11

Recommender systems and the next-generation web

In recent years, the way we use the web has changed. Today's web surfers are no longer mere consumers of static information or users of web-enabled applications. Instead, they play a far more active role. Today's web users connect via social networks, they willingly publish information about their demographic characteristics and preferences, and they actively provide and annotate resources such as images or videos or share their knowledge in community platforms. This new way of using the web (including some minor technical innovations) is often referred to as *Web 2.0* (O'Reilly 2007).

A further popular idea to improve the web is to transform and enrich the information stored in the web so that machines can easily interpret and process the web content. The central part of this vision (called the Semantic Web) is to provide defined meaning (semantics) for information and web services. The Semantic Web is also vividly advertised, with slogans such as "enabling computers to read the web" or "making the web readable for computers". This demand for semantics stems from the fact that web content is usually designed to be interpreted by humans. However, the processing of this content is extremely difficult for machines, especially if machines must capture the intended semantics. Numerous techniques have been proposed to describe web resources and to relate them by various description methods, such as how to exchange data, how to describe taxonomies, or how to formulate complex relations among resources.

These recent developments also open new opportunities in the area of recommender systems. One of the basic challenges in many domains is the sparsity of the rating databases and the limited availability of user preference information. In Web 2.0 times, these valuable pieces of information are increasingly available through social networks, in which users not only exhibit their preferences to others but are also explicitly connected to possibly like-minded, trusted

users. One research question, therefore, consists of finding ways to exploit this additional knowledge in the recommendation process.

Web 2.0 also brought up new types of public information spaces, such as web logs (blogs), wikis, and platforms for sharing multimedia resources. Because of the broad success and sheer size of many of those public platforms, finding interesting content becomes increasingly challenging for the individual user. Thus, these platforms represent an additional application area for recommender systems technology.

Furthermore, in knowledge-based recommendation the acquisition and maintenance of knowledge is an important precondition for a successful application. Semantic information offered by the Semantic Web allows us to apply knowledge-based recommender techniques more efficiently and to employ these techniques to new application areas. Semantic information about users and items also allows us to improve classical filtering methods.

All these new capabilities of Web 2.0 and the Semantic Web greatly influence the field of recommender systems. Although almost all areas of recommender technologies are touched by the introduction of new web technologies and the changed usage of the web, our goal in this chapter is to provide a central overview about recent developments triggered by these innovations.

For example, we can exploit information provided by online communities to build trust-aware recommender systems to avoid misuse or to improve recommendations (Section 11.1). Furthermore, the (free) annotation of web resources are forming so-called folksonomies (Section 11.2), which give additional valuable information to improve recommendations. If these annotations, which can be considered as additional characterization of items, are interpreted more formally (i.e., by a formal description of a taxonomy), then we receive some additional means to enhance recommender systems, as described in Section 11.3. On one hand, semantics are valuable for computing recommendations, but on the other hand, semantic information must be acquired. In Section 11.4 we discuss approaches as to how this task can be solved more efficiently in the context of recommender systems.

11.1 Trust-aware recommender systems

When we talked about "user communities" – for instance, in the context of collaborative filtering approaches – the assumption was that the community simply consists of all users of the online shop (and its recommender system) and that the members of the community are only implicitly related to each other through their co-rated items. On modern consumer review and price comparison

platforms as well as in online shops, however, users are also given the opportunity to rate the reviews and ratings of other users. On the Epinions.com consumer review platform, for instance, users may not only express that they liked a particular item review, but they can also state that they generally trust specific other users and their opinions. By this means, users are thus becoming explicitly connected in a "web of trust". In the following section, we show how such trust networks can serve as a basis for a recommender system.

The phrase "trust in recommender systems" is interpreted in the research community in three different ways: first, in the sense of getting users to believe that the recommendations made by the system are correct and fair, such as with the help of suitable explanations; second, that recommender systems assess the "trustworthiness" of users to discover and avoid attacks on recommender systems. These system-user and user-system trust relationships were discussed in previous chapters. In this section, we focus on the third interpretation of trust, which is based on trust relationships between users – users put more trust in the recommendations of those users to which they are connected.

11.1.1 Exploiting explicit trust networks

In general, trust-enhanced nearest-neighbor recommender systems, such as the one presented by Massa and Avesani (2007), aim to exploit the information from trust networks to improve the systems "performance" in different dimensions. The hope is that the accuracy of the recommendations can be increased – for instance, by taking the opinions of explicitly trusted neighbors into account instead of using peers that are chosen based only on a comparison of the rating history. In particular, the goal here is also to alleviate the cold-start problem and improve on the user coverage measure – in other words, if no sufficiently large neighborhood can be determined based on co-rated items, the opinions of trusted friends can serve as a starting point for making predictions. In addition, one conjecture when using explicit trust networks is that they help make the recommender system more robust against attacks, because desired "incoming" trust relationships to a fake profile cannot easily be injected into a recommender database.

Next, we briefly sketch the general architecture of the "trust-aware recommender system" (TARS) proposed by Massa and Avesani (2007) as an example and summarize their findings of an evaluation on real-world datasets.

There are two inputs to the proposed recommender system: a standard rating database and the *trust network*, which can be thought of as a user-to-user matrix T of trust statements. The possible values in the matrix range from 0 (no trust)

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

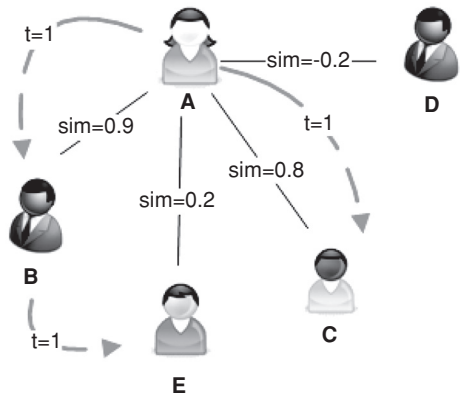

Figure 11.1. Similarity and trust network.

to 1 (full trust). A special null value such as $\perp$ (Massa and Avesani 2004) is used to indicate that no trust statement is available.

In principle, the TARS system aims to embed the trust knowledge in a standard nearest-neighbor collaborative filtering method. Basically, the prediction computation is the same as the usual nearest-neighbor weighting scheme from Equation 11.1, which predicts the rating for a not-yet-seen item p based on the active user's average rating $\overline{r_a}$ and the weighted opinions of the N most similar neighbors.

$$pred(a, p) = \overline{r_a} + \frac{\sum_{b \in N} sim(a, b) * (r_{b,p} - \overline{r_b})}{\sum_{b \in N} sim(a, b)} \tag{11.1}$$

In the TARS approach, the only difference to the original scheme is that the similarity of the active user a and neighbor b, $sim(a, b)$, is not calculated based on Pearson's correlation coefficient (or something similar). Instead, the trust value $T_{a,b}$ is used to determine the weights of the opinions of the neighbors.

Figure 11.1 shows an example of a situation in which the problem is to generate a recommendation for user A (Alice). Both the standard similarity measures (denoted as *sim*) and some trust statements (denoted as t) are shown in the figure: A has issued two explicit trust statements: she trusts users B and C, probably because in the past, she found the opinions of B and C valuable. Assume that in this setting A's actual rating history (similarity of 0.8 and 0.9) is also quite similar to the ones of B and C, respectively.

Furthermore, assume that in this example we parameterize a nearest-neighbor algorithm to make a prediction for A only if at least three peers

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

with a similarity above the 0.5 level can be identified. With these parameters and the given trust network, no recommendation can be made. If we use the trust values instead of Pearson's similarity measure, the situation will not improve, as A has issued only two trust statements. The key point of trust-aware recommendation, however, is that we may assume that the trust-relationships are *transitive* – that is, we assume that if A trusts B and B trusts E, A will also trust E. Although A might not fully trust E, because of the missing direct experience, we may assume that A trusts E at least to some extent. In the example setting shown in Figure 11.1, a third peer, namely E, is therefore assumed trustworthy and a recommendation can be made if we again assume a lower bound of at least three neighbors.

It is relatively easy to see in the example how trust-aware recommenders can help alleviate the cold-start problem. Although the matrix of trust statements may also be sparse, as in a typical rating database, the transitivity of the trust relationship allows us to derive an indirect measure of trust in neighbors not directly known to the target user. In addition, if the trust matrix is very sparse, it can be combined in one way or another with the standard similarity measure and serve as an additional source of knowledge for the prediction.

An important property of trust relationships is that they are usually not assumed to be symmetric – that is, the fact that A trusts the opinion of C does not tell us that C also trusts A. This aspect helps us make trust networks more robust against attacks: although it is easy to inject profiles that express trust both to real users and other fake users, it is intuitively not easy to attract trust statements from regular users – that is, to connect the fake network to the network of the real users, which is important to propagate trust relationships to fake users. To limit the propagation of false trust values, the number of propagating steps can be limited or the trust values multiplied by some damping factor.

11.1.2 Trust metrics and effectiveness

How indirect trust and distrust values should be calculated between users that are not directly connected in a social network is a relatively new area of research. One typical option is to use a multiplicative propagation of trust values in combination with a maximum propagation distance and a minimum trust threshold (Massa and Avesani 2007, Golbeck 2006); these parameters, for instance, can be determined empirically. The algorithm implementations are relatively straightforward (Massa and Avesani 2007) and may differ in the way the graph formed by the social connections is explored.

