular document. tf_{max} denotes the largest tf value for one document. In order to assess the relatedness of documents which are *several* hops away we consider the product of similarities along the path between the observed document and the initial document:

$$\Pi_{i=0}^{|path(d,d')|-1} sim(d_i, d_{i+1})$$

Tag Distance. We assume that documents which are closer to each other in the graph have more related tags. Thus, an important metric is the distance from from one document d to another document d'. We define this distance as the shortest path (in terms of the number of graph edges): $dist(d, d') = path(d, d')$. We will use this distance as a dampening factor reducing the score of those tags that are located further away in the graph.

Due to the top-k fashion of our algorithm that explores the neighbors in a greedy way, we cannot (as we will see in Section 4) use the exact shortest path length. Instead, we compute the path with the smallest "cost" and use its length to approximate the minimal distance between the documents. The "cost" of a path is defined in this case in terms of 1 minus the similarity score of the path as described above.

3.3 Combined Scoring Function

We combine the four principles described above and produce a scoring function, used to assess the degree of relevance of a tag t' (observed in the neighborhood of d) for a document d.

$$score(t', d) = \sum_{d' \in N(d) \wedge t' \mapsto d'}^{m} S * O$$

The final score of tag t' observed up to m times in the neighborhood of the initial document d represents the sum of partial scores for each of its single occurrences. The parameter m controls the number of tag occurrences to be considered in the final aggregation. As we will see in the following section, m is required to compute the upper and lower bound scores to enable tag pruning in the top-k retrieval and therefore influences the size of the neighborhood which will be explored.

The partial score for each observation of t' is computed as a product of two factors. The first factor, coined S, consists of the aggregated similarities along the path between a document d' and the initial document d, i.e., principle (iv), dampened by the distance between the documents. The rationale behind this dampening stems from the fact that the relevance of a tag decreases with its distance to the initial document, which is one of our fundamental assumptions, i.e., principle (iii).

$$S = \frac{\Pi_{i=0}^{|path(d,d')|-1} sim(d_i, d_{i+1})}{log(1 + dist(d, d'))}$$

The second factor, denoted O, expresses the relatedness of a particular tag t' to the set of tags obtained from the initial document $T(d)$, which reflects the second principle (ii).

$$O = log(1 + co\_occ(t', T(d)))$$

In the overall scoring function, the sum aggregates the various occurrences of tag t' in the neighborhood of d, $N(d)$, following principle (i). Overall, the term (i.e., tag) specific influence is weighted by the objective importance of the term. This importance is in our case modeled by the distance from the point of observation to the initial document.

4 Top-k Tag Selection

Our algorithm starts from an initial document for which new tags should be inferred and traverses its neighborhood while computing scores for each observed tag. In the following, we show how tag propagation can be cast into a top-k selection problem that finds the most relevant tags for a given document in the graph while minimizing the number of documents visited.

As of today, the most efficient top-k query processing algorithms are based on the family of threshold algorithms (TA) (cf., e.g., [5]). In particular, the NRA (or TA-sorted) algorithm [6] only uses sequential (sorted) data access and hence is particularly interesting for our graph-based top-k problem. Our top-k tag selection algorithm is based on the concepts developed in this context, but adapts this family of algorithms to fit our unique context.

A common feature of these threshold algorithms is that they consider monotone aggregation functions, on which we focus here as well. We employ summation as the aggregation function, as it reflects the paradigm of considering the tag occurrences in a neighborhood.

Our computational model is the following: We consider the tag propagation along multiple edges in the graph towards an initial document d as a top-k aggregation problem where every document $d' \in N(d)$ represents one particular index list consisting of a series of $(tag, score)$ pairs. These $(tag, score)$ pairs are sorted in descending order w.r.t. to their score and express the relevance of a particular tag to the initial document d. Each edge in the document graph leads from one index list to other index lists (the actual number of index lists it leads to depends on the out-degree). The algorithm starts at the document for which we want to infer new tags and traverses the graph while computing scores for each observed tags t'.

Algorithm 1 depicts our nested top-k graph traversal and top-k aggregation algorithm. The algorithm performs a greedy traversal of the graph as follows: In the beginning, the algorithm inserts all neighbors of the initial document into a priority queue (Q_{docs}). Then, it chooses the neighbor with the highest document similarity (line 8) and aggregates the $(tag, score)$ pairs of the selected document incorporating the document similarity as a weight as mentioned above. Subsequently, the neighbors of the current document are inserted in the priority queue. Index lists are thus accessed on-demand depending on their expected contribution to the final result.

