Social Tagging Recommender Systems

Leandro Balby Marinho, Alexandros Nanopoulos, Lars Schmidt-Thieme, Robert Jäschke, Andreas Hotho, Gerd Stumme, and Panagiotis Symeonidis

Abstract The new generation of Web applications known as (STS) is successfully established and poised for continued growth. STS are open and inherently social; features that have been proven to encourage participation. But while STS bring new opportunities, they revive old problems, such as information overload. Recommender Systems are well known applications for increasing the level of relevant content over the "noise" that continuously grows as more and more content becomes available online. In STS however, we face new challenges. Users are interested in finding not only content, but also tags and even other users. Moreover, while traditional recommender systems usually operate over 2-way data arrays, STS data is represented as a third-order tensor or a hypergraph with hyperedges denoting (user, resource, tag) triples. In this chapter, we survey the most recent and state-of-the-art work about a whole new generation of recommender systems built to serve STS. We describe (a) novel facets of recommenders for STS, such as user, resource, and tag recommenders, (b) new approaches and algorithms for dealing with the ternary nature of STS data, and (c) recommender systems deployed in real world STS. Moreover, a concise comparison between existing works is presented, through which we identify and point out new research directions.

Leandro Balby Marinho · Alexandros Nanopoulos · Lars Schmidt-Thieme
Information Systems and Machine Learning Lab (ISMLL), University of Hildesheim, Marienburger Platz 22, 31141 Hildesheim, Germany, http://www.ismll.uni-hildesheim.de, e-mail: {marinho,nanopoulos,schmidt-thieme}@ismll.uni-hildesheim.de

Robert Jäschke · Andreas Hotho · Gerd Stumme
Knowledge & Data Engineering Group (KDE), University of Kassel, Wilhelmshöher Allee 73, 34121 Kassel, Germany, http://www.kde.cs.uni-kassel.de, e-mail: {jaeschke,hotho,stumme}@cs.uni-kassel.de

Panagiotis Symeonidis
Department of Informatics, Aristotle University, 54124 Thessaloniki, Greece, e-mail: symeon@csd.auth.gr

1 Introduction

With the advent of affordable domestic high-speed communication facilities, inexpensive digitization devices, and the open access nature of the Web, a new and exciting family of Web applications known as Web 2.0 has been born. The underlying idea is to decentralize and cheapen content creation, thus leading the Web into a more open, connected, and democratic environment. In this chapter we will focus on a particular family of Web 2.0 applications known as Social Tagging Systems (STS for short). STS assign a major role to the ordinary user, who is not only allowed to publish and edit resources, but also and more importantly, to create and share lightweight metadata in the form of freely chosen keywords called *tags*. The exposure of users to both tags and resources creates a fundamental trigger for communication and sharing, thus lowering the barriers to cooperation and contributing to the creation of collaborative lightweight knowledge structures known as *folksonomies*[1]. Some notable examples of STS are sites like Delicious[2], BibSonomy[3], and Last.fm[4], where Delicious allows the sharing of bookmarks, BibSonomy the sharing of bookmarks and lists of literature, and Last.fm the sharing of music. These systems are characterized by being easy to use and free to anyone willing to participate. Once a user is logged in, he can add a resource to the system, and assign arbitrary tags to it.

If on the one hand this new family of applications brings new opportunities, it revives old problems on the other, namely the problem of information overload. Millions of individual users and independent providers are flooding STS with content and tags in an uncontrolled way, thereby lowering the potential for content retrieval and information sharing. One of the most successful approaches for increasing the level of relevant content over the "noise" that continuously grows as more and more content becomes available online lies on Recommender Systems (RS for short). In STS however, we face several new challenges. Users are interested in finding not only content, but also tags, and even other users. Moreover, while traditional RS usually operate over 2-way data arrays, folksonomy data is represented as a third-order tensor or a hypergraph with hyperedges denoting (user, resource, tag) triples. Furthermore, while there is an extensive literature for rating prediction based on explicit user feedback, i.e., a numerical value denoting the degree of preference of a user for a given item, in folksonomies there are usually no ratings. Thus, before arguing why not to simply use an old solution to a recurrent problem, we need to investigate to which extent the traditional RS paradigm and approaches apply to STS.

Social tagging recommender systems is a young research area that has attracted significant attention recently, which is expressed by the increasing number of publications (e.g., [15, 11, 36, 34, 30]) and is poised for continued growth. Furthermore,

[1] The term folksonomy refers to a blend of the two words folk and taxonomy, i.e., a collaborative classification system created and maintained by ordinary users.

[2] http://delicious.com/

[3] http://www.bibsonomy.org/

[4] http://www.last.fm/

real and large scale STS, such as Delicious, BibSonomy, and Last.fm, for example, already offer some recommender services to their users, which implies an increasing commercial interest in the area. In this chapter we survey in a concise manner, the most recent and state-of-the-art work about a whole new generation of RS built to serve STS. We describe: (a) novel facets of RS for STS, such as user, resource, and tag recommenders, (b) the challenges for deploying RS in real-world STS, (c) new approaches and algorithms for dealing with the inherent ternary relational data of folksonomies, and (d) approaches for tag acquisition. Emphasis is given on presenting a concise comparison between existing works, through which we identify and point out new research directions.

The chapter is structured as follows. In Section 2 we characterize the data structure of folksonomies and point out some of the differences between the traditional RS paradigm and social tagging RS. In Section 3 we discuss the challenges of deploying RS in real world STS and present the BibSonomy system as a study case. Section 4 presents several families of social tagging RS, such as: graph/content-based algorithms for recommending users, resources or tags. Section 5 provides comparisons and discussions about the algorithms presented in Section 4; and finally Section 6 closes the chapter pointing out new directions of research in this area.

2 Social Tagging Recommenders Systems

Folksonomies are the underlying structures of STS and result from the practice of collaboratively creating tags to annotate and categorize content. Tags, in general, are a way of grouping content by category to make them easy to view by topic. This is a grassroot approach to organize a site and help users find content they are interested in. Note that with the introduction of tags, the usual binary relation between users and resources, which is largely exploited by traditional RS, turns into a ternary relation between users, resources, and tags.

Since tags are voluntarily and freely provided by ordinary users, problems such as unwillingness to tag and diverging vocabulary can easily arise. As we will see in the course of this chapter, a possible way to address these problems is through tag RS. Tags also represent additional and personalized information about resources, which if properly exploited, can eventually boost the performance of resource RS. But before we delve into how RS can deal and benefit from the additional information provided by tags, we need to formally define folksonomies and its data structures, elaborate on the differences between traditional RS and social tagging RS, and the challenges involved in deploying RS in real world STS; topics which are covered in the following sections.

2.1 Folksonomy

Formally, a *folksonomy* is a tuple $\mathbb{F} := (U, T, R, Y)$ where

- U, T, and R are non-empty finite sets, whose elements are called *users*, *tags*, and *resources*, resp., and
- Y is a ternary relation between them, i. e., $Y \subseteq U \times T \times R$, whose elements are called tag assignments.[5]

Users are typically described by their user ID, and tags may be arbitrary strings. What is considered a resource depends on the type of system. For instance, in Delicious, the resources are URLs, in BibSonomy URLs or publication references, and in Last.fm, the resources can be artists, song tracks or albums.

Folksonomy data can be represented in different ways, and as we will see in Section 4, each representation can lead to different recommendation algorithms.

Folksonomies as Tensors The set of triples in Y can be represented as third-order *tensors* (3-dimensional arrays) $\mathcal{A} = (a_{u,t,r}) \in \mathbb{R}^{|U|\times|T|\times|R|}$. There are different ways to represent Y as a tensor (see left-hand sinde of Figure 1). Symeonidis et al. [34], for example, proposed to interpret Y as a sparse tensor in which 1 indicates positive feedback and 0 missing values:

$$a_{u,t,r} = \begin{cases} 1, & (u,t,r) \in Y \\ 0, & \text{else} \end{cases}$$

Rendle *et al.* [26], on the other hand, distinguish between positive/negative examples and missing values in order to learn personalized ranking of tags (see Section 4). The idea is that positive and negative examples are only generated from observed tag assignments. Observed tag assignments are interpreted as positive feedback, whereas the non observed tag assignments of an already tagged resource are negative evidences. All other entries are assumed to be missing values (see right-hand side of Figure 1).

Note that in folksonomies, differently from typical RS, there are usually no numerical ratings indicating the explicit preference of a user for a given resource/tag.

Folksonomies as Hypergraphs An equivalent, but maybe more intuitive representation of a folksonomy, is a tripartite (undirected) hypergraph $G := (V, E)$, where $V := U \dot\cup T \dot\cup R$ is the set of nodes, and $E := \{\{u, t, r\} \mid (u, t, r) \in Y\}$ is the set of hyperedges (see Figure 2).

[5] In the original definition [12], it is introduced additionally a subtag/supertag relation, which we omit here. The version used here is known in Formal Concept Analysis [7] as a *triadic context* [21, 33].