These relatively simple metrics already lead to some measurable increase in recommendation accuracy in special cases. Still, various other schemes for trust calculation that, for instance, also take into account explicit distrust statements differently than "low trust" have been proposed. A broader discussion of this topic and more complex propagation schemes are discussed, for instance, by Guha et al. (2004), Ziegler and Lausen (2004, 2005), and Victor et al. (2006); proposals based on subjective logic or in the context of the Semantic Web are presented by Jøsang et al. (2006) and Richardson et al. (2003).

Although the aforementioned metrics calculate an estimate of the "local" trust between a source and a target user, it would be possible to use a "global" trust metric as well. A very simple metric would be to average the trust values received for every user to compute an overall user "reputation". Such an approach often can be found, for example, on online auction platforms. Google's PageRank algorithm (Brin and Page 1998) can also be seen as an example of a global trust metric, which, however, goes beyond simple averaging of the number of links pointing to a page.

To what extent the information in the trust networks can really help to improve the quality of recommendations has been evaluated by Massa and Avesani (2004) on the basis of a dataset from the Epinions.com platform. On this web site, users can not only review all types of items (such as cars, books, or movies) but also rate the reviewers themselves by adding them to their personal web of trust if they have the feeling that they "have consistently found [the reviews] to be valuable". Because no other information is available, adding a user to the web of trust can be interpreted as a direct trust rating of 1. On this platform, users may also put other users on a block list, which can be seen as a trust rating of 0. More fine-grained ratings, however, are not possible on Epinions.com.

Interestingly, the item rating behavior of Epinions.com users is reported to be different from that of MovieLens users. Compared to MovieLens, not only are the rating database and the trust matrix even sparser, but about half the ratings had the highest possible value (5) and another 30 percent had the second-highest value (4). Furthermore, more than half the users are cold starters who have voted for fewer than five items. Standard accuracy metrics such as mean absolute error do not reflect reality very well in such settings, because every prediction error will be weighed in the same way, although significant differences for heavy raters and cold starters are expected. Massa and Avesani (2007) therefore propose to use the mean absolute user error (MAUE) to ensure that all users have the same weight in the accuracy calculation. In addition, the usage of a *user coverage* metric is recommended, which measures the number

of users for which a recommendation can be made – a problem that virtually does not appear in the MovieLens dataset, because it is guaranteed that every user has rated at least twenty items.

The results of an evaluation that used different algorithms (standard CF, unpersonalized, and trust-based ones) and views on the dataset to measure the impact on heavy raters, cold starters, niche items, and so forth can be summarized as follows:

- *Effectiveness of simple algorithms.* Given such a specific, but probably in practice not uncommon, distribution of ratings, simple algorithms such as "always predict value 5" or "always predict the mean rating value of a user" work quite well and are only slightly worse than standard CF algorithms with respect to MAE.

  Another simple technique is to predict the average item rating. In this application scenario, this unpersonalized algorithm even outperforms standard CF techniques, an effect that cannot be observed when using other datasets, such as the MovieLens database. With respect to rating coverage, the simple method is better, in particular, when it comes to the many cold-start users. When the different slices of the rating database are analyzed, it can be observed, however, that CF techniques are effective for controversial items for which the standard deviation is relatively large and an averaging technique does not help.
- *Using direct trust only.* In this setting, a trust-based technique is employed that uses only the opinions of users for which an explicit trust statement is available; no trust propagation over the network is done. Although the overall MAE of this method is between that of standard CF and the simple average technique, it works particularly well for cold-start users, niche items (items for which only very few ratings exist), and opinionated users (users having a high standard deviation in their ratings). When the MAUE measure is used, the "direct-trust" method called MT1 achieves the highest overall accuracy of all compared methods. With respect to coverage, it is finally observed that "MT1 is able to predict fewer ratings than CF but the predictions are spread more equally over the users (which can then be at least partially satisfied)" (Massa and Avesani 2007).
- *Trust propagation.* The evaluations with propagation levels of 2, 3, and 4 lead to the following observations: First, when the propagation level is increased, the trust network quickly grows from around ten directly trusted neighbors to nearly 400 in the second level, and several thousand in the third level. An increase in the propagation distance thus leads directly to an increase in rating coverage. At the same time, however, the prediction accuracy constantly

decreases, because the opinions of the faraway neighbors are not the best predictors.

- *Hybrids.* In their final experiments, Massa and Avesani tried to calculate predictions based on both a standard similarity measure and the trust measure and combine the results as follows: when only one method was able to compute a weight, this value was taken. If both methods were applicable, a weighted average was computed. Although such a combination quite intuitively leads to increased coverage, the performance did not increase and typically fall between the CF and the trust-based algorithms.

To determine whether global trust metrics (in which every user gets the same trust value from everyone) work well, the performance of using an adapted version of PageRank to determine global trust weights was also evaluated. The experiments showed, however, that such nonpersonalized metrics cannot compete with the personalized trust metrics described earlier.

In summary, the evaluation showed two major things. First, the exploitation of existing trust relations between users can be very helpful for fighting the cold-start problem. Second, there are still many open questions in the context of the evaluation of recommender systems, most probably because research evaluations are often made only on the MovieLens dataset.

11.1.3 Related approaches and recent developments

The concept of trust in social networks has raised increased interest during the past few years. Among others, the following topics have also been covered in the area of recommender systems.

- *Similar approaches.* Several approaches to exploiting and propagating trust information have been proposed in the last years that are similar to the work by Massa and Avesani (2007). Golbeck (2005) and Golbeck and Hendler (2006), report on an evaluation of the trust-enhanced movie recommender FilmTrust. Although the size of the rating database was very small (only about 500 users), observations could be made that were similar to those with the larger Epinions.com dataset. The distribution of ratings in the database was not very broad, so a "recommend the average" approach generally worked quite well. Again, however, the trust-based method worked best for opinionated users and controversial ratings and outperformed the baseline CF method. The propagation method used in the FilmTrust system is similar to the one used by Massa and Avesani (2007); however, the trust ratings can be chosen from a scale of 1 to 10.

Another approach that exploits explicit trust statements and combines them with another measure of item importance was recently proposed by Hess et al. (2006). In their work, a trust network among scientific reviewers is combined with standard visibility measures for scientific documents to personalize the document visibility measure in a community. Although no real-world evaluation of this approach has been made, it can be counted as another interesting idea that shows how in the future trust information from various sources might be combined to generate personalized information services.

- *Implicit trust.* Although their paper is titled "Trust in Recommender Systems", the trust concept used by O'Donovan and Smyth (2005) is not based on direct trust statements as in the work described above. Instead, they propose new neighbor selection and weighting metrics that go beyond simple partner similarity. The real-life analogy on which they base their work is that one will typically ask friends who have similar overall tastes for a movie recommendation. However, not every one of those "general" neighbors might be a good advisor for every type of item/movie. In their work, therefore, trustworthiness is determined by measuring how often a user has been a reliable predictor in the past – either on average or for a specific item. In contrast to the work of Massa and Avesani, for instance, these trust values can be automatically extracted from the rating database.

  When these numbers are available, predictions can be computed in different ways: by using a weighted mean of the standard Pearson similarity and the trust measure, by using the trust value as a filter for neighbor selection, or by a combination of both. O'Donovan and Smyth evaluated their approach on the MovieLens dataset and showed that an accuracy improvement of more than 20 percent can be achieved when compared with the early CF approach of Resnick et al. Resnick et al. (1994).

  The usage of the term *trust* for this approach is not undisputed. The term *competence* or *reputation* would probably have been more suitable for this approach. Further approaches that automatically infer such "trust" relationships from existing rating data are those by Papagelis et al. (2005) and Weng et al. (2006).
- *Recommending new friends.* Before statements in an explicit trust network can be exploited, new users must be connected with a sufficient number of other members of the community – in other words, we face another form of cold-start problem here. Many of today's social web platforms aim to increase the connectivity of their members by suggesting other users as friends. The suggestions are based, for example, on existing relationships (friend-of-a-friend) or on a random selection.

However, because the number of other users a person will connect to is limited, it might be a good idea for the system to automatically recommend community members whose opinions are particularly valuable – the ones who, in our setting, are good predictors. How such users are characterized and how a good selection influences prediction quality is analyzed by Victor et al. (2008a, 2008b). In their work, the authors differentiate among the following particular user types (key figures): *mavens*, who write a lot of reviews; *frequent raters*, who rate many items; and *connectors*, who issue many trust statements and are trusted by many – that is, when trust is propagated over them, many other users are reached. To measure the impact of the opinion of one user on the other and to analyze the tradeoff between coverage and accuracy, several measures, such as accuracy change or betweenness (from the field of social network analysis; see Wasserman and Faust 1994), are used. Overall, Victor et al. can show – based again on an evaluation of an Epinions.com dataset – that the inclusion of key figures in the network leads to better accuracy and coverage results when compared with a situation in which users are initially connected to a randomly selected set of other users.

In summary, recent research has already shown that the exploitation of existing trust information can improve the accuracy of recommender systems. We believe, however, that this is only the beginning of a more comprehensive exploitation of all the different information available in today's and tomorrow's online social networks.