The algorithm maintains a list of potential candidate tags (Q_{cand}) and a list of the current top-k results ($topk$). Following the standard concepts of top-k query processing [6], we define $worstscore(t')$ (line 11) as the lower bound score for tag t' by considering those scores that have already been reported for t'. Similarly, $bestscore(t')$ denotes the upper bound score for tag t', that is the

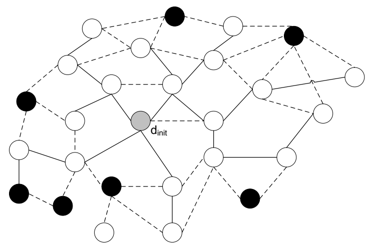

Fig. 2. An example of a particular state during the top-k aggregation. The thick lines indicate the edges of the spanning tree. All nodes touching these edges have been visited so far. The black circles denote the current border line (candidate nodes to be visited in the next step) which is responsible for the score upper bounds of the tags.

```
getTopKTags(Document $d_{init}$)
1  PriorityQueue<document, score> $Q_{docs} = \emptyset$
2  PriorityQueue<tag, score> $Q_{cand} = \emptyset$
3  Array($T, float$) $topk = \emptyset$
4  $Q_{docs}.enqueue$(firstlevelneighbors($d_{init}$))
5  $s_{max} = 1$
6  $min\text{-}k = 0$
7  while ($Q_{cand}.size > 0$ and $m * s_{max} > min\text{-}k$){
8     $d_{current} = Q_{docs}.dequeue()$
9     $top_{border} = Q_{docs}.peek().score$
10    for $t' \in d_{current}${
11       $t'.worstscore+ = score(t, d_{current})$
12       $t'.seen + +$
13       if $t'.worstscore > min\text{-}k$ then {
14          $topk.update(t')$
15          $min\text{-}k = topk[k]$
16       }
17       $t'.bestscore = ...$ (see below) ...
18       if $t'.bestscore \leq min\text{-}k$ then $Q_{cand}.delete(t')$
19    }
20    $Q_{docs}.enqueue$(firstlevelneighbors($d_{current}$))
21    $Q_{cand}.updateBestScore(m, s_{max})$
22    $s_{max} = top_{border} * min(co\_occ)$
23 }
24 return $topk$
```

Algorithm 1. Top-K Algorithm

score considering all previously seen scores plus the largest possible scores for t' coming from unexplored regions of the graph. We denote the *worstscore* of the tag currently at rank-k as *min-k*. A tag t' with $bestscore(t') < min\text{-}k$ can be safely dismissed since it has no chance of getting into the final top-k results (line 18).

The main difficulty in our scenario stems from the fact that the number of documents in the neighborhood is unknown. Considering summation as the aggregation function is then problematic since it does not allow for pruning candidate tags during the top-k query processing, as the expected score mass is potentially unbounded. In our algorithm, the document queue describes, at any point in time, the borderline to the currently unexplored graph regions (cf. the solid black circles in Figure 2). Since we cannot look beyond that line, we are not aware of the tags and documents occurring there. However, we know the score of the best path (or of the multiple best paths) to one (or more) of the documents at the border line, coined top_{border}. The *bestscore* (upper bound score) of a tag t' observed so far m' times with *worstscore* (lower bound score) is based on the most promising document on the borderline and on the co-occurrence value of the tag w.r.t. the initial document. A tag that has been seen m' out of m possible observations can at most gain $m - m'$ times the current maximum score, which is similar to existing work, solely based on the tag scores. More formally we can write:

$$bestscore = worstscore + (m - m')\left(\frac{top_{border}}{log(1 + dist(d_{init}, d_{border}))} * log(1 + co\_occ)\right).$$

5 Experiments

5.1 Experimental Setup

We experimentally confirm the validity of our approach using two datasets which we have crawled in 2007 from two popular tagging portals. All algorithms have been implemented in Java 1.6. The simulation ran on a 2x2.33 GHz Quad-Core Intel Xeon CPU with 8GB RAM. The data is stored in an Oracle 11g database.

