R

Recommender Systems

Prem Melville, Vikas Sindhwani
Machine Learning,
IBM T. J. Watson Research Center,
Route 134/P.O. Box 218,
1101 Kitchawan Rd,
Yorktown Heights,
NY 10598, USA

Definition

The goal of a recommender system is to generate meaningful recommendations to a collection of users for items or products that might interest them. Suggestions for books on Amazon, or movies on Netflix, are real-world examples of the operation of industry-strength recommender systems. The design of such recommendation engines depends on the domain and the particular characteristics of the data available. For example, movie watchers on Netflix frequently provide ratings on a scale of 1 (disliked) to 5 (liked). Such a data source records the quality of interactions between users and items. Additionally, the system may have access to user-specific and item-specific profile attributes such as demographics and product descriptions, respectively. Recommender systems differ in the way they analyze these data sources to develop notions of affinity between users and items, which can be used to identify well-matched pairs. ►Collaborative Filtering systems analyze historical interactions alone, while ►Content-based Filtering systems are based on profile attributes; and hybrid techniques attempt to combine both of these designs. The architecture of recommender systems and their evaluation on real-world problems is an active area of research.

Motivation and Background

Obtaining recommendations from trusted sources is a critical component of the natural process of human decision making. With burgeoning consumerism buoyed by the emergence of the web, buyers are being presented with an increasing range of choices while sellers are being faced with the challenge of personalizing their advertising efforts. In parallel, it has become common for enterprises to collect large volumes of transactional data that allows for deeper analysis of how a customer base interacts with the space of product offerings. Recommender systems have evolved to fulfill the natural dual need of buyers and sellers by automating the generation of recommendations based on data analysis.

The term "collaborative filtering" was introduced in the context of the first commercial recommender system, called Tapestry (Goldberg, Nichols, Oki, & Terry, 1992), which was designed to recommend documents drawn from newsgroups to a collection of users. The motivation was to leverage social collaboration in order to prevent users from getting inundated by a large volume of streaming documents. Collaborative filtering, which analyzes usage data across users to find well-matched user-item pairs, has since been juxtaposed against the older methodology of content filtering, which had its original roots in information retrieval. In content filtering, recommendations are not "collaborative" in the sense that suggestions made to a user do not explicitly utilize information across the entire user-base. Some early successes of collaborative filtering on related domains included the GroupLens system (Resnick, Neophytos, Bergstrom, Mitesh, & Riedl, 1994b).

As noted in Billsus and Pazzani (1998), initial formulations for recommender systems were based on straightforward correlation statistics and predictive modeling, not engaging the wider range of practices in statistics and machine learning literature. The collaborative filtering problem was mapped to classification, which allowed dimensionality reduction techniques to be brought into play to improve the quality of the solutions. Concurrently, several efforts attempted

C. Sammut, G. Webb (eds.), *Encyclopedia of Machine Learning*, DOI 10.1007/978-0-387-30768-8, © Springer-Verlag Berlin Heidelberg, 2010

to combine content-based methods with collaborative filtering, and to incorporate additional domain knowledge in the architecture of recommender systems.

Further research was spurred by the public availability of datasets on the web, and the interest generated due to direct relevance to e-commerce. Netflix, an online streaming video and DVD rental service, released a large-scale dataset containing 100 million ratings given by about half-a-million users to thousands of movie titles, and announced an open competition for the best collaborative filtering algorithm in this domain. Matrix Factorization (Bell, Koren, & Volinsky, 2009) techniques rooted in numerical linear algebra and statistical matrix analysis emerged as a state-of-the-art technique.

Currently, recommender systems remain an active area of research, with a dedicated ACM conference, intersecting several subdisciplines of statistics, machine learning, data mining, and information retrievals. Applications have been pursued in diverse domains ranging from recommending webpages to music, books, movies, and other consumer products.

Structure of Learning System

The most general setting in which recommender systems are studied is presented in Fig. 1. Known user preferences are represented as a matrix of n users and m items, where each cell $r_{u,i}$ corresponds to the rating given to item i by the user u. This *user ratings matrix* is typically sparse, as most users do not rate most items. The *recommendation task* is to predict what rating a user would give to a previously unrated item. Typically, ratings are predicted for all items that have not been observed by a user, and the highest rated items are presented as recommendations. The user under current consideration for recommendations is referred to as the *active user*.