11.2 Folksonomies and more

Besides social networks such as Facebook.com, on which user communities willingly share many of their preferences and interests, platforms that support *collaborative tagging* of multimedia items have also become popular in Web 2.0. In contrast to classical keyword-assignment schemes that superimpose a defined classification hierarchy, so-called folksonomies (folk taxonomies) represent a far more informal way of allowing users to annotate images or movies with keywords. In these systems, users assign arbitrary tags to the available items; tags can describe several dimensions or aspects of a resource such as content, genre, or other metadata but also personal impressions such as *boring*. When annotating such tags, users frequently express their opinions about products and services. Therefore, it seems to be self-evident to exploit this information to provide recommendations.

The goals of the Semantic Web also include the semantic annotation of resources. In contrast to tagging systems, however, Semantic Web approaches postulate the usage of formal, defined, and machine-processible annotations. Although folksonomies are sometimes referred to as "lightweight ontologies", they are actually at the opposite spectrum of annotation options: although formal ontologies have the advantages of preciseness and definedness, they are hard to acquire. Folksonomies, on the other hand, not only can be used by everyone but also directly reflect the language and terms that are actually used by the community.

Recommender systems and folksonomies can be related to each other in different ways. First, in analogy to trust-enhanced approaches, one can try to exploit the information of how items are tagged by the community for predicting interesting items to a user. Second, because the arbitrary usage of tags also has its problems, recommender system technology can be used to recommend tags to users – for example, to narrow the range of used tags in the system. We will shortly summarize example systems of each type in the following.

11.2.1 Using folksonomies for recommendations

In collaborative tagging systems, users annotate resources with tags. The question in folksonomy-enhanced recommender approaches is how we can leverage this information for recommending items or improving prediction accuracy. Roughly speaking, there are two basic approaches to integrate tags in the recommendation process. Tags can be viewed as information about the content of items. Consequently, standard content-based methods are the starting point of extension. Tags can also be viewed as an additional dimension of the classical user–item matrix, and collaborative techniques serve as a basis for advancements.

In the following subsections, we introduce the basic ideas from both sides, using some recent works that clearly show the underlying considerations.

11.2.1.1 Folksonomies and content-based methods

We start our introduction by the exploitation of so-called tag clouds. Then we present two approaches to deal with the ambiguity and redundancy of free formulated tags. One approach tries to overcome this problem by linguistic methods, whereas the other direction is to exploit statistical information.

Recommendations based on tag clouds. Szomszor et al. (2007) exploits folksonomies for recommendations in the movie domain by combining the information from two data sources: the Netflix rating dataset and the *Internet*

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

Movie Database (IMDb). The Netflix database contains millions of ratings for several thousand movies. The IMDb database is a large collection of movie data and comprises nearly a million titles. In this experiment, only the community-provided keywords (tags) in the IMDb have been used: typically, several dozen keywords are associated with a popular movie. The two databases can be relatively easily combined in a single schema by matching the movie names.

Based on the ratings and keywords associated to films by various users, the recommendation method tries to estimate the future ratings of users. The central concept for tag-based recommendation by Szomszor et al. (2007) is what the authors call the user's *rating tag cloud*. The basic idea is to identify the keywords typically selected to annotate films that the user u has assigned a rating r. For instance, we might identify that users typically assign the keywords "army," "based-on-novel," or "blockbuster" to movies a specific user u usually rates with a rating value 5. Based on this information, we can search for movies not rated by u but annotated with keywords "army," "based-on-novel," or "blockbuster" and guess that these movies will also be rated 5 by the user u.

In particular, let a specific user be denoted by $u \in U$, where U is the set of all users and $m \in M$ is a movie, where M is the set of all available movies. A rating value is denoted by $r \in R$, where R is the set of possible rating values – for instance, $R = \{1, 2, 3, 4, 5\}$. The set of movies rated by user u is M_u. The rating value for a user $u \in U$ and a movie $m \in M_u$ is denoted by $f_u(m) \in R$.

K is the global set of keywords, and K_m is the set of keywords associated with movie m. N_k denotes the global frequency of occurrences of keyword $k \in K$ for all movies.

Based on this information, a rating tag-cloud $T_{u,r}$ is introduced by Szomszor et al. (2007) for a given user u and rating r. $T_{u,r}$ is defined as the set of -tuples $\langle k, n_k(u, r)\rangle$, where $k \in K$ refers to a keyword and $n_k(u, r)$ is the frequency of keyword k assigned to movies, which user u has rated with rating value r:

$$n_k(u, r) = |\{m | m \in M_u \wedge k \in K_m \wedge f_u(m) = r\}| \tag{11.2}$$

In other words, given a user u, a rating value r, and a keyword k, $n_k(u, r)$ gives the number of movies annotated by keyword k that have been assigned a rating r by user u. The tag-cloud $T_{u,r}$ expresses the keywords assigned to movies that have been rated with r by user u and how often these keywords were used (which can be seen as a weight/significance factor).

In general, a tag cloud can serve as means of visually depicting the importance of individual tags: more frequent terms are accentuated by a larger font size (Figure 11.2). Figure 11.2 could, for instance, be the tag cloud of the movies that user A has rated with "5."

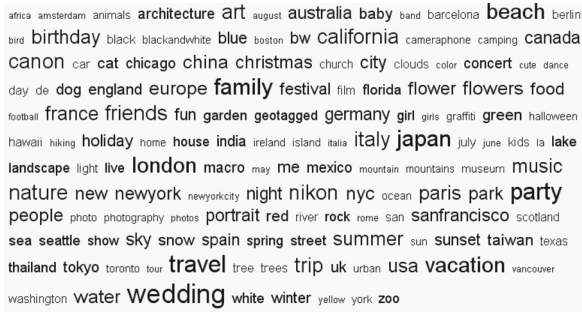

Figure 11.2. Tag cloud.

Szomszor et al. (2007), use the rating tag clouds as the only basis for recommending items according to the following two schemes.

- *Simple tag cloud comparison.* This metric makes a prediction for an unseen movie m^* by comparing the keywords K_{m^*} associated with m^* with each of the active user's u rating tag clouds. In particular, for an unseen movie m^*, the rating r^* is guessed where the intersection of the tags of tag cloud T_{u,r^*} and the tags associated to movie m^* is maximal – that is, r^* is assigned to the r where $\sigma(u, m^*, r) = |\{(k, n_k) \in T_{u,r} | k \in K_{m^*}\}|$ is maximal.
- *Weighted tag cloud comparison.* This slightly more sophisticated metric not only measures keyword overlap, but also takes the weight of the tags in the clouds into account. This is done by defining a similarity measure, which follows the idea of TF-IDF weighting – the relative frequency of term occurrences and global keyword frequencies are considered when comparing a tag cloud and a keyword set.

Given a user u, a movie m^*, and a rating r^*, the appropriateness of r^* is estimated by:

$$\sigma(u, m^*, r^*) = \sum_{\{\langle k, n_k(u,r^*)\rangle \in T_{u,r^*} | k \in K_{m^*}\}} \frac{n_k(u, r^*)}{log(N_k)} \tag{11.3}$$

In this measure, Szomszor et al. (2007) sum the frequency of all keywords K_{m^*} of movie m^* where user u has used these keywords to rate movies with rating value r^*. The frequencies are weighted by dividing them by the logarithm of the global frequency N_k of keyword k, as commonly done in term-weighting schemes. The weighted average for all possible rating values is defined by

$$\overline{\sigma}(u, m^*) = \frac{1}{S(u, m^*)} \sum_{r \in R} r \times \sigma(u, m^*, r) \tag{11.4}$$

where $S(u, m^*) = \sum_{r \in R} \sigma(u, m^*, r)$ is a normalization factor.

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

In the experiments conducted by Szomszor et al. (2007) the estimation of r^* is based on combining $\overline{\sigma}(u, m^*)$ with the average rating of movie m^*. The average rating is simply defined as usual, where U_{m^*} is the set of users who rated m^*:

$$\overline{r}(m^*) = \frac{1}{|U_{m^*}|} \sum_{u \in U_{m^*}} f_u(m^*) \tag{11.5}$$

Finally, the weighted estimated rating value of a movie m^* for users u is computed by

$$\sigma^*(u, m^*) = 0.5\, \overline{r}(m^*) + 0.5\, \overline{\sigma}(u, m^*) \tag{11.6}$$

In the experiments, the accuracy of these two metrics is compared with an unpersonalized prediction method, which always predicts the average value of all ratings an item in question has received. What is shown in these preliminary evaluations is that recommendations can, in principle, be made solely based on the tag clouds and that the weighted approach performs better than the unweighted approach. For the Netflix dataset, a root mean squared error of roughly 0.96 was achieved. The analysis of the experiments showed that the proposed approach does pretty well for average ratings but has potential for improvements for extreme ratings. The work of Szomszor et al. (2007) is a first indication that the rating tag clouds can serve as an additional source of user profile information in a hybrid system.

Linguistic methods for tag-based recommendation. The work of de Gemmis et al. (2008) goes in a similar direction as the previously described approach; however, the main difference is that sophisticated techniques are exploited to identify the intended sense of a tag (keyword) associated with an item and to apply a variant of a naive Bayesian text classifier.

Basically, de Gemmis et al. implemented a content-based recommender system for recommending descriptions about paintings. They assumed that such descriptions are structured in slots – there are slots for the title, the painter, and a general painting description. These slots are called *static slots* because they do not change over time. Compared with plain content-based methods, the exploitation of slots is just an additional feature because the approach is also applicable if the content of slots are merged in just one slot.