We consider the following real-world datasets:

CiteULike: This dataset consists of documents and tags from CiteULike, a tagging portal for academic publication. We take advantage of the bibliographic information, i.e., citations, to define the document graph: we add an edge $d_i \rightarrow d_j$ whenever d_i cites d_j. We automate the citation extraction by relying on the citation information available on CiteSeer. Our dataset consists of approximately $540,000$ papers annotated with $195,000$ distinct tags. From this set of papers we have randomly crawled a total of 2200 pdf documents which we indexed using *Lucene* (*http://lucene.apache.org*).

Del.icio.us: We have obtained the del.icio.us crawl from [19]. This dataset consists of approximately $120,000$ html pages, annotated by users with $59,143$ distinct tags. We have built the document graph by adding an edge $d_i \rightarrow d_j$ whenever a page d_i has a hyperlink pointing to another page d_j. The Web pages were crawled and indexed using *Lucene.*

We tested our approach having three specific metrics in mind:

– *Precision:* We considered the "precision at k" of the tag propagation, which is defined as follows: $precision@k = \#relevant\_tags/k$, where the relevance of the tag is either taken from a human expert evaluation or automatically computed. In order to have an assessment as fair as possible, we disregard all inferred tags that were among the initial tags of the document, i.e., we concentrate on k *new* tags systematically.
– *NDCG:* In addition to the precision we also report on the NDCG (Normalized Discounted Cumulative Gain) [12]. This measure takes into consideration the ranking of the results in addition to their relevance.
– *Number of documents visited:* To assess the practical viability of our algorithm, we decided to measure the number of nodes visited in the document graph during the tag inference process. This was preferred over CPU or processing time in order to get easily reproducible results.

Note that we do not report on recall since (i) we focus on top-k retrieval, and (ii) the number of relevant tags for a document is unknown and potentially very large.

In order to compute the precision, we consider two different kinds of relevance assessments: human expert evaluation and automatic evaluation. In the first case we ask a group of experts to mark the newly inferred tags as relevant or non-relevant. This measure is referred to as *absolute precision.* Secondly, we make use of existing tags attached to documents in order to run an automatic precision evaluation. We "hide" parts of the already assigned sets of tags, then try to infer new tags with our algorithm. We use the "hidden" tags to assess the relevance of the inferred ones, i.e., an inferred tag is considered relevant only if it occurred in the set of initial (but hidden) tags. This is actually a *relative precision* measure, since it is reported w.r.t. to the set of already assigned tags. Naturally, the relative precision values tend to be smaller, as potentially relevant tags are deemed irrelevant because they did not appear in the initial set of tags. However this approach allows us to run experiments on a larger scale which is difficult to achieve when relying on manual relevance assessments.

As an illustrative example, consider for instance the paper entitled "Routing Indices For Peer-to-Peer Systems". The set of initial tags contained the tags {*"index", "search", "20050923", "p2p", "p2p-search", "20050923", "peer-to-p", "semantic", "survey", "unstructured-network"*} which our algorithms extended for $k = 5$ with the additional tags {*"network", "search", "distributed", "security", "system"*}, yielding a precision of 80%.

Algorithms under Comparison. Our experiments start by randomly selecting a set of initial documents (academic papers or bookmarked Web pages). For each of these initial documents we run our algorithm to infer the top-k most relevant tags which are new for the initial document (i.e., not in the set of already assigned tags). We always report on the average values of precision@k, NDCG@k and visited neighbors for the set of initial documents.

In the following we present the set of algorithms on whose performance we will later report. Besides our approach, we include a set of baselines which implement different measures of tag relevance for a given document.

Document Similarity Baseline: Our first baseline ignores the document graph and only considers the document similarity as an indicator for tag relevance. For each initial document d we compute a ranked list of the most similar documents $d' \in \mathcal{D}$ from the dataset. As a measure for document similarity we employed the cosine similarity $sim(d, d')$ presented in Section 3.1. In order to build a set of the most relevant k tags, we traverse the ranked list starting with the most similar document and copy its tags $T(d')$ into our top-k list. We stop when we have copied k new tags and compute the corresponding precision@k. In case the last document selected delivers more than k tags we randomly select among them.

Tag Co-Occurrence Baseline: Our second baseline is build upon the tag co-occurrence as a measure of relevance of a tag t' to a document d. For each of the initial documents d we build a ranked list of tags which have the highest co-occurrence scores $co\_occ(t', T(d))$ w.r.t. the initial set of tags $T(d)$ of the document. These scores are computed as described in Section 3.1. The precision@k is computed by selecting the first k tags from this list.