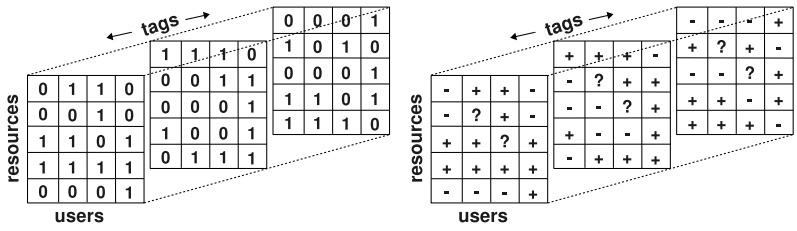


Fig. 1 *Left* [34]: 0/1 sparse tensor representation where positive feedback is interpreted as 1 and the remaining data as 0. *Right* [26]: Non observed tag assignments for a given already tagged resource are negative examples. All other entries are missing values.

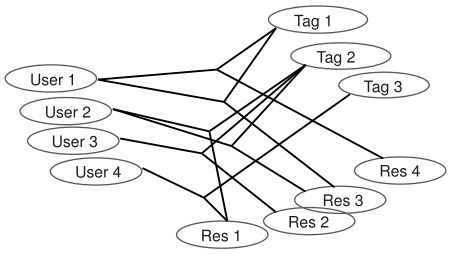


Fig. 2 Tripartite undirected hypergraph representation of a folksonomy.

2.2 The Traditional Recommender Systems Paradigm

Recommender systems are software applications that aim at predicting the user interest for a particular resource based on a collection of user profiles, e.g., the user's history of purchase/resources' ratings, click-stream data, demographic information, and so forth. Usually RS predict ratings of resources or suggest a list of new resources that the user hopefully will like the most. Traditionally, for m users and n resources, the user profiles are represented in a sparse user-resource matrix $\mathbf{X} \in \mathbb{R}^{m \times n} \cup \{.\}$, where $\{.\}$ denote missing values. The matrix can be decomposed into row vectors:

$$\mathbf{X} := [\mathbf{x}_1, ..., \mathbf{x}_m]^{\mathrm{T}} \text{ with } \mathbf{x}_u := [x_{u,1}, ..., x_{u,n}], \text{ for } u := 1, \ldots, m,$$

where $x_{u,r}$ indicates that user u rated resource r by $x_{u,r} \in \mathbb{R}$. Each row vector $\mathbf{x}_u$ corresponds thus to a user profile representing the resource's ratings of a particular user. This decomposition usually leads to algorithms that leverage user-user similarities, such as the well known user-based collaborative filtering (CF) [27]. The matrix can alternatively be represented by its column vectors:

$$\mathbf{X} := [\mathbf{x}_1, ..., \mathbf{x}_n] \text{ with } \mathbf{x}_r := [x_{1,r}, ..., x_{m,r}]^{\mathrm{T}}, \text{ for } r := 1, \ldots, n,$$

in which each column vector $\mathbf{x}_r$ corresponds to a specific resource's ratings by all m users. This representation usually leverages item-item similarities and leads to

item-based CF algorithms [3]. For a survey on neighborhood-based recommendation methods, such as CF, see Chapter **??**.

Note that because of the ternary relational nature of folksonomies, traditional RS cannot be applied directly. Therefore, in order to develop RS for folksonomies, one needs to either (i) reduce the ternary relation Y to a lower dimensional space (usually second-order tensors) where traditional RS can be applied, or develop new algorithms that operate over third-order tensors or tripartite undirected hypergraphs. Note that if one follows (i), care must be taken during the dimensionality reduction since important information can be discarded, which can lower the overall accuracy of the recommendations. In Section 4 we present and discuss both families of algorithms.

2.3 Multi-mode Recommendations

Differently from the traditional RS paradigm, where one is usually concerned only with rating prediction or resource recommendations, STS users may be interested in finding resources/tags, or even other users, and therefore recommendations can be provided for any of these entity types.

The recommendation of tags is used in several systems, like Delicious and BibSonomy, for example. It usually involves the recommendation of tags to users, based on the tags other users have provided for the same resources. Tag recommendations can expose different facets of an information item and relieve users from the obnoxious task of coming up with a good set of tags. Moreover, tag recommendation can reduce the problem of tag sparsity, which results from the unwillingness of users to tag. Figure 5 illustrates tag recommendations in BibSonomy.

It is important to note that differently from traditional RS, where there is usually no repeat-buying, i.e., the user usually does not buy the same book, movie, CD, etc. twice, re-occurring tags are a common feature of STS. A tag that has already been used to annotate a resource can be reused to annotate other different resources. This means that while traditional RS usually only recommend items that the user has not yet bought or rated, tag recommenders can eventually recommend tags that the user has already used for other resources.

The recommendation of resources is largely used in e-commerce and advertising, like in Amazon for example. With the actual trend towards STS, the current resource recommendation services will also be able exploit the tags to boost the recommendation quality, for example, by recommending resources to users based on the tags they have in common with other similar users. The movie recommendation website movielens[6], where users rate the movies they like and receive recommendations about other movies in which they might be interested, is a notable example. It started as a traditional recommender service operating over the typical user-rating

[6] http://www.movielens.org

binary matrix, and just recently added social tagging features, whereby new tag-aware algorithms are being developed and deployed [29].

A third type of recommendation concerns recommending interesting users to a target user, which can help to connect people with common interests and encourage them to contribute and share more content. With the term *interesting* users, we mean those users who have similar profile to the target user. If a set of tags is frequently used by many users, for example, then these users implicitly form a group of users with common interests, even though they may not have any physical or online connections. The tags represent the common interests to this user group.

Each mode of recommendation, i.e., tag, resource, or user, is useful, depending of course on the context of the particular application. Algorithms that are able to provide integrated multi-mode recommendations are very appealing, as one can spare the effort of implementing and maintaining several mode-specific recommender systems.

3 Real World Social Tagging Recommender Systems

3.1 What are the Challenges?

For a recommender system to be successful in a real world application, it must approach several challenges. First, the provided recommendations must match the situation, i.e., tags should describe the annotated resource, products should awake the interest of the user, suggested resources should be interesting and relevant. Second, the suggestions should be traceable such that one easily understands why he got the items suggested. Third, they must be delivered timely without delay and they must be easy to access (i.e., by allowing the user to click on them or to use tab-completion when entering tags). Furthermore, the system must ensure that recommendations do not impede the normal usage of the system.

In this section we focus on tag recommendations as example of recommenders in STS. Most STS contain a tag recommender which suggests tags to the user when she is annotating a resource. Recommending tags can serve various purposes, such as: increasing the chances of getting a resource annotated, reminding a user what a resource is about, and consolidating the vocabulary across the users. Furthermore, as Sood et al. [32] point out, tag recommendations "fundamentally change the tagging process from generation to recognition" which requires less cognitive effort and time.

More formally, given a user u and a resource r, the task of a tag recommender is to predict the tags $\mathrm{tags}(u, r)$ the user will assign to the resource. We will depict the (ordered!) set of recommended tags by $\hat{T}(u, r)$. Although we do not take the order of tags as the user entered them into account, the order of tags as given by the recommender plays an important role for the evaluation.

Fig. 3 detail showing a single bookmark post

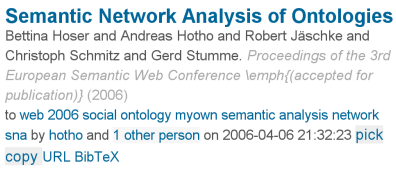


Fig. 4 detail showing a single publication post

3.2 BibSonomy as Study Case

3.2.1 System Description

BibSonomy started as a students project at the Knowledge and Data Engineering Group of the University of Kassel[7] in spring 2005. The goal was to implement a system for organizing BIBTEX [25] entries in a way similar to bookmarks in Delicious – which was at that time becoming more and more popular. BIBTEX is a popular literature management system for LATEX [20], which many researchers use for writing scientific papers. After integrating bookmarks as a second type of resource into the system and upon the progress made, BibSonomy was opened for public access at the end of 2005 – first announced to collegues only, later in 2006 to the public.

A detailed view of one bookmark post in BibSonomy can be seen in Figure 3. The first line shows in bold the title of the bookmark which has the URL of the bookmark as underlying hyperlink. The second line shows an optional description the user can assign to every post. The last two lines belong together and show detailed information: first, all the tags the user has assigned to this post (*web, service, tutorial, guidelines* and *api*), second, the user name of that user (*hotho*) followed by a note, how many users tagged that specific resource. These parts have underlying hyperlinks, leading to the corresponding tag pages of the user, the users page and a page showing all four posts (i. e., the one of user *hotho* and those of the three other people) of this resource. The structure of a publication post is very similar, as seen in Figure 4.

3.2.2 Recommendations in BibSonomy

To support the user during the tagging process and to facilitate the tagging, BibSonomy includes a tag recommender (see Figure 5). When a user finds an interesting web page (or publication) and posts it to BibSonomy, the system offers up to ten recommended tags on the posting page.