The idea of de Gemmis et al. (2008) is to merge tags assigned by users to descriptions in special slots. These slots are called *dynamic slots* because they change as users add tags. In particular, given item I, the set of tags provided by all the users who rated I is called *SocialTags(I)* and set of tags provided by

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

a specific user U for I is called *PersonalTags(U, I)*. For each set of tags, slots are added to the items.

Because tags may be formulated freely by users, the sense of tags can be ambiguous or tags may be a synonym of other tags. This problem must be addressed for content-based recommender systmes in general. To solve this problem, de Gemmis et al. (2008) propose the application of semantic indexing of documents. In particular, words in a slot are replaced by synsets using WORDNET. A *synset* (*syn*onym *set*) is a structure containing sets of words with synonymous meanings, which represents a specific meaning of a word. This addresses the problem of synonyms. The problem of an ambiguous word sense is tackled by analyzing the semantic similarity of words in their context. A detailed description of word sense disambiguation (WSD) is presented by Semeraro et al. (2007). The result of applying WSD to the content of slots is that every slot contains a set of synsets, which are called *semantic tags*.

Following the common practice of content-based recommendation, the user classifies items he or she likes and items he or she dislikes. This classification information is exploited to compute the parameters of a naive Bayesian classifier – the conditional probability $P(c|d_j)$ is computed where c has the values *likes/dislikes* and d_j represents a specific item.

In order to apply Bayes' rule, the specialty of de Gemmis et al. (2008) is to compute the required conditional probability $P(d_j|c)$ by exploiting slots. Let M be the number of slots, s a slot, and d_j^s representing the slot s of document d_j; then according to the naive assumption of conditional independence, $P(d_j|c)$ is computed by

$$P(d_j|c) = \prod_{s=1}^{M} P(d_j^s|c) \tag{11.7}$$

Furthermore, $P(d_j^s|c)$ is estimated by applying a multivariate Poisson model in which the parameters of this model are determined by the average frequencies of tokens in the slots and in the whole collection of items. The method does not distinguish between various types of slots. Hence, static and dynamic slots – the slots representing the social and personal tags – are processed equally.

The evaluation by Gemmis et al. (2008) is based, as usual, on a k-fold cross-validation. Five configurations were explored: items were described (a) only with social tags, (b) with personal tags, (c) with social tags and static slots, (d) with personal tags and static slots, and (e) with static slots, neglecting dynamic slots. The best-performing combinations with respect to an F-measure are static slots combined with social or personal tags. However, it was observed that if only a few training examples are available, social tags without static tags

performed best. The more examples that were available to assess the preferences of an individual, the better was the performance of a combination of static and dynamic slots.

Tag clustering. Although folksonomies provide many opportunities to improve recommendations, the free formulation of tags leads to unique challenges. Unsupervised tagging results in redundant, ambiguous, or very user-specific tags. To overcome this problem, Shepitsen et al. (2008) propose the clustering of tags.

The basic idea is to compute the interest of a user u in a resource r by the following formula:

$$I(u, r) = \sum_{c \in C} ucW(u, c) \times rcW(r, c) \tag{11.8}$$

C is the set of all tag clusters; $ucW(u, c)$ is the user's interest in cluster c, calculated as the ratio of times user u annotated a resource with a tag from that cluster over the total annotations by that user; $rcW(r, c)$ determines the closeness of a resource to a cluster c by the ratio of times the resource was annotated with a tag from the cluster over the total number of times the resource was annotated.

This user interest in a resource is exploited to weight the similarity (denoted by $S(q, r)$) between a user query q and a resource r, where a user query corresponds to a tag. $S(q, r)$ is computed by the cosine similarity using term frequencies. In particular, the tag frequency $tf(t, r)$ for a tag t and a resource r is the number of times the resource has been annotated with the tag. T is the set of tags.

$$S(q, r) = cos(q, r) = \frac{tf(q, r)}{\sqrt{\sum_{t \in T} tf(t, r)^2}} \tag{11.9}$$

This similarity $S(q, r)$ between a query q and a tag t is personalized by the interest of user u in a resource r resulting in similarities $S'(u, q, r)$ relating users, queries, and resources: $S'(u, q, r) = S(q, r) \times I(u, r)$.

For the computation of clusters of tags, tags are represented as a vector of weights over the set of resources. Both TF and TF-IDF can be applied; however, TF-IDF showed better results. The idea of clustering is that similar terms, such as *web design* and *design*, appear in similar clusters. Shepitsen et al. (2008) experimented with various cluster methods, showing in their experiments that agglomerative clustering leads to better improvements. Furthermore, Shepitsen et al. (2008) they propose a query-dependent cluster method. The idea is that given a query term (e.g., baseball), the clustering algorithm starts to build a cluster of tags around the query term.

In their evaluation, Shepitsen et al. (2008) showed the improvements compared with a standard recommendation technique based on cosine similarity exploiting test data from last.fm and del.icio.us; last.fm is a music community web site; del.icio.us is a social bookmarking web service. Using a query-dependent clustering technique showed higher improvements in the del.icio.us test domain compared with last.fm. The authors assume that this comes from the fact that the tags in last.fm have a higher density and are less ambiguous.

Comparison with classical collaborative methods. Sen et al. (2009) compare various algorithms that explore tags for recommendation tasks. This comparison is based on the ratings of MovieLens users. The algorithms evaluated are classified in two groups. *Implicit-only algorithms* do not explore an explicit rating of items, but examine only feedback from the users, such as clicks or searches. For example, if a user clicks on a movie, this could be interpreted as a sign of interest. *Explicit algorithms* exploit the rating of items provided by the user. Implicit-only methods are of special interest for domains in which the users cannot provide ratings.

For the evaluation, the performance of the algorithms with respect to the so-called recommendation task was measured. In particular, the task was to predict the top-five rated movies for a user. All these movies should be top-rated (4 or 5 stars in the MovieLens domain) by the users in the test set. Precision is applied as a metric. Twelve methods were compared, consisting of three naive baseline approaches, three standard CF methods, and six tag-based algorithms.

With respect to the recommendation task, the conclusion of this evaluation was that tag-based algorithms performed better than traditional methods and explicit algorithms showed better results than implicit-only methods. The best-performing approach was a combination of the best-performing explicit tag-based algorithm with Funk's value decomposition algorithm (described by Koren et al. 2009).

In addition to the recommendation task, the so-called prediction task was evaluated. In this task, the recommender system has to predict the rating of a user. Sen et al. (2009) use the MAE to assess the quality of algorithms. Two traditional CF methods, three baseline methods, and four tag-based algorithms were compared. In this evaluation, the tag-based methods did not show an improvement over the traditional CF methods. However, the evaluation was conducted on one particular domain.

11.2.1.2 Folksonomies and collaborative filtering

We now present two approaches that extend collaborative filtering techniques by tag information. The first one follows the memory-based method (Herlocker

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

et al. 1999), whereas the second one extends probabilistic matrix factorization (Koren et al. 2009). Finally, we show how CF methods can be applied for retrieving tagged items, given a query formulated as a set of tags.

Extensions of classical collaborative filtering methods. In contrast to a content-based view of tags, recently some researchers viewed tags as additional information for discovering similarities between users and items in the sense of CF.

In particular, Tso-Sutter et al. (2008) viewed tags as additional attributes providing background knowledge. Although there is a reasonable amount of work on integrating such additional information into CF, there are some important differences. *Attributes* in the classical sense are global descriptions of items. However, when tags are provided by users for items, the usage of tags may change from user to user. Consequently, tag information is local and three-dimensional: tags, items, and users.

The basic idea of Tso-Sutter et al. (2008) is to combine user-based and item-based CF. In user-based CF, usually the k most similar users are selected to compute a recommendation. The similarity of users is based on their ratings of items. Likewise, this similarity among users is influenced if users assign the same (or similar) tags. Consequently, the tags are just viewed as additional items. Therefore, the user–item rating matrix is extended by tags. The entries of this matrix are just Boolean values indicating whether a user is interested in an item or used a specific tag.

Similarly, item-based CF usually exploits the k most similar items rated by a user. Tags are viewed as additional users and the user–item matrix is extended by new users. The entries for these new users (i.e., the tags) are set to true if an item was labeled by the tag.

The ratings of unrated items i for user u is computed by the following formulas proposed by Tso-Sutter et al. (2008). The rating matrix $O_{x,y}$ has entry 1 or 0 for users x and items y. The dimensions of this matrix differ, depending on whether we apply user-based or item-based CF. Let N_u be the k most similar neighbors based on some traditional CF method. The user-based prediction value is computed by:

$$p^{\text{ucf}}(O_{u,i} = 1) := \frac{|\{v \in N_u | O_{v,i} = 1\}|}{|N_u|} \tag{11.10}$$

For the computation of an item-based prediction value, the k most similar items are exploited. N_i denotes these items, which are computed by applying some standard CF similarity function. The similarity between items i and j is

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

denoted by $w(i, j)$. The item-based prediction value is computed by:

$$p^{\text{icf}}(O_{u,i} = 1) := \sum_{j \in N_i \cap O_{u,j}=1} w(i, j) \tag{11.11}$$

Because the ranges of p^{ucf} and p^{icf} are different, they are normalized in the following combination formula. The parameter λ adjusts the significance of the two predictions and must be adjusted for a specific application domain.