The myriad approaches to recommender systems can be broadly categorized as:

- *Collaborative Filtering (CF)*: In CF systems, a user is recommended items based on the past ratings of all users collectively.
- *Content-based recommending*: These approaches recommend items that are similar in content to items the user has liked in the past, or matched to predefined attributes of the user.

		Items					
		1	*2*	...	*i*	...	*m*
Users	*1*	5	3		1	2	
	2		2				4
	:			5			
	u	3	4		2	1	
	:					4	
	n			3	2		
	a	3	5		?	1	

Recommender Systems. Figure 1. User ratings matrix, where each cell $r_{u,i}$ corresponds to the rating of user *u* for item *i*. The task is to predict the missing rating $r_{a,i}$ for the active user *a*

- *Hybrid approaches*: These methods combine both collaborative and content-based approaches.

Collaborative Filtering

Collaborative filtering (CF) systems work by collecting user feedback in the form of ratings for items in a given domain and exploiting similarities in rating behavior amongst several users in determining how to recommend an item. CF methods can be further subdivided into *neighborhood-based* and *model-based* approaches. Neighborhood-based methods are also commonly referred to as *memory-based* approaches (Breese, Heckerman, & Kadie, 1998).

Neighborhood-based Collaborative Filtering In neighborhood-based techniques, a subset of users are chosen based on their similarity to the active user, and a weighted combination of their ratings is used to produce predictions for this user. Most of these approaches can be generalized by the algorithm summarized in the following steps:

1. Assign a weight to all users with respect to similarity with the active user.
2. Select k users that have the highest similarity with the active user – commonly called the *neighborhood*.
3. Compute a prediction from a weighted combination of the selected neighbors' ratings.

In step 1, the weight $w_{a,u}$ is a measure of similarity between the user u and the active user a. The most commonly used measure of similarity is the Pearson correlation coefficient between the ratings of the two users (Resnick, Iacovou, Sushak, Bergstrom, & Reidl,

1994a), defined below:

$$w_{a,u} = \frac{\sum_{i \in I} (r_{a,i} - \bar{r}_a)(r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i \in I} (r_{a,i} - \bar{r}_a)^2 \sum_{i \in I} (r_{u,i} - \bar{r}_u)^2}} \quad (1)$$

where I is the set of items rated by both users, $r_{u,i}$ is the rating given to item i by user u, and $\bar{r}_u$ is the mean rating given by user u.

In step 3, predictions are generally computed as the weighted average of deviations from the neighbor's mean, as in:

$$p_{a,i} = \bar{r}_a + \frac{\sum_{u \in K} (r_{u,i} - \bar{r}_u) \times w_{a,u}}{\sum_{u \in K} w_{a,u}} \quad (2)$$

where $p_{a,i}$ is the prediction for the active user a for item i, $w_{a,u}$ is the similarity between users a and u, and K is the neighborhood or set of most similar users.

Similarity based on Pearson correlation measures the extent to which there is a linear dependence between two variables. Alternatively, one can treat the ratings of two users as a vector in an m-dimensional space, and compute similarity based on the cosine of the angle between them, given by:

$$w_{a,u} = \cos(\vec{r}_a, \vec{r}_u) = \frac{\vec{r}_a \cdot \vec{r}_u}{\|\vec{r}_a\|_2 \times \|\vec{r}_u\|_2} = \frac{\sum_{i=1}^{m} r_{a,i} r_{u,i}}{\sqrt{\sum_{i=1}^{m} r_{a,i}^2} \sqrt{\sum_{i=1}^{m} r_{u,i}^2}} \quad (3)$$

When computing cosine similarity, one cannot have negative ratings, and unrated items are treated as having a rating of zero. Empirical studies (Breese et al., 1998) have found that Pearson correlation generally performs better. There have been several other similarity measures used in the literature, including *Spearman rank correlation*, *Kendall's τ correlation*, *mean squared differences*, *entropy*, and *adjusted cosine similarity* (Herlocker, Konstan, Borchers, & Riedl, 1999; Su & Khoshgoftaar, 2009).