[7] http://www.kde.cs.uni-kassel.de/

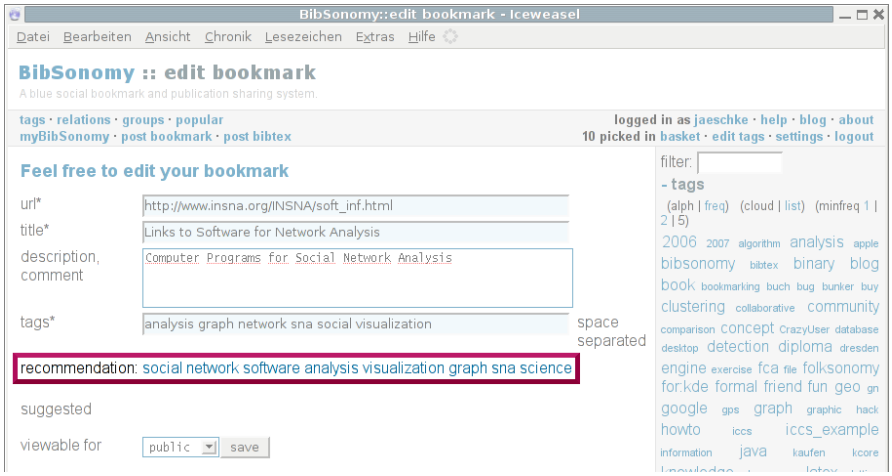


Fig. 5 Tag recommendations in BibSonomy during annotation of a bookmark.

3.2.3 Technological and Infrastructure Requirements

Implementing a recommendation service for BibSonomy required to tackle several problems, some of them we describe here.

First, having enough data available for recommendation algorithms to produce helpful recommendations is an important requirement one must address already in the design phase. The recommender needs access to the systems database and to what the user is currently posting (which could be accomplished, e.g., by (re)-loading recommendations using techniques like AJAX). Further data – like the full text of documents – could be supplied to tackle the cold-start problem (e.g., for content-based recommenders). The system must be able to handle large amounts of data, to quickly select relevant subsets and provide methods for preprocessing.

The available hardware and expected amount of data limits the choice of recommendation algorithms which can be used. Although some methods allow (partial) precomputation of recommendations, this needs extra memory and might not yield the same good results as online computation. Both hardware and network infrastructure must ensure short response times to deliver the recommendations to the user without too much delay. Together with a simple and non-intrusive user interface this ensures usability.

Further aspects which should be taken into account include implementation of logging of user events (e.g., clicking, key presses, etc.) to allow for efficient evaluation of the used recommendation methods in an online setting. Together with a live evaluation this also allows to tune the result selection strategies to dynamically choose the (currently) best recommendation algorithm for the user or resource at hand. The multiplexing of several available algorithms together with the simple inclusion of external recommendation services (by providing an open recommendation interface) is one of the recent developments in BibSonomy.

3.3 Tag Acquisition

The quality of tags can directly affect the recommendation performance of social tagging RS. Although folksonomies represent the "wisdom of crowds", social tagging can present problems, such as tag sparsity (users tend to provide a constrained number of tags), polysemy (tags are subject to multiple interpretations), or tag idiosyncrasy (tags used for personal organization like "to read", for example). All these problems can harm the quality of recommendations. For this reason, we consider alternative ways of acquiring tags. This will help us to better characterize the advantages and disadvantages of the social tagging process. We then examine the following tag acquisition methods:

- **Expert Tagging**: This approach usually relies on a small number of domain experts, who annotate resources using, mainly, structured vocabularies. Experts provide tags that are objective and cover multiple aspects. Pandora[8] is a notable example of a system that uses experts for tagging music resources. The main advantage of using experts is the resulting well agreed tag vocabulary. This comes, of course, at the cost of manual work, which is both time consuming and expensive.
- **Tagging based on annotation games**: Games with a purpose (GWAP) [38], like theESPGame[9], is a breakthrough idea to use a game to employ humans for the purpose of annotation. Two players observe simultaneously the same image and are asked to enter tags until they both enter the same tag. Following the success of ESPGame, several others appeared (e.g., ListenGame[10]) in the domain of music. Like social tagging, games exploit the "computational power of humans". By partnering two or more people, the resulting set of tags has the potential of being highly accurate. The problem with games is that players, opting for higher scores, may sacrifice the quality of tags. For example, they may enter more general tags in favor of more specific, just to increase the probability of match.
- **Content-based Tagging**: Resources like URLs, sound tracks, etc., contain a rich content. By crawling associated information from the Web and by converting this data into a suitable representation, tags can be generated using data mining algorithms. In the tag recommendation task of the ECML PKDD Discovery Challenge 2008[11], for example, some of the tags to be predicted in the test set never appeared in the training set, which forced the participants (e.g., [23]) to use the textual content of the resources to come up with new tags. In the music domain, this approach is called *auto-tagging* and has been proposed to avoid the cold-start problem [5]. The advantage of content-based tags is that no humans must be directly involved during the tagging process. The disadvantages are that these tags can be noisy, their computation is intensive, and users are forced to agree with the tags generated by the algorithms.

[8] http://www.pandora.com/

[9] http://www.gwap.com/gwap/gamesPreview/

[10] http://www.listengame.org/

[11] see http://www.kde.cs.uni-kassel.de/ws/rsdc08/.

Compared to the alternative methods, social tagging has the advantage of producing large-scale tag collections. The quality of tags generally improves with a large number of taggers. Nevertheless, social tagging is prone to the cold-start problem, as new resources are seldom tagged. In Table 1 we summarize the main advantages and disadvantages of the described approaches.

Table 1 Characterization of tag collection methods.

Method	Advantages	Disadvantages
Social tagging	scalable, "wisdom of crowds"	idiosyncrasy, polysemy, cold-start
Experts	accurate tags	expensive process, non scalable
Games	"wisdom of crowds", potentially scalable	cold-start, manipulation prone
Mined tags	automation, avoids cold-start	noisy, computationally intensive

Although social tagging is prone to idiosyncrasy, sparsity, and cold-start problems, the quality of tags generally improves with a large number of taggers. Furthermore, social tagging systems (as well as annotation games) represent a human computation paradigm with enormous potential to address problems that content-based mechanisms for tag acquisition cannot yet tackle on their own. But unlike computers, humans require some incentive to become part of the "collective computation", which can be provided, for example, through the recommendation of tags.

4 Recommendation Algorithms for Social Tagging Systems

As we pointed out in Section 2, there are some particularities in folksonomies that one needs to take into account before designing RS, such as:

- Folksonomy data is represented as tensors or tripartite undirected hypergraphs and thus one needs to either transform the data in order to apply traditional recommender algorithms or extend the existent methods to operate over tensors or hypergraphs.
- Folksonomy users might be interested in multi-mode recommendations, so algorithms that serve all modes with minor or no changes during mode switching are ideally desired.
- Folksonomies allow multi-media resources, and thereby content-based algorithms should be able to efficiently incorporate content information in the folksonomy data structure.

In this section we survey some of the most recent and prominent methods about social tagging RS, showing and discussing how they address the aforementioned issues.

4.1 Collaborative Filtering

Collaborative Filtering is one of the most used and successfully applied methods for personalized RS, for which a large and continuously active literature exists (see Chapters **??** and **??**). Basically, it is an algorithm for matching people with similar interests for the purpose of making recommendations. As pointed out in Section 2.2, traditional recommender systems typically operate on second-order tensors representing a binary relation between users and resources. Thus, because of the ternary relational nature of folksonomies, traditional CF cannot be applied directly, unless the ternary relation Y is reduced to a lower dimensional space [24]. To this end, in the case of user-based CF, we consider as matrix $\mathbf{X}$ alternatively the two 2-dimensional projections $\pi_{UR}Y \in \{0,1\}^{|U|\times|R|}$ with $(\pi_{UR}Y)_{u,r} := 1$ if there exists $t \in T$ s. t. $(u,t,r) \in Y$ and 0 else, and $\pi_{UT}Y \in \{0,1\}^{|U|\times|T|}$ with $(\pi_{UT}Y)_{u,t} := 1$ if there exists $r \in R$ s. t. $(u,t,r) \in Y$ and 0 else (Figure 6). One could eventually also consider the resource-tag projection matrix, what would lead to unpersonalized content-based models.

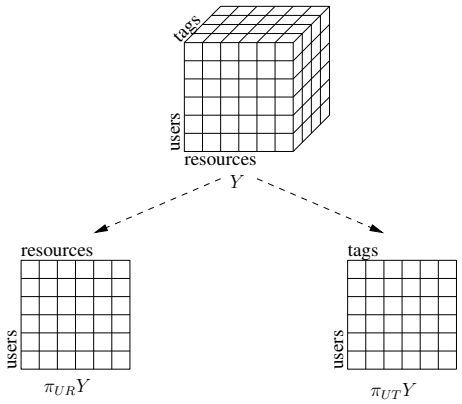


Fig. 6 Projections of Y into the user's resource and user's tag spaces.