The combined prediction value is computed by

$$p^{\text{iucf}}(O_{u,i} = 1) := \lambda \frac{p^{\text{ucf}}(O_{u,i} = 1)}{\sum_i p^{\text{ucf}}(O_{u,i} = 1)} + (1 - \lambda) \frac{p^{\text{icf}}(O_{u,i} = 1)}{\sum_i p^{\text{icf}}(O_{u,i} = 1)} \tag{11.12}$$

Based on these combined prediction values, the unrated items of user u can be ranked and the N top-ranked items are displayed to the user as a recommendation.

Tso-Sutter et al. (2008) evaluated this method based on the data of last.fm. Best results were achieved with $\lambda = 0.4$ and a neighborhood size of 20. The evaluation showed a significant improvement when tag information was exploited. Interestingly, this improvement could be shown only if item-based and user-based CF methods were combined.

The CF approach of Tso-Sutter et al. (2008) is a memory-based method. The second main approach for CF is the so-called model-based method, which tries to learn a model by employing statistical learning methods. Zhen et al. (2009) extended probabilistic matrix factorization (PMF) to exploit tag information. The key idea is to use tagging information to regularize the matrix factorization procedure of PMF.

Tag-based collaborative filtering and item retrieval. In contrast to the previously described approaches, social ranking – as introduced by Zanardi and Capra (2008) – is a method that aims to determine a list of potentially interesting items in the context of a user query. This query can either consist of a list of words provided by the user in a search engine scenario or be implicitly derived in one way or another from the user profile in a more classical recommendation scenario. Standard Web 2.0 search engines of this style use relatively simple metrics that combine a measure of overlap of search keywords and resource tags (ensuring accuracy) with a measure of how many users employed the particular tags for annotating the item (ensuring high confidence). Such measures, however, are good at retrieving popular items but not the so-called long tail of not-so-popular items.

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

In particular, we can distinguish between the long tail of tags and the long tail of items. The long tail of tags refers to the phenomenon that most of the tags are used only by a small subset of the user population. In the domain studied by Zanardi and Capra (2008), roughly 70 percent of the tags were used by twenty or fewer users, which represents roughly 0.08 percent of the whole user set. This suggests that standard keyword search will fail because of the small overlap of item tags and tags contained in a user query.

The long tail of items refers to the observation that most items are tagged by a small portion of the user population – 85 percent of the items are tagged by five or fewer users, as reported by Zanardi and Capra (2008). This suggests that standard recommender techniques fall short because of the almost empty overlap of user profiles.

Social ranking aims to overcome this problem by applying traditional CF ideas in a new way – by using user and tag similarities to retrieve a ranked list of items for a given user query. The similarity between users is determined based on an analysis of their tagging behavior. A simple definition of similarity is employed that states that users are considered similar if they have used the same set of tags (ignoring on which resources these tags have been put). When this information is encoded as a vector of tag usage counts, cosine similarity can be used as a metric to quantify the similarity. Similarly, tag similarity is calculated based on the co-occurrence of tags for a resource. Again, the cosine similarity measure can be used.

The calculated similarities are then used in the subsequent, two-step query phase as follows. To cope with the problem of free formulated tags, the original query u is expanded with a set of similar tags to a larger query u^*. The main idea here is to improve on the coverage measure, as users do not always use the same set of keywords to describe items. An expansion based on tag similarity will thus help to overcome this problem.

Next, all items that are tagged with at least one tag of u^* are retrieved and ranked. The ranking process is based on a function that consists of a combination of (a) the relevance of the tags with respect to the query and (b) the similarity of taggers to the active user who issued the query. The ranking $R(p)$ of item p is computed by summing up for every user u_i the similarities of the tags t_x user u_i used to tag p with the tags t_j contained in the expanded query q^* posed by the active user $\overline{u}$. This sum is amplified if the active user $\overline{u}$ is similar to user u_i:

$$R(p) = \sum_{u_i} \left(\sum_{\{t_x | u_i \text{ tagged } p \text{ with } t_x\}, t_j \in q^*} sim(t_x, t_j) \right) \times (sim(\overline{u}, u_i) + 1) \quad (11.13)$$

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

The approach was evaluated on the CiteULike social bookmarking data. Different views of the data have been used (compare with Massa and Avesani 2007), including heavy, medium, and low taggers, as well as popular and unpopular items. As a baseline, a standard method was used, as described above. It showed that social ranking consistently led to better results when it came to the long tail – that is, when searches for medium- or low-taggers or unpopular items were made. The query expansion method, quite intuitively, also helps improve on the coverage measure and thereby reduces the number of unsuccessful searches. Although the results are promising, further improvements seem possible – for example, through the usage of other methods for finding similar users or through the exploitation of external, semantic information (e.g., from WordNet) to make the query expansion more precise (Zanardi and Capra 2008).

As a side point, traditional recommender system algorithms (as well as standard quality measures; see Huang et al. 2008 and Firan et al. 2007) have been designed to work mostly on user-item matrices or content information. In social web and Web 2.0 scenarios, additional sources of information (such as trust relationships or tags) are available, however. Therefore, it can be expected that additional ways of exploiting these knowledge sources in an integrated way in Web 2.0 recommendation scenarios will be required in the future.

11.2.2 Recommending tags

As we see from these methods, one of the problems of folksonomies is that users differ in the way they annotate resources, so some fuzziness is introduced in the tags. This, in turn, hampers the computerized use of this information.

One way of alleviating this problem could be to enhance such a Web 2.0 system – such as a media-sharing platform – with an intelligent agent that supports the user in selecting and assigning tags to resources. This could not only help to align the sets of user tags but also serve as a motivator for end users to annotate the resources. Some of today's image and bookmark sharing, as well as music, platforms already provide some user guidance and tag recommendations. How these tagging suggestions are generated is, however, not published. It can be assumed that tag usage frequencies are at the core of these metrics.

The question arises of whether CF techniques can be a means for recommending a set of tags. As mentioned earlier, the problem setting is a bit different in folksonomies, as we do not have a two-dimensional user–item rating matrix but rather ternary relationships among users, resources, and tags. Jäschke et al. (2007) propose applying a nearest-neighbor approach, as follows: from the

original ternary relation (called Y), two different binary projections, the user resource table $\pi_{UR}Y$ and the user-tag table $\pi_{UT}Y$, can be derived. The values in these projections are binary. $(\pi_{UR}Y)_{u,r}$ is, for instance, set to 1 if there exists an element (u, t, r) in Y – that is, when the user u has rated resource r with an arbitrary tag. Based on one of these projections, we can compute N_u^k, the set of k most similar neighbors for a user u. The cosine similarity measure can be used for comparing two rows of the matrix.

After the set of nearest neighbors N_u^k is determined, and given a resource r and user u, we can compute the top n recommended tags $\tilde{T}(u, r)$ from the set of all tags T for user u as follows:

$$\tilde{T}(u, r) := argmax_{t \in T}^{n} \sum_{v \in N_u^k} sim(\vec{x}_u, \vec{x}_v)\delta(v, t, r) \tag{11.14}$$

where $\delta(v, t, r) := 1$ if $(v, t, r) \in Y$ and 0 otherwise. $\vec{x}_u$ and $\vec{x}_v$ are the rows of $\pi_{UT}Y$ or $\pi_{UR}Y$, depending on which of the possible projections was used.

As an alternative to this neighbor-based method, the same authors propose a recommendation technique based on FolkRank in Hotho et al. (2007). FolkRank is a graph-based search and ranking method for folksonomies. As the name suggests, it is inspired by PageRank, the underlying idea being that when resources are annotated with "important" tags by "influential" users, the resource will also become important and should be highly ranked in a search result. Because a direct application of PageRank is not possible because of the ternary nature of the relationships and the nondirectional relationships, an alternative form of weight-spreading is proposed by Hotho et al. (2007).

Jäschke et al. (2007) report the results of an evaluation of different approaches based on datasets from the popular social bookmarking systems Bibsonomy and last.fm. To determine the prediction accuracy, a variant of the standard leave-one-out method was chosen – that is, for a given resource the system should predict the tags a certain user would assign. Besides the CF variants and the graph-based FolkRank approach, an unpersonalized counting metric (*most popular tag by resource*) was also evaluated. For this metric, it was first counted how often every tag was associated with a certain resource. The tags that occurred most often with a given resource r were then used as a recommendation. The evaluation showed that FolkRank led to the best results with respect to both the recall and precision measure. The CF method performed slightly better when the user-tag projection was used; when the user-rating projection served as a basis, the performance of the CF algorithm was similar to the *most popular by resource* metric. The baseline method *recommend most popular tags* was outperformed by all other algorithms.

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

In summary, it can be seen that better prediction accuracy can be achieved with the computationally more expensive method based on FolkRank, and that an unpersonalized and easy-to-compute popularity-based method can already lead to relatively good results when compared with CF-based methods.

A new method that follows neither the PageRank nor the standard recommendation approach is proposed by Krestel et al. (2009). The authors employ the so-called latent Dirichlet allocation (LDA) to recommend tags. The basic idea is that various topics (e.g., *photography* or *how-to*) are hidden in the set of resources. Given a set of resources, tags, and users, LDA computes the probability that a tag t_i is related to a resource d by

$$P(t_i|d) = \sum_{j=1}^{Z} P(t_i|z_i = j)P(z_i = j|d) \tag{11.15}$$

where z_i represents a topic, $P(t_i|z_i = j)$ is the conditional probability that tag t_i is related to topic j, and $P(z_i = j|d)$ is the conditional probability that topic j is related to resource d. The number of topics Z is given as input. By exploiting the Gibbs sampling method, these conditional probabilities are estimated. As a result, by applying LDA, a numerical weight, expressing the association between tags and resources, can be computed. This weight is subsequently exploited to recommend tags for resources. Krestel et al. (2009) compare their approach to methods based on mining of association rules showing better results for LDA.