Several extensions to neighborhood-based CF, which have led to improved performance are discussed below.

Item-based Collaborative Filtering: When applied to millions of users and items, conventional neighborhood-based CF algorithms do not scale well, because of the computational complexity of the search for similar users. As a alternative, Linden, Smith, and York (2003) proposed *item-to-item* collaborative filtering where rather than matching similar users, they match a user's rated items to similar items. In practice, this approach leads to faster online systems, and often results in improved recommendations (Linden et al., 2003; Sarwar, Karypis, Konstan, & Reidl, 2001).

In this approach, similarities between pairs of items i and j are computed off-line using Pearson correlation, given by:

$$w_{i,j} = \frac{\sum_{u \in U} (r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in U} (r_{u,i} - \bar{r}_i)^2} \sqrt{\sum_{u \in U} (r_{u,j} - \bar{r}_j)^2}} \quad (4)$$

where U is the set of all users who have rated both items i and j, $r_{u,i}$ is the rating of user u on item i, and $\bar{r}_i$ is the average rating of the ith item across users.

Now, the rating for item i for user a can be predicted using a simple weighted average, as in:

$$p_{a,i} = \frac{\sum_{j \in K} r_{a,j} w_{i,j}}{\sum_{j \in K} |w_{i,j}|} \quad (5)$$

where K is the neighborhood set of the k items rated by a that are most similar to i.

For item-based collaborative filtering too, one may use alternative similarity metrics such as *adjusted cosine similarity*. A good empirical comparison of variations of item-based methods can be found in Sarwar et al. (2001).

Significance Weighting: It is common for the active user to have highly correlated neighbors that are based on very few co-rated (overlapping) items. These neighbors based on a small number of overlapping items tend to be bad predictors. One approach to tackle this problem is to multiply the similarity weight by a *significance weighting* factor, which devalues the correlations based on few co-rated items (Herlocker et al., 1999).

Default Voting: An alternative approach to dealing with correlations based on very few co-rated items is to assume a default value for the rating for items that have not been explicitly rated. In this way one can now compute correlation (Eq. 1) using the union of items rated by users being matched as opposed to the intersection. Such a *default voting* strategy has been shown to improve collaborative filtering by Breese et al. (1998).

Inverse User Frequency: When measuring the similarity between users, items that have been rated by all (and

universally liked or disliked) are not as useful as less common items. To account for this Breese et al. (1998) introduced the notion of *inverse user frequency*, which is computed as $f_i = \log n/n_i$, where n_i is the number of users who have rated item i out of the total number of n users. To apply inverse user frequency while using similarity-based CF, the original rating is transformed for i by multiplying it by the factor f_i. The underlying assumption of this approach is that items that are universally loved or hated are rated more frequently than others.

Case Amplification: In order to favor users with high similarity to the active user, Breese et al. (1998) introduced *case amplification* which transforms the original weights in Eq. (2) to

$$w'_{a,u} = w_{a,u} \cdot |w_{a,u}|^{\rho-1}$$

where ρ is the amplification factor, and $\rho \geq 1$.

Other notable extensions to similarity-based collaborative filtering include *weighted majority prediction* (Nakamura & Abe, 1998) and *imputation-boosted CF* (Su, Khoshgoftaar, Zhu, & Greiner, 2008).

Model-based Collaborative Filtering Model-based techniques provide recommendations by estimating parameters of statistical models for user ratings. For example, Billsus and Pazzani (1998) describe an early approach to map CF to a classification problem, and build a classifier for each active user representing items as features over users and available ratings as labels, possibly in conjunction with dimensionality reduction techniques to overcome data sparsity issues. Other predictive modeling techniques have also been applied in closely related ways.