The projections preserve the user information, and lead to RS based on occurrence or non-occurrence of resources or tags, resp., with the users. Notice that we have here two possible setups in which the k-neighborhood N_u^k of a user u can be formed, by considering either the resources or the tags as objects. Having defined matrix $\mathbf{X}$, and having decided whether to use $\pi_{UR}Y$ or $\pi_{UT}Y$ for computing user neighborhoods, we have the required setup to apply CF. We first compute, based on the row decomposed version of $\mathbf{X}$ and for a given k, the set N_u^k of the k users that are most similar to user u:

$$N_u^k := \underset{v \in U \setminus \{u\}}{\overset{k}{\operatorname{argmax}}} \operatorname{sim}(\mathbf{x}_u, \mathbf{x}_v) \tag{1}$$

where the superscript in the argmax function indicates the number $k \in \mathbb{N}$ of neighbors to be returned, and sim is any well defined similarity measure, such as, for example, the usual cosine similarity measure, i. e., $\mathrm{sim}(\mathbf{x}_u, \mathbf{x}_v) := \frac{\langle \mathbf{x}_u, \mathbf{x}_v \rangle}{\|\mathbf{x}_u\| \|\mathbf{x}_v\|}$.

Multi-mode Recommendations Having the neighborhood computed, we can extract the set $\hat{T}(u, r)$ of n recommended tags for a given user u, a given resource r, and some $n \in \mathbb{N}$, as follows:

$$\hat{T}(u, r) := \underset{t \in T}{\overset{n}{\mathrm{argmax}}} \sum_{v \in N_u^k} \mathrm{sim}(\mathbf{x}_u, \mathbf{x}_v) \delta(v, t, r) \tag{2}$$

where $\delta(v, t, r) := 1$ if $(v, t, r) \in Y$ and 0 else.

If one wants to recommend resources instead, the same principle used for tags can be applied. Note that if we use only the $\pi_{UR}Y$ projection, we would end up at the standard user-based CF algorithm (see Eq. 3). But since tags can provide additional information about user interests, they can eventually boost the recommendation quality and thereby should be exploited. A trivial tag-aware recommender method is to compute the user neighborhood based on the $\pi_{UT}Y$ projection matrix and aggregate the resources of the neighborhood to generate the recommendation list. A similar idea is presented in [6], where first the user-tag projection matrix $\pi_{UT}Y$ is used to compute a ranked list of tags, whereby the recommendation list of resources is extracted. But by using only $\pi_{UT}Y$ alone, one discards the resource information, which in this case, is the key mode of interest. In this sense, one needs to find a way to accommodate all the three modes of the folksonomy in a 2-way data structure so that standard CF can be applied. Tso-Sutter et al. [36] proposed an approach for doing that by extending the typical user-resource matrix with tags as pseudo users and pseudo resources (see Figure 7). Note that in this way, the user/resource profile is automatically enriched with tags. A fusion algorithm is then proposed for combining user-based CF (*ucf*) and item-based CF (*icf*) predictions over the extended matrix. Recall that in the standard user-based CF for the resource prediction problem, the interestingness score of a given user u for a particular resource r is computed as the averaged number of neighbors that co-occur with resource r, i.e.,

$$p^{\mathrm{ucf}}(x_{u,r} = 1) := \frac{|\{v \in N_u \,|\, x_{v,r} = 1\}|}{|N_u|}, \tag{3}$$

For item-based CF applied to the resource prediction problem, the algorithm suggested by [3] computes the interestingness score of a given user u for a particular resource r as the averaged sum of similarities between r and its neighboring resources N_r that co-occur with u, i.e.,

$$p^{\mathrm{icf}}(x_{u,r} = 1) := \sum_{\{r' \in N_r | x_{u,r'} = 1\}} \mathrm{sim}(r, r') \tag{4}$$

The fusion of these two scores is then computed by

$$p^{\text{iucf}}(x_{u,r}=1) := \lambda \cdot \frac{p^{\text{ucf}}(x_{u,r}=1)}{\sum_r p^{\text{ucf}}(x_{u,r}=1)} + (1-\lambda) \cdot \frac{p^{\text{icf}}(x_{u,r}=1)}{\sum_r p^{icf}(x_{u,r}=1)} \tag{5}$$

where λ is just a parameter controlling the influence of *ucf* or *icf*. Note that since the values of the prediction lists computed by *ucf* and *icf* have different units (user-based being the frequency of items and item-based the similarity of items), the predicted scores are normalized to unity. For some $n \in \mathbb{N}$, the top-n recommendation list is then generated by

$$\arg\max_r{}^{n} p^{\text{iucf}}(x_{u,r}=1) \tag{6}$$

A similar idea was proposed by Wetzker et al. [40], where the probabilistic latent semantic analysis (PLSA) model [10] is extended with tags for the recommendation of resources. In the standard PLSA, the probability that a resource co-occurs with a given user can be computed by

$$P(r|u) := \sum_z P(r|z)P(z|u), \tag{7}$$

where $Z := \{z_1, ..., z_q\}$ is a hidden topic variable and is assumed to be the origin of observed co-occurrence distributions between users and resources. The same hidden topics are then assumed to be the origin of resource/tag co-occurrences, i.e.,

$$P(r|t) := \sum_z P(r|z)P(z|t). \tag{8}$$

Both models are then combined on the common factor $P(r|z)$ by maximizing the log-likelihood function

$$L := \sum_r \left[\lambda \sum_u f(r,u) \text{log} P(r|u) + (1-\lambda) \sum_t f(r,t) \text{log} P(r|t) \right], \tag{9}$$

where $f(r,u)$ and $f(r,t)$ correspond to the co-occurrence counts between resources and users, and resources and tags respectively. Here λ is a predefined weight balancing the influence of each model. The usual Expectation-Maximization (EM) algorithm is then applied for performing maximum likelihood estimation for the model. Resources for a given user u are then weighted by the probability $P(r|u)$ (see Eq. 7), ranked, and the top ranked resources are finally recommended.

For recommending users, one can either recommend a neighborhood based on $\pi_{UT}Y$ or $\pi_{UR}Y$. In order to recommend a neighborhood that takes into account the three modes of the folksonomy, one could, for example, either use the matrix

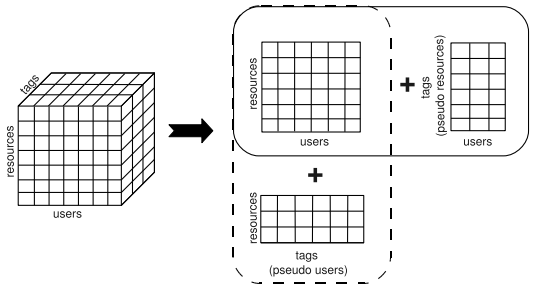


Fig. 7 Extending the user-resource matrix horizontally by including tags as pseudo resources and vertically by including tags as pseudo users.

extensions proposed by [36] (see Figure 7) or compute a linear combination of the user similarities based on the user-resource and user-tag projection matrices.

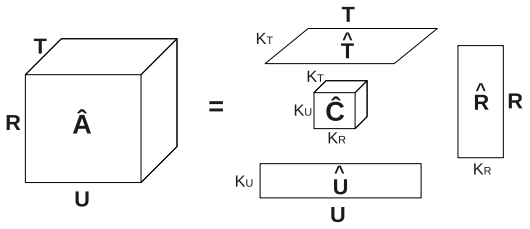
Remarks on Complexity CF usually suffers from scalability problems, given that the whole input matrix needs to be kept in memory. In STS, one may have to eventually keep more than one matrix in memory, depending on which kind of projections one wants to operate upon. To compute recommendations we usually need three steps:

1. Computation of projections: In order to compose the projections, we need to determine the $(u, r), (u, t)$ and/or (r, t) co-occurrences. For that, we just need to do a linear scan in Y.
2. Neighborhood computation: In traditional user-based CF algorithms, the computation of the neighborhood N_u is usually linear on the number of users as one needs to compute the similarity of a given test user with all the other users in the database. In addition, we need to sort the similarities in order to determine the k-nearest neighbors.
3. Recommendations: For predicting the top-n tag/resource recommendations for a given test user, we need to: (i) count the tags/resources co-occurrences with the nearest neighbors N_u, (ii) weight each co-occurrence by the corresponding neighbor similarity, and (iii) sort the tags/resources based on their weights (e.g., Eq. 2).

4.2 Recommendation based on Ranking

In the following we present recommendation algorithms that, inspired from Web ranking, base their recommendations on a ranking score. Their common characteristic is that the score is computed according to spectral attributes extracted from the underlying folksonomy data structure. However, the different ways to represent a folksonomy (see Section 2.1) can result in different ranking-based algorithms.

Fig. 8 Tensor Factorization.



4.2.1 Ranking based on Tensor Factorization

By representing Y as a tensor, one is able to exploit the underlying latent semantic structure in $\mathcal{A}$ formed by multi-way correlations between users, tags, and resources. This can be attained using recommendation algorithms that are based on tensor factorization, as the ones proposed in [26, 34, 42]. With such algorithms, multi-way correlations can be effectively detected, leading to improved performance. Chapter **??** presents several state-of-the-art methods for matrix factorization (i.e., second-order tensor factorization) for the problem of rating prediction.