Krestel and Fankhauser (2009) extend this approach to tag recommendation by also considering the content of resources and by using a mixture of the unfactorized unigram language models with latent topic models. The evaluation on the bibsonomy dataset, provided by the discovery challenge at European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD) (Buntine et al. 2009), showed that on this dataset, language models slightly outperform latent topic models, but the mixture of both models achieves the best accuracy. Moreover, combining content with tags also yielded significant improvements.

A different method for generating tag recommendations on the popular image-sharing platform Flickr.com is proposed by Sigurbjörnsson and van Zwol (2008). In contrast to the aforementioned work, their recommendation system is strongly based on "tag co-occurrence" and different aggregation/ranking strategies. The method starts with a small set of given user-defined tags for an image and bases the recommendation on the aggregation of tags that are most frequently used together with the start set. Instead of simply using the "raw" frequencies, some normalization is required (similar to the TF-IDF method) to

take the overall frequency of tag usages into account. The candidate tags are then ranked (different strategies are possible) and a top-n list is presented to the user.

Besides this ranking technique, Sigurbjörnsson and van Zwol (2008) also introduce the idea of tag "promotion". An analysis of "how users tag photos" in the Flickr.com dataset showed that the tag frequency distribution follows the power law and that this distribution can be used to determine a set of the most promising candidates for recommendation. Very popular tags are too general to be useful; rarely used words, on the other hand, are unstable predictors. The evaluation of different promotion techniques based on a large extract of the Flickr.com dataset showed that good improvements with respect to precision can be achieved.

In contrast to the FolkRank method described above, the work of Sigurbjörnsson and van Zwol (2008) is based solely on data of an image-sharing platform. To what extent their findings can also be transferred to other social web platforms and other types of resources has not been analyzed so far. Examples for other recent collaborative, content-based, or probabilistic methods for tag recommendation can be found in the work of Xu et al. (2006), Basile et al. (2007), Byde et al. (2007), and Mishne (2006).

11.2.3 Recommending content in participatory media

Besides resource tagging and participation in social networks, a third aspect of the second-generation web is the phenomenon of the increasing importance of *participatory media*, in which users contribute the content – for instance, in the form of blog messages.

On such media platforms – and in particular, on popular ones that have a broad community of contributors – the problem of filtering out the interesting messages quickly arises. At first glance, this problem is very similar to the classical news filtering problem discussed in Chapters 2 and 3. This time, however, we can hope to have more information available than just the active user's reading history and preference statements; we can additionally try to exploit the information in the social network to judge whether the message is important. Messages can, for instance, be judged to be important because other users rated them highly, which is the standard CF setting. We have also seen that explicit trust statements can help improve the accuracy of such estimates of interest. Being allowed to issue only one trust statement per person might, however, be too simplistic. In reality, we typically do not trust the opinion of a friend on every topic. Moreover, we are often embedded in a social environment,

in which there are particularly important persons whose opinions we generally believe.

Given that more of this social information is available in Web 2.0, it would be desirable to take these pieces of information properly into account when recommending messages to a user. A first approach to incorporating such phenomena that appear in real-world social relationships into a computerized system has been made by Seth et al. (2008). In their work, they aim to develop a metric of credibility that takes into account various factors that should implicitly help to determine whether a recently posted message is credible (i.e., possibly relevant and important) to a user.

One of the basic ideas of their approach is that their multidimensional credibility measure is subjective – that is, different users may judge the credibility of a posting differently, possibly depending on their context and their community. In addition, Seth et al. postulate that credibility must be topic-specific – that is, one can be a trusted expert in one field and not trusted when it comes to a different subject area.

The proposed metric is based on various insights from the fields of media studies, political science, and social networks and combines different aspects that contribute to a person's credibility. These aspects range from direct experiences to the credibility we attribute to someone because of his or her role or local community, and including, finally, the general opinion of the public about a certain user. In Seth et al.'s model, each of the individual credibility measures are captured in a real number $[0 \ldots 1]$ and combined in a Bayesian network to an overall credibility estimate.

The Bayesian model is trained based on stochastic methods using the following data, which are assumed to be available:

- Messages are labeled with their authors (or posters) and a set of ratings that describe the supposed credibility of the message. These ratings are, as usual, provided by the users of the community.
- Users are explicitly connected with their "friends".
- Every user can declare a list of topics in which he or she is interested. Based on this, a set of topic-specific social network graphs can be derived, in which clusters of users and links can be identified with the help of community identification algorithms. From this, strong and weak ties between users can be identified.

On arrival of a new message, the learned model can be used to make a probabilistic prediction as to whether a user will find this message credible.

For evaluation purposes, a prepared data set from the digg.com knowledge sharing platform was used. The digg.com web site allows users to post articles,

rate other items and also to connect to other users. Unfortunately, the first reported measurements by Seth et al. (2008) are not yet fully conclusive. Still, one can see this work as a further step toward the more extensive and integrated usage of information in social web platforms, with the ultimate goal of filtering interesting items more precisely in the future.

The idea to differentiate between users is also followed by Guy et al. (2009). Basically, the authors build on the insight to distinguish between people who are similar to a user and people who are familiar with a user.

The computation of a familiarity score between users is based on organizational charts, direct connections in a social network system, tagging of persons, and coauthorship of various items, such as papers, wikis, and patents. The computation of a similarity score is based on the co-usage of the same tag, co-bookmarking of the same web page, and co-commenting on the same blog entry. All these pieces of information are exploited to compute an overall score. Based on these scores, information items such as web pages, blog entries, and communities are recommended to users.

Furthermore, for a recommendation of an item, the names of the most related persons are displayed, serving as a kind of explanation why this item is recommended.

Guy et al. (2009) compared recommendations based exclusively on either the familiarity, similarity, or overall score. Test persons were asked to classify recommended items as *interesting*, *not interesting*, and *already known*. The evaluation showed that recommendations based on the familiarity score were classified as significantly more interesting than recommendations based on the similarity score. In addition, it was shown that providing explanations as described here results in a significant increase in the number of items classified as interesting.

The study was conducted in a work environment and did not ask the utility of the presented information objects for the users' tasks. Therefore, it is open whether the observed effect contributes rather to a persuasion or leads to the discovery of information objects that indeed support the work process.

The problem of filtering information in Web 2.0 collaborative media platforms is already in the focus of providers of commercial portal software. Nauerz et al. (2008), for instance, report on the implementation of a "recommender of background information" in the IBM WebSphere portal system. In their work, the selection of similar documents is based on automatically extracted or manually assigned meta-tags. Although details of the recommendation algorithms are not given and a straightforward method can be assumed, this example shows the practical relevance of providing additional support in Web 2.0 collaborative systems.

11.3 Ontological filtering

As mentioned earlier, a fundamental building block of the Semantic Web is the description of web resources by languages that can be interpreted by software systems. In particular, the idea is to better locate information on the web by such descriptions. Indeed, parts of the Semantic Web community deal with the classification of web content that best matches the information need of users by exploiting machine interpretable information – for example, web content is annotated and a formal description language (e.g., OWL) is applied to deduce classifications. This task shares many similarities with recommender systems that aim at the classification of items that best fulfill some user needs.

In particular, one central idea of the Semantic Web is to formulate a domain ontology (i.e., a logical theory) that is exploited to describe web resources and to reason about the properties of these resources by inference systems. Consequently, various researchers have applied ontologies to improve filtering techniques of recommender systems. Of course, one can argue that, in fact, long-known knowledge-based techniques have been applied, such as simple inheritance taxonomies and other forms of logical description of items and their relations. Therefore, these recommender systems are actually hybrid systems leveraging their capabilities by knowledge-based methods.

Most of the research in this area, however, was published in the context of the Semantic Web umbrella, and therefore we prefer to mirror this originally intended classification.

11.3.1 Augmentation of filtering by taxonomies

Assume that there is an ontology describing the domain by a super-/subconcept taxonomy with the meaning that every member of the subconcept is also a member of the superconcept, but not necessarily vice versa. For example, news can be classified by such an ontology, saying that news about "world soccer tournaments" is more specific than news about "soccer", which is more specific than news about "sport". In addition, news about "baseball" is different from news about "soccer" but more specific than "sport". In the following, we distinguish between parent relations (e.g., "sport" is a parent of "soccer") and grandparent relations (e.g., "sport" is a grandparent of "world soccer tournaments").

Based on this hierarchical ontology, items (e.g., pieces of information on the web) can be associated with such concepts. The set of concepts associated to an item is called the *item profile*. For example, a news item can be annotated by the label "soccer". Conversely, information about the interests of users can be provided by associating concepts to individual users and annotating these

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

associations by the strength of their interests. The set of concepts associated to a user is called the *user profile*. For example, a user is strongly interested in "soccer" but not in "baseball". Information about the interests of users can be either provided directly by the users or acquired by indirect means, such as observing which items are clicked. By such observations, the interests of users can be continuously enhanced.

Given these three pieces of information (item profile, user profile, and the domain taxonomy) Maidel et al. (2008) propose a similarity function between items and user interests, in which the taxonomy is exploited.