Tag Overlap Baseline: Our last baseline considers those documents which have the highest number of tags in common with the initial document as good candidates for tag propagation. Therefore we rank all documents in our dataset $d' \in \mathcal{D}$ according to the size of the intersection between their sets of tags $T(d')$ and the set of tags $T(d)$ of the initial document d: $I = T(d) \cap T(d')$. We then compose our top-k list by copying the remaining tags from the set $T(d') \setminus I$.

HAMLET: This is our ranking and graph traversal algorithm which combines the measures of tag relevance and the top-k ranking algorithm presented in the previous sections. For each of the initial documents d we explore the neighborhood $N(d)$ in a greedy best-first manner by carefully selecting only those neighbors d' which have a high chance of delivering potentially relevant tags for the initial document according to our principles of tag relevance. Our algorithm stops when there are no more possible top-k tags to be discovered.

5.2 Experimental Results

We start by presenting the results obtained for the *citeulike* benchmark where we considered 30 initial documents and collected manual assessments of tag relevance from 20 colleagues in order to evaluate the precision@k. Table 1 reports on the precision@k and NDCG@k values and on the number of visited neighbors for HAMLET. Our precision@k is promising in particular for $k < 10$, where we achieve 73% for $k = 3$ and 65% for $k = 5$. Intuitively, the precision values decrease with a higher k.

The parameter m, which controls the number of occurrences that are considered for a tag, influences the number of documents that are visited during run-time, e.g., for $k = 3$ our algorithm visits only 41 neighbors when $m = 3$ and 293 for $m = 10$. This is expected since m directly influences the upper bound scores of observed tags which are essential for the candidate pruning and hence the stopping of the algorithm. With m large, a higher number of tags is kept in the candidate queue as their *bestscore* might still be larger than the score of the current tag at rank-k (*min-k*). However, the influence of m on the precision@k is almost negligible which supports the initial hypothesis that the most promising

Table 1. Precision@k, NDCG@k and number of visited neighbors for the **citeulike** benchmark (30 initial documents) with manual precision evaluation

m	k	Precision@k	NDCG	Neighbors
3	3	0.73	0.74	41
3	5	0.65	0.68	93
3	7	0.55	0.60	74
3	10	0.51	0.58	217
3	15	0.44	0.53	234
5	3	0.70	0.72	124
5	5	0.65	0.69	153
5	7	0.57	0.63	247
5	10	0.52	0.59	328
5	15	0.45	0.53	386
7	3	0.72	0.74	243
7	5	0.65	0.69	257
7	7	0.57	0.64	356
7	10	0.51	0.59	421
7	15	0.46	0.54	486
10	3	0.74	0.76	293
10	5	0.65	0.69	385
10	7	0.57	0.64	468
10	10	0.51	0.59	517
10	15	0.46	0.54	543

Table 2. Relative Precision@k, relative NDCG@k and number of visited neighbors for the **del.icio.us** benchmark (140 initial documents) with automatic precision evaluation

m	k	Rel. Prec.@k	NDCG	Neighbors
3	3	0.50	0.51	1.65
3	5	0.42	0.46	1.65
3	7	0.36	0.42	1.67
3	10	0.30	0.37	1.67
3	15	0.23	0.31	1.67
5	3	0.50	0.51	1.67
5	5	0.42	0.46	1.65
5	7	0.36	0.42	1.67
5	10	0.30	0.37	1.67
5	15	0.23	0.31	1.67
7	3	0.50	0.51	1.67
7	5	0.42	0.46	1.65
7	7	0.36	0.42	1.67
7	10	0.30	0.37	1.67
7	15	0.23	0.31	1.67
10	3	0.50	0.51	1.67
10	5	0.42	0.46	1.66
10	7	0.36	0.42	1.67
10	10	0.30	0.37	1.67
10	15	0.23	0.31	1.67

Table 3. Precision@k for the three baselines for the **citeulike** benchmark (30 initial documents) with manual precision evaluation. Summarized HAMLET performance for ease of comparison.

k	CoOcc	DocDoc	Overlap	HAMLET
3	0.53	0.20	0.14	0.70 - 0.74
5	0.39	0.20	0.16	0.65
7	0.31	0.20	0.16	0.55 - 0.57
10	0.25	0.17	0.15	0.51 - 0.52
15	0.22	0.17	0.14	0.44 - 0.46

Table 4. NDCG@k for the three baselines for the **citeulike** benchmark (30 initial documents) with manual precision evaluation. Summarized HAMLET performance for ease of comparison.

k	CoOcc	DocDoc	Overlap	HAMLET
3	0.54	0.19	0.15	0.74 - 0.76
5	0.45	0.19	0.16	0.69 - 0.68
7	0.39	0.19	0.16	0.60 - 0.64
10	0.35	0.17	0.16	0.58 - 0.59
15	0.31	0.17	0.15	0.53 - 0.54

tags are actually found in the close vicinity of a document and that exploring further regions does not contribute substantially to the top-k result.