The factorization of $\mathcal{A}$ is expressed as follows (see Figure 8):

$$\hat{\mathcal{A}} := \hat{C} \times_u \hat{U} \times_r \hat{T} \times_t \hat{R} \tag{10}$$

where $\hat{U} \in \mathbb{R}^{|U| \times k_U}, \hat{T} \in \mathbb{R}^{|T| \times k_T}, \hat{R} \in \mathbb{R}^{|R| \times k_R}$ are low-rank feature matrices representing an entity, i.e., user, resources, and tags resp., in terms of its small number of latent dimensions k_U, k_R, k_T, and $\hat{C} \in \mathbb{R}^{k_U \times k_R \times k_T}$ is a tensor representing interactions between the latent factors called *core tensor*. The symbol $\times_i$ denotes the i-mode multiplication between a tensor and a matrix. The model parameters to be learned are then the quadruple $\hat{\theta} := (\hat{C}, \hat{U}, \hat{R}, \hat{T})$. This decomposition refers to a general factorization model known as Tucker decomposition [18]. After the parameters are learned, predictions can be done as follows:

$$\hat{a}_{u,r,t} = \sum_{\tilde{u}} \sum_{\tilde{r}} \sum_{\tilde{t}} \hat{c}_{\tilde{u},\tilde{r},\tilde{t}} \cdot \hat{u}_{u,\tilde{u}} \cdot \hat{r}_{r,\tilde{r}} \cdot \hat{t}_{t,\tilde{t}} \tag{11}$$

where indices over the feature dimension of a feature matrix are marked with a tilde, and elements of a feature matrix are marked with a hat (e.g. $\hat{t}_{t,\tilde{t}}$).

Symeonidis et al. [34] proposed to factorize $\mathcal{A}$, using the 0/1 interpretation scheme (see left-hand side of Figure 1), through higher order SVD (HOSVD), which is the multi-dimensional analog of SVD for tensors; see [18] for a recent survey. The basic idea is to minimize an element-wise loss on the elements of $\hat{\mathcal{A}}$ by optimizing the square loss, i.e.,

$$\underset{\hat{\theta}}{\operatorname{argmin}} \sum_{(u,t,r) \in U \times T \times R} (\hat{a}_{u,t,r} - a_{u,t,r})^2$$

Rendle *et al.* [26], on the other hand, propose RTF (Ranking with Tensor Factorization), a method for learning an optimal factorization of $\mathcal{A}$ for the specific problem of tag recommendations. First, the observed tag assignments are divided in positive, negative, and missing values as described in Section 2.1 (see right-hand side of Figure 1). Let $P_{\mathcal{A}} := \{(u,r)|\exists t \in T : (u,t,r) \in Y\}$ be the set of all distinct user/resource combinations in Y, the sets of positive and negative tags of a particular $(u,r) \in P_{\mathcal{A}}$ are then defined as:

$$T^+_{u,r} := \{t \,|\, (u,r) \in P_{\mathcal{A}} \wedge (u,t,r) \in Y\}$$
$$T^-_{u,r} := \{t \,|\, (u,r) \in P_{\mathcal{A}} \wedge (u,t,r) \notin Y\}$$

From this, pairwise tag ranking constraints can be defined for the values of $\hat{\mathcal{A}}$:

$$a_{u,t_1,r} > a_{u,t_2,r} \Leftrightarrow (u,t_1,r) \in T^+_{u,r} \wedge (u,t_2,r) \in T^-_{u,r} \tag{12}$$

Thus, instead of minimizing the least-squares as in the HOSVD-based methods, an optimization criterion that maximizes the ranking statistic AUC (area under the ROC-curve) is proposed. The AUC measure for a particular $(u,r) \in P_{\mathcal{A}}$ is defined as:

$$\mathrm{AUC}(\hat{\theta},u,r) := \frac{1}{|T^+_{u,r}||T^-_{u,r}|} \sum_{t^+ \in T^+_{u,r}} \sum_{t^- \in T^-_{u,r}} H_{0.5}(\hat{a}_{u,t^+,r} - \hat{a}_{u,t^-,r}) \tag{13}$$

where H_α is the Heaviside function:

$$H_\alpha := \begin{cases} 0, & x < 0 \\ \alpha, & x = 0 \\ 1, & x > 0 \end{cases} \tag{14}$$

The overall optimization task with respect to the ranking statistic AUC and the observed data is then:

$$\underset{\hat{\theta}}{\mathrm{argmax}} \sum_{(u,r) \in P_{\mathcal{A}}} \mathrm{AUC}(\hat{\theta},u,r) \tag{15}$$

The optimization problem is then solved by means of gradient descent [26]. Note that with this optimization criterion missing values are also taken into account since the maximization is only done on the observed tag assignments.

Multi-mode Recommendations Once $\hat{\mathcal{A}}$ is computed, the list with the n highest scoring tags for a given user u and a given resource r can be calculated by:

$$\mathrm{Top}(u,r,n) := \underset{t \in T}{\overset{n}{\mathrm{argmax}}}\, \hat{a}_{u,t,r} \tag{16}$$

Recommending n resources/users to a given user u for a particular t can be done in a similar manner (see [35]). Thus, tensor modeling may allow multi-mode recommendations in an easy way. However, for the RTF method described above, in which the factorization is learned by solving a specific tag ranking optimization problem, it might be necessary to define a specific optimization function for each mode of interest.

Remarks on Complexity A major benefit of a factorization model like RTF or HOSVD is that after a model is built, predictions only depend on the smaller factorization dimensions. HOSVD can be performed efficiently following the approach of Sun and Kolda [19]. Other approaches to improve the scalability to large data sets is through slicing [37] or approximation [4].

4.2.2 FolkRank

The web search algorithm PageRank [2] reflects the idea that a web page is important if there are many pages linking to it, and if those pages are important themselves.[12] In [12], Hotho et al. employed the same underlying principle for Google-like search and ranking in folksonomies. The key idea of the FolkRank algorithm is that a resource which is tagged with important tags by important users becomes important itself. The same holds, symmetrically, for tags and users. We have thus a graph of vertices which are mutually reinforcing each other by spreading their weights. In this section we briefly recall the principles of the FolkRank algorithm, and explain how it can be used for generating tag recommendations.

Because of the different nature of folksonomies compared to the web graph (undirected triadic hyperedges instead of directed binary edges), PageRank cannot be applied directly on folksonomies. In order to employ a weight-spreading ranking scheme on folksonomies, we overcome this problem in two steps. First, we transform the hypergraph into an undirected graph. Then we apply a differential ranking approach that deals with the skewed structure of the network and the undirectedness of folksonomies, and which allows for topic-specific rankings.

Folksonomy-Adapted PageRank First we convert the folksonomy $\mathbb{F} = (U, T, R, Y)$ into an *un*directed tri-partite graph $G_{\mathbb{F}} = (V, E)$. The set V of nodes of the graph consists of the disjoint union of the sets of tags, users and resources (i. e., $V = U \dot{\cup} T \dot{\cup} R$). All co-occurrences of tags and users, users and resources, tags and resources become edges between the respective nodes. I. e., each triple (u, t, r) in Y gives rise to the three undirected edges $\{u, t\}$, $\{u, r\}$, and $\{t, r\}$ in E.

Like PageRank, we employ the random surfer model, that is based on the idea that an idealized random web surfer normally follows links (e. g., from a resource page to a tag or a user page), but from time to time jumps to a new node without following a link. This results in the following definition.

[12] This idea was extended in a similar fashion to bipartite subgraphs of the web in HITS [17] and to n-ary directed graphs in [41].

The rank of the vertices of the graph is computed (like in PageRank) with the weight spreading computation

$$\mathbf{w}_{t+1} \leftarrow dA^{\mathrm{T}}\mathbf{w}_t + (1-d)\mathbf{p} \ , \tag{17}$$

where $\mathbf{w}$ is a weight vector with one entry for each node in V, A is the row-stochastic version of the adjacency matrix[13] of the graph $G_{\mathbb{F}}$ defined above, $\mathbf{p}$ is the random surfer vector – which we use as preference vector in our setting, and $d \in [0,1]$ is determining the strength of the influence of $\mathbf{p}$. By normalization of the vector $\mathbf{p}$, we enforce the equality $||\mathbf{w}||_1 = ||\mathbf{p}||_1$. This[14] ensures that the weight in the system will remain constant. The rank of each node is its value in the limit $\mathbf{w} := \lim_{t\to\infty} \mathbf{w}_t$ of the iteration process.

For a global ranking, one will choose $\mathbf{p} = \mathbf{1}$, i. e., the vector composed by 1's. In order to generate recommendations, however, $\mathbf{p}$ can be tuned by giving a higher weight to the user node and to the resource node for which one currently wants to generate a recommendation. The recommendation $\hat{T}(u,r)$ is then the set of the top n nodes in the ranking, restricted to tags.

As the graph $G_{\mathbb{F}}$ is undirected, most of the weight that went through an edge at moment t will flow back at $t+1$. The results are thus rather similar (but not identical, due to the random surfer) to a ranking that is simply based on edge degrees. In the experiments presented below, we will see that this version performs reasonable, but not exceptional. This is in line with our observation in [12] which showed that the topic-specific rankings are biased by the global graph structure. As a consequence, we developed in [12] the following differential approach.

FolkRank – Topic-Specific Ranking The undirectedness of graph $G_{\mathbb{F}}$ makes it very difficult for other nodes than those with high edge degree to become highly ranked, no matter what the preference vector is. This problem is solved by the *differential* approach in FolkRank, which computes a topic-specific ranking of the elements in a folksonomy. In our case, the topic is determined by the user/resource pair (u,r) for which we intend to compute the tag recommendation.