More recently, ►latent factor and matrix factorization models have emerged as a state-of-the-art methodology in this class of techniques (Bell et al., 2009). Unlike neighborhood based methods that generate recommendations based on statistical notions of similarity between users, or between items, latent factor models assume that the similarity between users and items is simultaneously induced by some hidden lower-dimensional structure in the data. For example, the rating that a user gives to a movie might be assumed to depend on few implicit factors such as the user's taste across various movie genres. Matrix factorization techniques are a class of widely successful latent factor models where users and items are simultaneously represented as unknown feature vectors (column vectors) $w_u, h_i \in \Re^k$ along k latent dimensions. These feature vectors are learnt so that inner products $w_u^T h_i$ approximate the known preference ratings $r_{u,i}$ with respect to some loss measure. The squared loss is a standard choice for the loss function, in which case the following objective function is minimized,

$$J(W,H) = \sum_{(u,i)\in L} \left(r_{u,i} - w_u^T h_i\right)^2 \qquad (6)$$

where $W = [w_1 \ldots w_n]^T$ is an $n \times k$ matrix, $H = [h_1 \ldots h_m]$ is a $k \times m$ matrix, and L is the set of user-item pairs for which the ratings are known. In the impractical limit where all user-item ratings are known, the above objective function is $J(W,H) = \|R - WH\|_{fro}^2$ where R denotes the $n \times m$ fully known user-item matrix. The solution to this problem is given by taking the truncated SVD of R, $R = UDV^T$ and setting $W = U_k D_k^{\frac{1}{2}}, H = D_k^{\frac{1}{2}} V_k^T$ where U_k, D_k, V_k contain the k largest singular triplets of R. However, in the realistic setting where the majority of user-item ratings are unknown, such a nice globally optimal solution cannot be directly obtained, and one has to explicitly optimize the non-convex objective function $J(W,H)$. Note that in this case, the objective function is a particular form of weighted loss, that is, $J(W,H) = \|S \odot (R - WH)\|_{fro}^2$ where $\odot$ denotes elementwise products, and S is a binary matrix that equals one over known user-item pairs L, and 0 otherwise. Therefore, weighted low-rank approximations are pertinent to this discussion (Srebro & Jaakkola, 2003). Standard optimization procedures include gradient-based techniques, or procedures like alternating least squares where H is solved keeping W fixed and vice versa until a convergence criterion is satisfied. Note that fixing either W or H turns the problem of estimating the other into a weighted ►linear regression task. In order to avoid learning a model that overfits, it is common to minimize the objective function in the presence of ►regularization terms, $J(W,H) + \gamma\|W\|^2 + \lambda\|H\|^2$, where γ, λ are regularization parameters that can be determined by cross-validation. Once W, H are learnt, the product WH provides an approximate reconstruction of the rating matrix from where recommendations can be directly read off.

Different choices of loss functions, regularizers, and additional model constraints have generated a large body of literature on matrix factorization techniques. Arguably, for discrete ratings, the squared loss is not the most natural loss function. The maximum margin matrix factorization (Rennie & Srebro, 2005) approach uses margin-based loss functions such as the hinge loss used in ▶SVM classification, and its ordinal extensions for handling multiple ordered rating categories. For ratings that span over K values, this reduces to finding $K-1$ thresholds that divide the real line into consecutive intervals specifying rating bins to which the output is mapped, with a penalty for insufficient margin of separation. Rennie and Srebro (2005) suggest a nonlinear conjugate gradient algorithm to minimize a smoothed version of this objective function.

Another class of techniques is the nonnegative matrix factorization popularized by the work of Lee and Seung (1999) where nonnegativity constraints are imposed on W, H. There are weighted extensions of NMF that can be applied to recommendation problems. The rating behavior of each user may be viewed as being a manifestation of different roles, for example, a composition of prototypical behavior in clusters of users bound by interests or community. Thus, the ratings of each user are an additive sum of basis vectors of ratings in the item space. By disallowing subtractive basis, nonnegativity constraints lend a "part-based" interpretation to the model. NMF can be solved with a variety of loss functions, but with the generalized KL-divergence loss defined as follows,

$$J(W,H) = \sum_{u,i \in L} r_{u,i} \log \frac{r_{u,i}}{w_u^T h_i} - r_{u,i} + w_u^T h_i$$

NMF is in fact essentially equivalent to probabilistic latent semantic analysis (pLSA) which has also previously been used for collaborative filtering tasks (Hofmann, 2004).