Five different cases of matches were considered, in which each concept c_i in the item profile is assigned to a matching score depending on the user profile.

Case *a*: The perfect match. The item concept c_i is also contained in the user profile – for example, both the user and item profiles contain "soccer".

Case *b*: The item concept c_i is a parent of a concept contained in the user profile – for instance, the item concept is "sport" and the user profile contains "soccer".

Case *c*: The item concept c_i is a child of a concept contained in the user profile – for example, the item concept is "soccer" and the user profile contains "sport".

Case *d*: The item concept c_i is a grandparent of a concept contained in the user profile – for instance, the item concept is "sport" and the user profile contains "world soccer tournaments".

Case *e*: The item concept c_i is a grandchild of a concept contained in the user profile – for example, the item concept is "world soccer tournaments" and the user profile contains "sport".

For all the these cases, matching scores must be determined. Whereas the matching score of case *a* (the perfect match) is 1 (the maximum), the score values of the other cases must be determined. Intuitively, matching scores for cases *d* and *e*, which are defined by grandparent/grandchild relations, are lower than scores for cases *b* and *c*, which are specified by parent/child relations.

The degree of similarity between an item and a user is based on the profiles, matching scores of the concepts in the two profiles, and on the weights of the concepts in the user profile. The overall similarity score between item and user is defined by

$$IS = \frac{\sum_{c_i \in I} N_{c_i} S_{c_i}}{\sum_{c_j \in U} N_{c_j}} \tag{11.16}$$

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

where I is the item profile, U is the user profile, c_i is a concept in the item profile, c_j is a concept in the user profile, S_{c_i} is the matching score of concept c_i, and N_{c_j} is the number of clicks on items containing concept c_j, representing the weight of concept c_j for the user. This weight of concepts for a user may be given implicitly by monitoring the user or may be specified explicitly by the user.

Formula 11.13 may be extended by weights representing the importance of concepts in items (concept/item weights). In particular, the matching scores S_{c_i} can be multiplied by the weight of c_i in the item. However, the evaluation by Maidel et al. (2008) showed no significant improvements if concept/item weights are considered. Furthermore, Maidel et al. (2008) experimented systematically with different settings of matching scores. It turned out that the best choice is around $a = 1$, $b = 0.8$, $c = 0.4$, $d = 0$, and $e = 0.2$. Consequently, after the perfect match, the next important parameter is b, representing the match in which the item's concept is a more general parent concept of a user's concept. The next important parameter is c, in which a user's concept is a more general parent concept of an item's concept. The different weights depend on the direction of the generalization. The parameters d and e, in which item and user concepts are in a grandparent/grandchild relation, turned out to be less important.

The extensive evaluation presented by Maidel et al. (2008) showed that if weights of concepts representing the interests of users are given explicitly, much better recommendation results can be achieved compared with the case in which the interest of users is acquired during a session. Moreover, it was shown that if the taxonomy is not considered in the overall similarity score IS (i.e., only perfect matches are counted), the recommendation quality significantly drops. Consequently, the utility of exploiting taxonomies in matching user and item profiles for improving the error in this type of recommender systems was shown.

The approach of Maidel et al. (2008) was applied to recommend news, and follows the idea of Middleton et al. (2004) to take advantage of the information contained in taxonomies. Unfortunately, a comparison with Middleton et al. (2004) is missing.

The underlying recommendation method of Middleton et al. (2004) is a combination of content-based and collaborative techniques. The goal of the system is to recommend "interesting" research papers to computer scientists. The interests of a computer scientist are described by a set of computer science topics (such as hypermedia, artificial intelligence, or agents); each topic is weighted by a numerical value. These topics are organized in a subconcept-superconcept

hierarchy, forming a simple ontology of research fields and their subfields. For example, papers in the field of recommender agents are papers in the agents field, which are papers in the artificial intelligence field. Computer scientists in a department read articles, and these articles form an article repository.

The idea of the Foxtrot system is based on the following information pieces; given that we know the topics in which a researcher is interested and the topic for each research paper, then we can estimate the interest a researcher possibly has in a particular paper. To estimate the topics of research papers, the Foxtrot system starts with a set of manually classified documents (the training set), in which their topic is known. The topic of unclassified research papers is estimated by classical content-based techniques. In particular, a research paper is represented by its term frequency, in which the set of terms is reduced by standard techniques, such as stemming and the application of a stop list to remove common words.

The interests of researchers are determined by various data sources. Researchers can explicitly declare in which topics they are interested. In addition, the Foxtrot system monitors which papers a researcher browsed. Because the topic of papers is known, it is assumed that the researcher is interested in this topic. Based on this information, a *topic interest value* is computed. This topic interest value is a summation of various single interest values. Given a user u and a specific topic t, the topic interest value $i(u, t)$ is increased for each paper browsed by user u if the paper's topic is classified to be t. Browsing papers that are recommended by the system is given a higher weight. Furthermore, the interest values of papers are weighted by time. The more recently a paper was browsed, the higher its weight. In addition, the topic interest value is increased or decreased by a certain amount if the user explicitly declared his or her interest or disinterest. The most important fact of the computation of the topic interest value is that the topic interest value of topic t is increased by 50 percent of the topic interest values of its subtopics.

Papers are ranked for a specific user by their recommendation value. The recommendation value for a paper p and a user u is computed, as usual, by summing over all topics t and multiplying the classification confidence (i.e., a measure of how confident we are that paper p deals with topic t) and the topic interest value of the user u in that topic t.

In an experimental evaluation, it was shown that the association of topics to users is more accurate if the ontology of topics is employed compared to the case in which only a flat list of topics was exploited. In addition, the recommendation accuracy could be improved by using an ontology.

The cold-start problem is addressed by Middleton et al. (2004) by using previously published papers of researchers. These unclassified papers are

compared with the classified papers of the repository to estimate their topic classification. The interest values of a user are computed based on the topics of his or her published papers, weighting previously published papers higher. In addition, the ontology of topics is exploited as described – the interest in a topic is increased by some fraction depending on the interest values of the subtopics. Furthermore, if the system is up and running and a new user is added, this new user is compared with already participating users. The interest values of a new user are adjusted depending on the interest values of similar users.

In contrast to the work of Middleton et al. (2004), the work of Ziegler et al. (2004) follows a different strategy of propagated interests in a topic along the taxonomy. Furthermore, a different strategy for computing recommendations is developed that also addresses the problem of topic diversification – recommending not always more of the same, but pinpointing different but still interesting items.

The basic idea of Ziegler et al. (2004) for incorporating taxonomic information is to base the recommendation algorithm on the user's interest in categories of products. Consequently, user similarity is determined by common interests in categories and not by common interests in items. The goal of this approach is to reduce problems of collaborative filtering methods if user ratings are sparse. The rationale is that although users have not rated the same item, they may show common interest in a category – for instance, even though different books were bought, these books can belong to the same topic.

In particular, Ziegler et al. (2004) assume a set of products $B = \{b_1, \ldots, b_m\}$ and a set of user ratings R_i for every user u_i where $R_i \subseteq B$. In this approach, a user is either interested in a product if this product is in R_i, or we have no information about his or her interest in a product if this product is not in R_i. This is the typical case of e-commerce applications, in which interests in products are implicitly rated by purchase data or product mentions. In addition, a set of product categories $D = \{d_1, \ldots, d_l\}$ is given, representing topics into which products may fall. These product topics are organized in a tree representing subconcept-superconcept relations between them. For every product, a descriptor assignment function is defined that associates to each product b_k a subset $D_k \subseteq D$ of topics. Product b_k is a member of each topic in D_k.

Instead of representing a user's interest by a vector of dimension $|B|$ (i.e., for each product the vector contains an entry), the user's interests are characterized by a vector of dimension $|D|$, in which each entry represents the interest of a user in a topic.

Based on the user rating R_i, the set of products in which the user is interested can be determined. By the descriptor assignment function, the topics can be computed in which a user is interested. The interest in a topic is propagated

from the subtopics to the supertopics. The propagating depends on the number of siblings a topic has. The fewer siblings a topic possesses, the more interest is assigned to its supertopic. The results of this computation are user interest vectors containing scores for topics. These user interest vectors are exploited to compute the similarities of users based on Pearson correlation. Finally, as usual, for each user the k nearest neighbors are determined.

The generation of recommendations combines two proximity values. *User proximity* takes into account how similar two users are. If user u_j recommends an item to user u_i, this recommendation receives more weight the closer the interest profiles of u_i and u_j are. The second proximity exploited is called *product proximity*. Here the idea is that the closer the topics of a product are to the topics in which a user is interested, the higher the weight for recommending this product. Based on these proximity values, weights for products are computed, and the ordered set of the top N products represents the recommendation.

Finally, Ziegler et al. (2004) propose a method for topic diversification to avoid the phenomenon that only products of the most interesting topic are recommended. Here the idea is that the recommendation list is incrementally expanded. The topics that have not been covered in the recommendation list so far receive higher weights. These weights are increased for a topic d_j depending on the length of the list – the longer the list that does not contain an item of d_j, the higher the weights.

Evaluations show the advantage of this approach compared with standard CF approaches and hybrid approaches combining content-based and collaborative filtering.