1. Let $\mathbf{w}^{(0)}$ be the fixed point from Equation (17) with $\mathbf{p} = \mathbf{1}$.
2. Let $\mathbf{w}^{(1)}$ be the fixed point from Equation (17) with $\mathbf{p} = \mathbf{1}$, but $\mathbf{p}[u] = 1 + |U|$ and $\mathbf{p}[r] = 1 + |R|$.
3. $\mathbf{w} := \mathbf{w}^{(1)} - \mathbf{w}^{(0)}$ is the final weight vector.

Thus, we compute the winners and losers of the mutual reinforcement of nodes when a user/resource pair is given, compared to the baseline without a preference vector. We call the resulting weight $\mathbf{w}[x]$ of an element x of the folksonomy the *FolkRank* of x.[15]

[13] $a_{ij} := \frac{1}{\mathrm{degree}(i)}$ if $\{i,j\} \in E$ and 0 else

[14] ...together with the condition that there are no rank sinks – which holds trivially in the undirected graph $G_{\mathbb{F}}$.

[15] In [12] we showed that $\mathbf{w}$ provides indeed valuable results on a large-scale real-world dataset while $\mathbf{w}^{(1)}$ provides an unstructured mix of topic-relevant elements with elements having high edge degree. In [13], we applied this approach for detecting trends over time in folksonomies.

Multi-mode Recommendations For generating tag recommendations for a given user/resource pair (u, r), we compute the ranking as described and then restrict the result set $\hat{T}(u, r)$ to the top n tag nodes. Similarly, one can compute recommendations for users (or resources) by giving preference to a certain user (or resource). Since FolkRank computes a ranking on all three dimensions of the folksonomy, this produces the most relevant tags, users, and resources for the given user (or resource).

Remarks on Complexity One iteration of the adapted PageRank requires the computation of $dA\mathbf{w} + (d-1)\mathbf{p}$, with $A \in \mathbb{R}^{s \times s}$ where $s := |U| + |T| + |R|$. If t marks the number of iterations, the complexity would therefore be $(s^2 + s)t \in \mathcal{O}(s^2 t)$. However, since A is sparse, it is more efficient to go linearly over all *tag assignments* in Y to compute the product $A\mathbf{w}$. After rank computation we have to sort the weights of the tags to collect the top n tags.

4.3 Content-Based Social Tagging RS

All the algorithms described so far do not exploit the content of resources, and hence can be applied to any folksonomy regardless the type of resource supported. Nonetheless, the content of resources is a valuable source of information, specially in cold-start scenarios where there is scarcity of explicit user feedback. In the following, we shortly discuss recommenders that make explicit use of resources' content.

4.3.1 Text-Based

Song et al. [31] proposed an approach based on graph clustering for tagging textual resources, like web pages or other kinds of documents. It does not perform personalized recommendations, as it does not examine users individually. In particular, it considers the relationship among documents, tags, and words contained in resources. These relationships are represented in two bipartite graphs. The approach is divided in two stages:

- In the offline stage, it efficiently performs low rank approximation for the weighted adjacency matrix of the two bipartite graphs, using the Lanczos algorithm [8] for symmetrically partitioning the graphs into multi-class clusters. Moreover, a novel node ranking scheme is proposed to rank the nodes corresponding to tags within each cluster. Next, it applies a Poisson mixture model to learn the document distributions for each class.
- In the online stage, given a document vector, based on the joint probabilities of the tags and the document, tags are recommended for this document based on their within-cluster ranking.

As explained in [31], this two-stage framework can be interpreted as an unsupervised-supervised learning procedure. During the offline stage, nodes are partitioned into clusters (unsupervised learning) and cluster labels are assigned to document nodes,

acting as "class" labels. Moreover, tag nodes are given ranks in each cluster. A mixture model is then built based on the distribution of document and word nodes. In the online stage, a document is classified (supervised learning) into predefined clusters acquired in the first stage by naive Bayes, so that tags can be recommended in the descending orders of their ranks.

Song et al. [31] emphasize the efficiency of the approach, which is guaranteed by the Poisson mixture modeling that allows recommendations in linear-time. Experimental results with two large data sets crawled from CiteULike (9,623 papers and 6,527 tags) and Delicious (22,656 URLs and 28,457 tags) show that recommendations can be provided within one second.

Different content-based methods to suggest tags, given a resource, have also been investigated recently by Illig et al. [14].

4.3.2 Image-Based

Abbasi et al. [1] proposed to use tags as high level features, along with low level image features to train an image classifier on Flickr. Even though this method is not directly applied for RS, it gives some interesting insights about how one can combine tags with low level image features, which could eventually serve as input for a RS. The idea is to first create a vector space from tagging information of images, and then a low level feature space of images using the wavelet transform. These two feature spaces are then joined and used to train a One Class Support Vector Machine (SVM) classifier on the combined feature space.

For creating a feature vector of images based on its tags, a bag-of-words model is used to represent tags as features. Then, the feature vectors are normalized using Term Frequency normalization. Note that this gives low weights to tags in images having a lot of tags. The tag feature vector is then represented as

$$\mathbf{f_t} := \big(tf(t_1, r), tf(t_2, r), ..., tf(t_{|T|}, r)\big)^{\mathrm{T}},$$

where $tf(t, r)$ represents the normalized term frequency value of tag t co-occurring with resource r.

RGB colors are used for the low level feature extraction. Each image r is represented as a four-dimensional vector

$$\mathbf{f_r} := (c_{1,r}, c_{2,r}, c_{3,r}, c_{4,r})^{\mathrm{T}},$$

where the first component $c_{1,r}$ is the mean pixel value in the image r and the remaining components represent the red, green, and blue channel respectively.

The feature vectors are then combined, i.e.,

$$\mathbf{f_{t,r}} := \big(tf(t_1, r), tf(t_2, r), tf(t_{|T|}, r), c_{1,r}, c_{2,r}, c_{3,r}, c_{4,r}\big)^{\mathrm{T}}.$$

This combined feature vectors are then used as input for training a One Class SVM classifier. Experiments were done in real data collected for Flickr and it was

shown that the classifier trained with the combined feature vectors performed considerably better than if trained only with the tag feature vectors or low level image feature vectors alone.

4.3.3 Audio-Based

Eck et al. [5] proposed a method for predicting social tags directly from MP3 files. These tags are called automatic tags (shortly, autotags), because they are directly generated from the musical content. Autotags help in the case where there exist several songs in a collection that are either untagged or poorly tagged. Thus, autotags help to address the "cold-start problem" in music RS. Nevertheless, autotags can be used to smooth the tag space by providing a set of comparable baseline tags for all tracks in a music RS.

Autotags are generated with a machine learning model which uses the meta-learning algorithm AdaBoost to predict tags from audio features, such as: Mel-Frequency Cepstral Coefficients (MFCCs), auto-correlation coefficients computed for selected tags inside songs, and spectrogram coefficients sampled by constant-Q frequency. The audio features are calculated over short sliding windows within the song, leading to an overwhelming total number of features. To reduce this number, "aggregated" features were created by computing individual means and standard deviations (i.e., independent Gaussians) of the previous audio features over short (e.g., 5 seconds) windows of feature data.

Feature selection is performed as follows. The model selects features based on a features ability to minimize empirical error. Therefore, it discards features when weak learners associated with those features are being selected too late by AdaBoost.

Due to the high skewness in the frequency of tags, the prediction task is treated as a classification problem. For each tag, prediction is about whether a particular artist has "none", "some" or "a lot" of a particular tag relative to other tags. The quantification of the boundaries between the 3 classes is done by summing the normalized tag counts of all artists, generating a 100-bin histogram for each tag and moving the category boundaries such that an equal number of artists fall into each of the categories.

To generate autotags, the classes have to be transformed into a bag of words to be associated with an artist. Based on the semantics of the 3 classes, this is done by subtracting the value of the "none" bin from the value of the "a lot" bin, because "none" is the opposite of "a lot" (thus the "some" class serves just to make the classifier more selective in predicting "none" and "a lot").

Experimental evaluation of autotag generation was done with 89,924 songs for 1,277 artists, which resulted in more than 1 million 5 seconds aggregated features. Focus was given on the 60 most popular tags from the social online radio station Last.fm. These tags included genres such as "Rock", and tags related to mood like "chillout". The classification errors where significantly lower than the random errors. As described by the authors [5], performance should be compared against other

classifiers, like SVM or neural networks, in order to better assess the merit of the approach.

4.4 Evaluation Protocols and Metrics

In the following we present some of the protocols and metrics used for the evaluation of the different recommendation modes. For a more comprehensive survey on evaluating recommender systems refer to Chapter **??**.

Resource Recommendations For evaluating tag-aware resource recommenders, the usual protocols and metrics used for traditional RS can be directly used [9] (c.f. Chapter **??**).