The results of the three baseline algorithms for the *citeulike* experiment are presented in Table 3 (precision) and Table 4 (NDCG). As expected, the precision values of the individual baselines are much lower than those achieved by our combined scoring scheme. This is mainly due to the fact that without any links between the documents we loose the context information captured by document graph and therefore the quality of the chosen tags, based only on global statistics, drops considerably. Still, the co-occurrence baseline achieves a precision of 53% which supports the general claim that the co-occurrence score of two tags is closely related to their semantics.

For the *del.icio.us* dataset we took advantage of the fact that popular bookmarks are annotated with a large set of tags and we implemented the above described automatic method for evaluating our precision. We selected bookmarks which were annotated with more than 20 and less than 40 tags and for each of them we randomly chose 4 tags (corresponding to the average number of tags per document in the whole dataset) as an input for our algorithm and disregarded the rest. We then ran our algorithm for each initial document and its 4 initial tags and computed the relative precision@k w.r.t. the hidden tags. We expect the real precision values to be higher, since the set of relevant tags for a document is potentially much bigger than the tags which have already been assigned by users. Table 2 reports on the results of our algorithm on the *del.icio.us* benchmark consisting of 140 initial documents and for which we automatically assessed the relative precision@k and NDCG@k. This second set of results shows the performance of our algorithm (e.g., 50% precision for $k = 3$) when ran on a larger scale. As described above this is a measure of relative precision as we use as the ground truth the set of initial tags already attached to a document. This set of tags presumably represents only a small portion of all the relevant tags for that document. Another observation is that the number of visited neighbors is much lower and remains constant as compared to the *citeulike* experiment. This is caused by the sparsely annotated general Web graph, i.e., documents have often only a few neighbors which are also tagged in *del.icio.us*. Therefore, in this the setting parameter m looses its importance since the annotated neighborhood of a document is small. However, our reported relative precision proves that our approach works well in both settings and does not depend significantly on the properties of the considered datasets.

Table 5. Relative Precision@k for the three baselines for the **del.icio.us** benchmark (140 initial documents) with automatic precision evaluation. Summarized HAMLET performance for ease of comparison.

k	CoOcc	DocDoc	Overlap	HAMLET
3	0.04	0.006	0.13	0.50
5	0.03	0.005	0.13	0.42
7	0.02	0.003	0.13	0.36
10	0.02	0.004	0.12	0.30
15	0.02	0.004	0.11	0.23

Table 6. Relative NDCG@k for the three baselines for the **del.icio.us** benchmark (140 initial documents) with automatic precision evaluation. Summarized HAMLET performance for ease of comparison.

k	CoOcc	DocDoc	Overlap	HAMLET
3	0.05	0.007	0.13	0.51
5	0.04	0.006	0.13	0.46
7	0.04	0.005	0.13	0.42
10	0.03	0.005	0.13	0.37
15	0.03	0.005	0.12	0.31

The results of the baseline algorithms for the *del.icio.us* benchmark can be observed in Table 5 (precision) and Table 6 (NDCG). Similarly to the previous experiment the relative precision@k values are much lower than those achieved by our algorithm. The baseline precisions are very low in this case and this is mainly due to the heterogeneity of the *del.icio.us* dataset. While *citeulike* represents a more restrictive domain of academic publications, our *del.icio.us* dataset represents a small subset of the Web, which means that the global measures will give far less accurate results. Our algorithm achieves higher precision values by limiting the scope of the tag inference to the neighborhood of a document and thus filtering out tags which are not directly related to that particular context.

In general, we can consider our method as accurate, given the inherent complexity of the task at hand. By taking advantage of a user-defined document graph and devising specific scoring and pruning methods, we reach a precision of 73% for the newly inferred tags when we use manual relevance assessments and 50% relative precision in the automatic case. Furthermore, our approach operates without any domain-specific knowledge, and with a number of computational operations sufficiently low to accommodate large settings.