11.3.2 Augmentation of filtering by attributes

Mobasher et al. (2004) exploited semantic information to enhance item-based CF. The basic idea is to use semantic information about items to compute similarities between them. These semantic similarities are combined with similarities based on past user ratings to estimate future user ratings. In particular, it is assumed that an ontology describes a domain, such as movies. This ontology describes the attributes used to characterize items. For example, in the movie domain, these attributes are genre, actors, director, and name. In a further step, the method assumes the instantiation of the ontology by items. Mobasher et al. (2004) accomplished this instantiation process by web mining techniques.

To support the computation of item similarities, the instances are converted into a vector representation. This conversion includes normalization and discretization of continuous attributes. The process also results in the addition of new attributes, such as representing different intervals in a continuous range or

representing each unique discrete value for categorial attributes. The outcome of this process is a $n \times d$ matrix $S_{n \times d}$, where n is the number of items and d is the number of unique semantic attributes. Matrix S is called the *semantic attribute matrix*.

In a further step, singular value decomposition (SVD) is applied to reduce the number of attributes of the matrix. SVD is a well-known technique of latent semantic indexing (Deerwester et al. 1990), which has been shown to improve accuracy of information retrieval. Each dimension in the reduced space is a latent variable representing groups of highly correlated attributes. Reducing the dimensionality of the original matrix reduces noise of the data and its sparsity, thus addressing inherent problems of filtering techniques.

Based on the semantic attribute matrix similarities, $SemSim(i_p, i_q)$ for all pairs of items i_p and i_q are computed. For this computation the standard vector-based cosine similarity is applied. In addition, the usual item similarities $RateSim(i_p, i_q)$ are computed based on the ratings of users. Finally, these two similarities are combined by a linear function: $CombinedSim(i_p, i_q) = \alpha \cdot SemSim(i_p, i_q) + (1 - \alpha) \cdot RateSim(i_p, i_q)$. The best value for α depends on the domain and is determined by a sensitivity analysis. The predictions of ratings for a user u_a regarding item i_t is realized by a weighted sum approach exploiting the combined similarity values of k nearest neighbors (see Section 2.1). Mobasher et al. (2004) reported results of an evaluation that show that the proposed approach improves the prediction accuracy and that the application of SVD results in further enhancements. Furthermore, the experiments show that the approach produces reasonably accurate predictions of user ratings for new items, thus alleviating the "new item problem" of CF techniques.

11.4 Extracting semantics from the web

Semantic information can provide valuable means for improving recommendations. However, where does this information come from, and how costly and reliable is the acquisition process?

To address this problem, we can distinguish two approaches to generate semantic information. The first approach assumes that humans are providing semantics by annotating content and by declaring logical sentences. The second approach is to develop software systems that are able to generate semantics with little or no human intervention. Given the lessons learned in applying knowledge-based systems, this second path is particularly attractive, as it reduces the needed development and maintenance efforts.

In the work described by Shani et al. (2008), the basic idea is to generate the information needed for CF systems through web mining. They developed two

different methods, *WebCount* and *WebCrawl*. WebCount is based on the cosine score $cosine(i_1, i_2)$ for binary ratings (i.e., the user expresses only that he or she likes or does not like an item) where i_1 and i_2 are items, and $count(i_1, i_2)$ is the number of users who liked both item i_1 and i_2. $count(i)$ is the number of users who just liked i.

$$cosine(i_1, i_2) = \frac{count(i_1, i_2)}{count(i_1)count(i_2)} \tag{11.17}$$

Based on these scores, item-based recommendations can be computed as described in Section 2.2. Shani et al. (2008) propose to use the number of pages that list an item i as an approximation of the count of i. Similarly, $count(i_1, i_2)$ is estimated by the number of pages that mention both items. The simplest way to acquire these numbers is to input the names of items into a web search engine and to approximate the count by the number of hits returned – for instance, in the movie domain, the movie names are exploited.

Obviously, this approach gives a rough estimation, which could be improved by more sophisticated query and mining techniques. For the movie domain, a simple improvement extends the query with additional keywords and filters, such as "movie recommendations" OR "recommended movies" OR "related movies" (Shani et al. 2008).

The WebCrawl method applies a more sophisticated strategy that, however, may not be generally applicable. The idea is that in web systems, which host a web community, the members of such a community provide data for item ratings in their profiles. In particular, in some web communities (such as MySpace), members declare the list of movies or musicians they like. Each page in WebCrawl is treated as a user, and the items recognized on this web page are counted as a positive rating. Obviously, such a method could be easily enhanced by more sophisticated web crawling techniques such as detecting if the item is really a positive rating or rather negative.

Shani et al. (2008) describe a comparison of WebCount and WebCrawl with a standard approach. This approach exploits the Netflix dataset, in which users explicitly rated movies. This comparison showed that WebCrawl provided the best recommendations. Ratings based on Netflix were close behind, with WebCount somewhat worse. Given that the methods for mining the web could be easily enhanced for both WebCount and WebCrawl, it seems reasonable that web mining techniques will play an important role in improving the data collection for CF techniques. The results suggest that in the movie domain, the user–item matrix generated by crawling MySpace has similar or better quality compared with the one derived from the Netflix ratings. In addition, the results of the simple WebCount method were surprisingly good, as more than

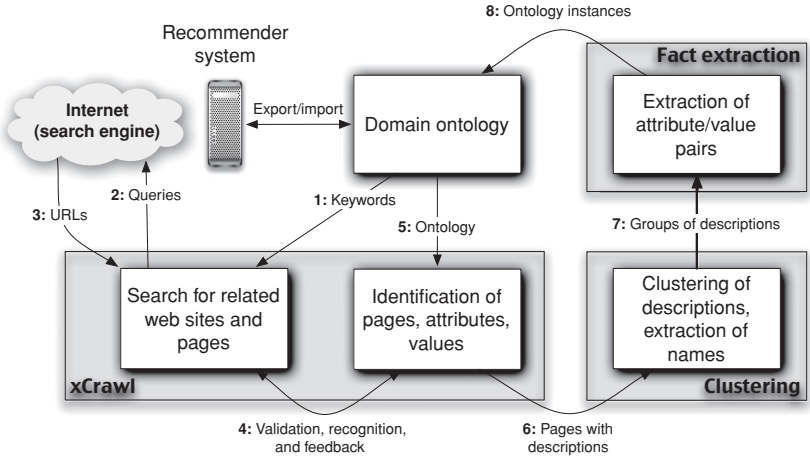

Figure 11.3. Workflow of the AllRight ontology instantiation system.

70 percent of the recommendation lists computed by WebCount were classified as reasonable by the users.

As shown by the contribution of Shani et al. (2008), CF systems can take advantage of the huge amount of information on the web. However, the exploitation of the web to enhance recommendation technology is not limited to collaborative methods. In fact, web mining techniques have a high potential to reduce the efforts for implementing and maintaining the knowledge bases of knowledge-based recommender systems.

In particular, knowledge-based recommenders require a description of all available products and their attributes. Acquiring and maintaining this data could be a costly and error-prone task. Jannach et al. (2009) describe the AllRight system, which instantiates an ontology by exploiting tabular web sources.

The basic idea of the AllRight system is to search the web for tables describing products (e.g., digital cameras), which are the information sources to populate product ontologies. For this task, numerous problems must be solved, as the web content is designed to be easily comprehended by humans, but not by machines. Figure 11.3 depicts the workflow of the AllRight system, which also shows the major challenges.

In a first step, the knowledge engineer must specify the set of attributes that describe the products of a domain (e.g., digital cameras). This description includes the domains of the attributes and their units. In addition, the knowledge engineer can associate keywords to attributes and units reflecting the fact that

https://doi.org/10.1017/CBO9780511763113.013 Published online by Cambridge University Press

various names are used to name attributes and units. As shown in Figure 11.3, the system exploits these keywords (1) to crawl the web by xCrawl, by posting queries (2) and downloads (3) all web pages that describe products of the domain. The downloaded pages are analyzed by the Identification component (4) to check if the pages contain the desired product descriptions. This is supported by the domain ontology, which provides constraints exploited by the validation. To correct errors, it is desirable to have many redundant descriptions of products, as this allows us to clean data by statistical methods. The identified pages containing product information are forwarded to a module that clusters the pages so each cluster describes one product (7). In a further step, attribute/value pairs are extracted from these clusters, which serve as an input to create instances of the domain ontology (8). The result is a fact extraction system that achieves an F-measure between 0.8 and 0.9 for digital cameras and laptops, which served as test domains.

11.5 Summary

In this chapter we have shown the opportunities, current methods, and realizations of Web 2.0 and the Semantic Web for the field of recommender systems. With these new developments of the web, we are able to exploit additional information so users can be better served by recommender technology. We have shown how this information can contribute to more trustworthy and qualitative enhanced recommendations satisfying information needs. In particular, we started with approaches exploiting very little semantics (lightweight ontologies) and moved to semantically richer domains. Finally, we pointed out how semantics can be extracted from the web.

We must acknowledge, however, that the advancements of the web are still tremendously fast. Consequently, we have provided the current state, which outlines the fundamental development paths of web-based recommender technology. Both Web 2.0 and the Semantic Web in combination not only drive new technologies but, maybe even more important, also have huge impacts on society regarding the communication and interaction patterns of humans. This can be impressively observed through the success and growth rate of various web-based service and communication platforms. Technological as well as social developments provide, on one hand, new means to improve recommendation methods, but on the other hand, generate new demands for supporting humans with advice in a more complex and faster moving world. Consequently, we will see many new developments and high-impact applications of recommender technology in the context of Web 2.0 and the Semantic Web, boosted by a growing interest of major Internet players.