Tag Recommendations For evaluating tag recommenders, there are two possible scenarios that can eventually lead to two different evaluation protocols:

- 'New post': A user selects a resource that he has not tagged yet and the system tries to suggest a personalized list of n tags for this resource to the user. This protocol was first used in [24, 16] and was called *LeaveOnePostOut* protocol. For each user u, a separate training set is generated by removing all tags of one of his resources r_u, which has been selected randomly. The training set is then the folksonomy (U, T, R, Y') with $Y' := Y \setminus (\{u\} \times \mathrm{tags}(u, r_u) \times \{r_u\})$ where $\mathrm{tags}(u, r_u)$ is the set of tags used by user u for r_u. The task is then to generate a prediction that is close to $\mathrm{tags}(u, r_u)$. This reproduces the scenario in which a user has an untagged resource for which the system tries to generate useful recommendations.
- 'Tagging refinement': A user selects a resource that he has already tagged in the past and the system suggests a personalized list of n additional tags that the user might want to use for improving his tagging for the resource. This protocol was first introduced in [34]. The idea is to split the tag assigments into past (training set) and future tag assigments (test set). This reflects the scenario where users gradually tag items and receive recommendations before they provide all their tags.

For both protocols the usual precision, recall and f1-measure metrics are commonly used [15, 34, 11, 26]:

$$\text{Recall}\left(\hat{T}(u,r_u)\right) = \frac{|\,\text{tags}(u,r_u) \cap \hat{T}(u,r_u)|}{|\,\text{tags}(u,r_u)|} \tag{18}$$

$$\text{Precision}\left(\hat{T}(u,r_u)\right) = \frac{|\,\text{tags}(u,r_u) \cap \hat{T}(u,r_u)|}{|\hat{T}(u,r_u)|} \tag{19}$$

$$\text{F1-measure}\left(\hat{T}(u,r_u)\right) = \frac{2 \cdot \text{Recall}\left(\hat{T}(u,r_u)\right) \cdot \text{Precision}\left(\hat{T}(u,r_u)\right)}{\text{Recall}\left(\hat{T}(u,r_u)\right) + \text{Precision}\left(\hat{T}(u,r_u)\right)} \tag{20}$$

Furthermore, both scenarios can be applied in an online setting, where the recommendations are computed in real time and shown to the user during annotation of a resource.[16] One can then record if the user clicked on one of the recommended tags or otherwise used the recommendation (e.g., with autocompletion mechanisms). This setting is probably the most realistic one and gives a good measure on how the user liked the recommendation. However, it is pretty laborious to set up and needs a system with active users. There are also some users which don't click on recommendations or use autocompletion which whould affect this evaluation.

User Recommendations For the task of user recommendation, the system suggests to the target user a personalized list of n users, which form his neighborhood. Some systems, like Last.fm, provide information about links between users (in Last.fm socially connected users are called *neighbors*). If such information is available, then it can serve as a ground truth for the evaluation of user recommendation. In cases where such ground truth is not available (like in Bibsonomy), the evaluation of user recommendation can be performed along the lines of evaluating user communities, most notably by calculating the item similarities within them as described in [22]. In particular, in a social bookmarking system (like BibSonomy), for each web resource, its first (home) page can be crawled and preprocessed to create a vector of terms. This way, between any two web resources, their cosine similarity can be computed as follows. For each test user's neighborhood (i.e., most similar users), the Average Cosine Similarity (ACS) of all web resource pairs inside the neighborhood can be computed. ACS corresponds to the intra-neighborhood similarity. Moreover, from a selected number of random neighborhood pairs among all test users' neighborhoods, the average pairwise web resource similarity between every two neighborhoods can be computed. This measure corresponds to the inter-neighborhood similarity. Therefore, the quality of recommended user-neighborhood is judged according both to its intra-neighborhood similarity (the higher the better) and to its inter-neighborhood-similarity (the lower the better).

[16] This was one of the tasks of the ECML PKDD Discovery Challenge 2009 – see http://www.kde.cs.uni-kassel.de/ws/dc09/.

5 Comparison of Algorithms

In this section, we briefly discuss the main advantages and disadvantages of the algorithms presented in Section 4. Note that we just consider the non content-based algorithms since they can be compared under a common basis.

We saw in Section 4.1 that in order to apply standard CF-based algorithms to folksonomies, some data transformation must be performed. Such transformations lead to information loss, which can lower the recommendation quality. Another well known problem with CF-based methods is that large projection matrices must be kept in memory, which can be time/space consuming thus compromising the ability to perform real-time recommendations. Another problem is that for each different mode to be recommended, the algorithm must be eventually changed, demanding an additional effort for offering multi-mode recommendations.

FolkRank builds on PageRank and proved to give significantly better tag recommendations than CF [15]. This method also allows for mode switching with no change in the algorithm. Moreover, as well as CF-based algorithms, FolkRank is robust against online updates since it does not need to be trained every time a new user, resource or tag enters the system. However, FolkRank is computationally expensive and not trivially scalable, making it more suitable for systems where real-time recommendations is not a requirement.

Similarly to FolkRank, tensor factorization methods work directly on the ternary relations of folksonomies. Although the learning phase can be costly, it can be performed offline. After the model is learned, the recommendations can be done fast, making these algorithms suitable for real-time recommendations. A potential disadvantage of tensor factorization methods is that easy mode switching can only be achieved if one consider that the different recommendation problems, i.e., user/resource/tag, can be addressed by minimizing the same error function. If one chooses HOSVD, for example, the model can be used for multi-mode recommendations with trivial mode switching, but at the cost of evtl. solving the wrong problem: HOSVD minimizes a least-square error function while social tagging RS are more related to ranking. If one tries to optimally reconstruct the tensor w.r.t. an error function targeted to a specific recommendation mode, on the other hand, accuracy is eventually improved for the targetted mode, but at the cost of making mode switching more involved. Figure 9 shows a comparison between some of the aforementioned algorithms in a snapshot of the BibSonomy dataset for the tag recommendation problem [26]. Note that the best method is RTF followed by FolkRank and HOSVD.

Table 2 summarizes this discussion. Note that the absence of a “X” in Table 2 indicates that the corresponding property is not trivially achieved by the algorithm being considered.

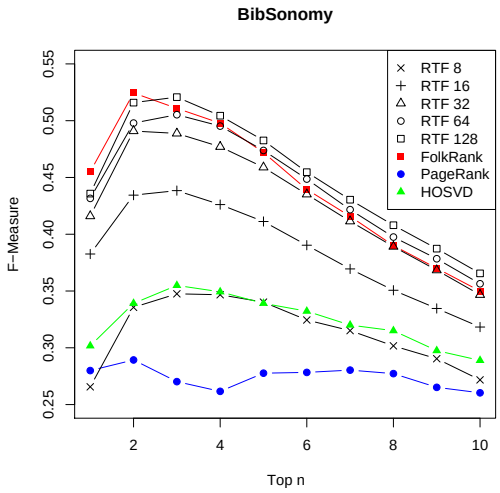


Fig. 9 F-Scores for Top-1, Top-2 to Top-10 lists on a snapshot of the BibSonomy dataset. FolkRank, PageRank and HOSVD are compared to RTF with an increasing number of dimensions under the *LeaveOnePostOut* protocol [26].

Table 2 Summary of advantages and disadvantages of the presented algorithms.

Method	Scalable	Multi-mode recommendation	Keeps Ternary Relation	Online Update
CF-based				X
Folkrank		X	X	X
HOSVD	X	X	X	
RTF	X		X	

6 Conclusions and Research Directions

The Web 2.0 represents a shift of paradigm from the Web-as-information-source to a Web-as-participation-platform where users can upload content, exercise control over that content, and therefore add value to the application as they use it. Among the most prominent family of Web 2.0 applications are the Social Tagging Systems, which promote the sharing of user's tags/resources by exposing them to other users. Due to the increasing popularity of these systems, issues like information overload rapidly become a problem. RS proved to be well suited for this kind of problem in the past and are thus a prominent solution for tackling the information overload in the Web's next generation. In this chapter we presented:

- The data structures of folksonomies, stressing the differences in comparison to the ones used by traditional RS.
- The different modes that can be recommended in STS.
- RS deployed in real STS, stressing the challenges and requirements for doing so.
- Different ways of acquiring tags and how they can affect recommender algorithms.
- Algorithms that:

  - reduce the data dimensionality in order to apply standard CF algorithms,
  - operate directly on the ternary relational data of folksonomies,
  - exploit the content of resources.
- Evaluation protocols and metrics.
- Comparison of the algorithms in terms of pros and cons.

Although the methods that transform the original folksonomy data allow the direct application of standard CF-based algorithms, the transformation inevitably cause some loss of information, which can lower the overall recommendation quality. Some methods try to overcome this problem by doing some sort of ensemble over the different data projections resulting from the transformation, which adds additional free parameters to the problem in order to control the influence of each component. A more natural solution is to operate over the original ternary relation of a folksonomy, which requires the development of new RS algorithms such as FolkRank, that explores the folksonomy hypergraph, or the ones based on tensor factorization. Although FolkRank is known for its high predictive quality, it suffers from scalability problems, and so an interesting research direction is to investigate ways of making it scale.

Tensor factorization for social tagging RS is a recent and prominent field. The research work on this topic has just started to uncover the benefits that those methods have to offer. A particularly appealing research direction concerns investigating tensor factorization models that feature both high recommendation accuracy and easy mode switching.