6 Conclusions

Retrieving information from Web 2.0 applications opens up new challenges, as well as many exciting opportunities. We have put forward HAMLET, an approach for metadata propagation which is based on a socially-driven graph relating similar documents. The techniques we have developed throughout this work can potentially be extended beyond our showcase scenario focusing on academic papers and html bookmarks. As concrete examples, we believe it would be reasonable to extend our algorithm to images, songs, or movies explored online by communities of users. In addition, our approach could be used to rank existing tags of a document based on their support from the neighborhood.

References

1. Angelova, R., Weikum, G.: Graph-based text classification: learn from your neighbors. In: SIGIR (2006)
2. Bao, S., Xue, G., Wu, X., Yu, Y., Fei, B., Su, Z.: Optimizing web search using social annotations. In: WWW (2007)

3. Chirita, P.A., Costache, S., Nejdl, W., Handschuh, S.: P-tag: large scale automatic generation of personalized annotation tags for the web. In: WWW (2007)
4. Davison, B.D.: Topical locality in the web. In: SIGIR (2000)
5. Fagin, R.: Combining fuzzy information from multiple systems. J. Comput. Syst. Sci. 58(1) (1999)
6. Fagin, R., Lotem, A., Naor, M.: Optimal aggregation algorithms for middleware. J. Comput. Syst. Sci. 66(4) (2003)
7. Golder, S., Huberman, B.A.: Usage patterns of collaborative tagging systems. Journal of Information Science 32(2) (2006)
8. Halpin, H., Robu, V., Shepherd, H.: The complex dynamics of collaborative tagging. In: WWW (2007)
9. Heymann, P., Koutrika, G., Garcia-Molina, H.: Can social bookmarking improve web search? In: WSDM (2008)
10. Heymann, P., Ramage, D., Garcia-Molina, H.: Social tag prediction. In: SIGIR (2008)
11. Hotho, A., Jäschke, R., Schmitz, C., Stumme, G.: Information retrieval in folksonomies: Search and ranking. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 411–426. Springer, Heidelberg (2006)
12. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Trans. Inf. Syst. 20(4) (2002)
13. Jäschke, R., Marinho, L.B., Hotho, A., Schmidt-Thieme, L., Stumme, G.: Tag recommendations in folksonomies. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS, vol. 4702, pp. 506–514. Springer, Heidelberg (2007)
14. Kinsella, S., Budura, A., Skobeltsyn, G., Michel, S., Breslin, J.G., Aberer, K.: From Web 1.0 to Web 2.0 and back - how did your grandma use to tag? In: WIDM (2008)
15. Kleinberg, J.M., Tardos, É.: Approximation algorithms for classification problems with pairwise relationships: Metric labeling and markov random fields. In: FOCS (1999)
16. Marlow, C., Naaman, M., Boyd, D., Davis, M.: Ht06, tagging paper, taxonomy, flickr, academic article, to read. In: HYPERTEXT (2006)
17. Mika, P.: Ontologies Are Us: A Unified Model of Social Networks and Semantics. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 522–536. Springer, Heidelberg (2005)
18. Rattenbury, T., Good, N., Naaman, M.: Towards automatic extraction of event and place semantics from flickr tags. In: SIGIR (2007)
19. Schenkel, R., Crecelius, T., Kacimi, M., Michel, S., Neumann, T., Parreira, J.X., Weikum, G.: Efficient top-k querying over social-tagging networks. In: SIGIR (2008)
20. Schmitz, P.: Inducing ontology from flickr tags. In: Workshop on Collaborative Tagging at WWW (2006)
21. Sigurbjörnsson, B., van Zwol, R.: Flickr tag recommendation based on collective knowledge. In: WWW (2008)
22. Song, Y., Zhuang, Z., Li, H., Zhao, Q., Li, J., Lee, W., Giles, C.: Real-time automatic tag recommendation. In: SIGIR (2008)
23. Wu, X., Zhang, L., Yu, Y.: Exploring social annotations for the semantic web. In: WWW (2006)
24. Yanbe, Y., Jatowt, A., Nakamura, S., Tanaka, K.: Towards improving web search by utilizing social bookmarks. In: Baresi, L., Fraternali, P., Houben, G.-J. (eds.) ICWE 2007. LNCS, vol. 4607, pp. 343–357. Springer, Heidelberg (2007)