As pointed out before, folksonomies usually do not contain numerical ratings, but recently the GroupLens[17] research group released a folksonomy dataset in which numerical ratings for the tagged resources are also given.[18] This represents several research opportunities on how to exploit the resource's rating information in order to improve recommendations. In this case, a single data structure for all the modes, such as tensors or hypergraphs, would evtl. fail since the ratings are only related to user-resource pairs and not to tags. Similar issues can be investigated for content-based methods. We saw that content-based methods usually disregard the user information, but past research shows that hybrid methods that combine user preferences with resource's content usually lead to better recommenders. Here, again, tensor or hypergraph representations would evtl. fail since resources' content are only related to the resources but not to the users or tags. So hybrid-based methods that perform some sort of fusion between folksonomy representations and resources' content would be a valuable contribution to the area.

Other topics that were not covered in this chapter, but are nevertheless interesting research directions, concern, for example, recommendations' novelty and serendipity [43], i.e., tags, users and/or resources that are potentially interesting but not obvious; modeling social wisdom for recommendations [39, 28], i.e., explicit friendship strength and mutual trust relations among users, that are evtl. orthogonal to similar-

[17] http://www.grouplens.org/

[18] This dataset can be downloaded at http://www.grouplens.org/node/73

ities of interests and behavior, can be modeled and used to improve the quality of RS.

References

1. Rabeeh Abbasi, Marcin Grzegorzek, and Steffen Staab. Using colors as tags in folksonomies to improve image classification. In *SAMT '08: Poster at Semantics And digital Media Technologies*, 2008.
2. Sergey Brin and Lawrence Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, April 1998.
3. Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, 2004.
4. P. Drineas and M. W. Mahoney. A randomized algorithm for a tensor-based generalization of the svd. *Linear Algebra and Its Applications*, 420(2–3):553–571.
5. Douglas Eck, Paul Lamere, Thierry Bertin-Mahieux, and Stephen Green. Automatic generation of social tags for music recommendation. In *NIPS'07: Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems*, volume 20, 2007.
6. Claudiu S. Firan, Wolfgang Nejdl, and Raluca Paiu. The benefit of using tag-based profiles. In *LA-WEB '07: Proceedings of the 2007 Latin American Web Conference*, pages 32–41, Washington, DC, USA, 2007. IEEE Computer Society.
7. Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin – Heidelberg, 1999.
8. Gene H. Golub and Charles F. Van Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.
9. Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.
10. Thomas Hofmann. Probabilistic latent semantic indexing. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57, New York, NY, USA, 1999. ACM.
11. Andreas Hotho, Dominik Benz, Robert Jäschke, and Beate Krause, editors. *ECML PKDD Discovery Challenge 2008 (RSDC'08)*. Workshop at 18th Europ. Conf. on Machine Learning (ECML'08) / 11th Europ. Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD'08), 2008.
12. Andreas Hotho, Robert Jäschke, Christoph Schmitz, and Gerd Stumme. Information retrieval in folksonomies: Search and ranking. In York Sure and John Domingue, editors, *The Semantic Web: Research and Applications*, volume 4011 of *Lecture Notes in Computer Science*, pages 411–426, Heidelberg, June 2006. Springer.
13. Andreas Hotho, Robert Jäschke, Christoph Schmitz, and Gerd Stumme. Trend detection in folksonomies. In *SAMT '06: Proceedings of the first International Conference on Semantics And Digital Media Technology*, volume 4306 of *LNCS*, pages 56–70, Heidelberg, Dec 2006. Springer.
14. Jens Illig, Andreas Hotho, Robert Jäschke, and Gerd Stumme. A comparison of content-based tag recommendations in folksonomy systems. In *KPP '07: Postproceedings of the International Conference on Knowledge Processing in Practice*, 2009 (to appear).
15. Robert Jäschke, Leandro Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. Tag recommendations in social bookmarking systems. *AI Communications*, pages 231–247, 2008.
16. Robert Jäschke, Leandro Balby Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. Tag recommendations in folksonomies. In *PKDD '07: Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 4702 of *Lecture Notes in Computer Science*, pages 506–514, Berlin, Heidelberg, 2007. Springer.

17. Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
18. Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, September 2009.
19. Tamara G. Kolda and Jimeng Sun. Scalable tensor decompositions for multi-aspect data mining. In *ICDM '08: Proceedings of the 8th IEEE International Conference on Data Mining*, pages 363–372, December 2008.
20. Leslie Lamport. *LaTeX: A Document Preparation System*. Addison-Wesley, 1986.
21. F. Lehmann and R. Wille. A triadic approach to formal concept analysis. In G. Ellis, R. Levinson, W. Rich, and J. F. Sowa, editors, *Conceptual Structures: Applications, Implementation and Theory*, volume 954 of *Lecture Notes in Computer Science*, pages 32–43. Springer, 1995.
22. Xin Li, Lei Guo, and Yihong Eric Zhao. Tag-based social interest discovery. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 675–684, New York, NY, USA, 2008. ACM.
23. M. Srikanth M. Tatu and T. D'Silva. Tag recommendations using bookmark content. In *ECML/PKDD '08: Proceedings of the ECML PKDD Discovery Challenge at 18th Europ. Conf. on Machine Learning / 11th Europ. Conf. on Principles and Practice of Knowledge Discovery in Databases*, 2008.
24. Leandro Balby Marinho and Lars Schmidt-Thieme. Collaborative tag recommendations. In *GFKL '07: Proceedings of the 31st Annual Conference of the Gesellschaft für Klassifikation (GfKl), Freiburg*, pages 533–540. Springer, 2007.
25. Oren Patashnik. BibTeXing, 1988. (Included in the BIBTEX distribution).
26. Steffen Rendle, Leandro B. Marinho, Alexandros Nanopoulos, and Lars S. Thieme. Learning optimal ranking with tensor factorization for tag recommendation. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 727–736. ACM, 2009.
27. Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, USA, 1994. ACM.
28. Ralf Schenkel, Tom Crecelius, Mouna Kacimi, Thomas Neumann, Josiane Parreira, Marc Spaniol, and Gerhard Weikum. Social wisdom for search and recommendation, June 2008. Accepted for publication.
29. Shilad Sen, Jesse Vig, and John Riedl. Tagommenders: connecting users to items through tags. In *WWW '09: Proceedings of the 18th international conference on World Wide Web*, pages 671–680, New York, NY, USA, 2009. ACM.
30. Andriy Shepitsen, Jonathan Gemmell, Bamshad Mobasher, and Robin Burke. Personalized recommendation in social tagging systems using hierarchical clustering. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*, pages 259–266, New York, NY, USA, 2008. ACM.
31. Yang Song, Ziming Zhuang, Huajing Li, Qiankun Zhao, Jia Li, Wang-Chien Lee, and C. Lee Giles. Real-time automatic tag recommendation. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 515–522, New York, NY, USA, 2008. ACM.
32. Sanjay Sood, Sara Owsley, Kristian Hammond, and Larry Birnbaum. Tagassist: Automatic tag suggestion for blog posts. In *ICWSM '07: Proceedings of the International Conference on Weblogs and Social Media*, 2007.
33. Gerd Stumme. A finite state model for on-line analytical processing in triadic contexts. In *ICFCA*, pages 315–328, 2005.
34. Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos. Tag recommendations based on tensor dimensionality reduction. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*, pages 43–50, New York, NY, USA, 2008. ACM.
35. Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos. A unified framework for providing recommendations in social tagging systems based on ternary semantic analysis. *IEEE Transactions on Knowledge and Data Engineering*, 22(2), 2010.

36. Karen H. L. Tso-Sutter, Leandro Balby Marinho, and Lars Schmidt-Thieme. Tag-aware recommender systems by fusion of collaborative filtering algorithms. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, pages 1995–1999, New York, NY, USA, 2008. ACM.
37. P. Turney. Empirical evaluation of four tensor decomposition algorithms. *Technical Report (NRC/ERB-1152)*, 2007.
38. Luis von Ahn and Laura Dabbish. Designing games with a purpose. *Commun. ACM*, 51(8):58–67, 2008.
39. Frank E. Walter, Stefano Battiston, and Frank Schweitzer. Personalised and dynamic trust in social networks. In *RecSys '09: Proceedings of the 2009 ACM conference on Recommender systems*, New York, NY, USA, 2009. ACM. to appear.
40. Robert Wetzker, Winfried Umbrath, and Alan Said. A hybrid approach to item recommendation in folksonomies. In *ESAIR '09: Proceedings of the WSDM '09 Workshop on Exploiting Semantic Annotations in Information Retrieval*, pages 25–29. ACM, 2009.
41. Wensi Xi, Benyu Zhang, Zheng Chen, Yizhou Lu, Shuicheng Yan, Wei-Ying Ma, and Edward Allan Fox. Link fusion: a unified link analysis framework for multi-type interrelated data objects. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 319–327, New York, NY, USA, 2004. ACM.
42. Yanfei Xu, Liang Zhang, and Wei Liu. Cubic analysis of social bookmarking for personalized recommendation. pages 733–738. 2006.
43. Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *WWW '05: Proceedings of the 14th International Conference on World Wide Web*, pages 22–32, New York, NY, USA, 2005. ACM.