The recently concluded million-dollar Netflix competition has catapulted matrix factorization techniques to the forefront of recommender technologies in collaborative filtering settings (Bell et al., 2009). While the final winning solution was a complex ensemble of different models, several enhancements to basic matrix factorization models were found to lead to improvements. These included:

1. The use of additional user-specific and item-specific parameters to account for systematic biases in the ratings such as popular movies receiving higher ratings on average.
2. Incorporating temporal dynamics of rating behavior by introducing time-dependent variables.

$$J(W,H) = \sum_{(u,i) \in L} \left(r_{u,i}(t) - b_u(t) - b_i(t) - \hat{r} - w_u^T(t) h_i\right)^2$$

where t denotes a time-stamp and W includes time-dependent user dimensions.

In many settings, only implicit preferences are available, as opposed to explicit like–dislike ratings. For example, large business organizations, typically, meticulously record transactional details of products purchased by their clients. This is a one-class setting since the business domain knowledge for negative examples – that a client has no interest in buying a product ever in the future – is typically not available explicitly in corporate databases. Moreover, such knowledge is difficult to gather and maintain in the first place, given the rapidly changing business environment. Another example is recommending TV shows based on watching habits of users, where preferences are implicit in what the users chose to see without any source of explicit ratings. Recently, matrix factorization techniques have been advanced to handle such problems (Pan & Scholz, 2009) by formulating confidence weighted objective function, $J(W,H) = \sum_{(u,i)} c_{u,i} \left(r_{u,i} - w_u^T h_i\right)^2$, under the assumption that unobserved user-item pairs may be taken as negative examples with a certain degree of confidence specified via $c_{u,i}$.

Content-based Recommending

Pure collaborative filtering recommenders only utilize the user ratings matrix, either directly, or to induce a collaborative model. These approaches treat all users and items as atomic units, where predictions are made without regard to the specifics of individual users or items. However, one can make a better personalized recommendation by knowing more about a user, such as demographic information (Pazzani, 1999), or about an item, such as the director and genre of a movie (Melville, Mooney, & Nagarajan, 2002). For instance, given movie

genre information, and knowing that a user liked "Star Wars" and "Blade Runner," one may infer a predilection for science fiction and could hence recommend "Twelve Monkeys." Content-based recommenders refer to such approaches, that provide recommendations by comparing representations of content describing an item to representations of content that interests the user. These approaches are sometimes also referred to as *content-based filtering*.

Much research in this area has focused on recommending items with associated *textual* content, such as web pages, books, and movies; where the web pages themselves or associated content like descriptions and user reviews are available. As such, several approaches have treated this problem as an information retrieval (IR) task, where the content associated with the user's preferences is treated as a query, and the unrated documents are scored with relevance/similarity to this query (Balabanovic & Shoham, 1997). In NewsWeeder (Lang, 1995), documents in each rating category are converted into *tf-idf* word vectors, and then averaged to get a prototype vector of each category for a user. To classify a new document, it is compared with each prototype vector and given a predicted rating based on the cosine similarity to each category.

An alternative to IR approaches, is to treat recommending as a classification task, where each example represents the content of an item, and a user's past ratings are used as labels for these examples. In the domain of book recommending, Mooney and Roy (2000) use text from fields such as the title, author, synopses, reviews, and subject terms, to train a multinomial ►naïve Bayes classifier. Ratings on a scale of 1 to k can be directly mapped to k classes (Melville et al., 2002), or alternatively, the numeric rating can be used to weight the training example in a probabilistic binary classification setting (Mooney & Roy, 2000). Other classification algorithms have also been used for purely content-based recommending, including ►k-nearest neighbor, ►decision trees, and ►neural networks (Pazzani & Billsus, 1997).

Hybrid Approaches

In order to leverage the strengths of content-based and collaborative recommenders, there have been several hybrid approaches proposed that combine the two. One simple approach is to allow both content-based and collaborative filtering methods to produce separate ranked lists of recommendations, and then merge their results to produce a final list (Cotter & Smyth, 2000). Claypool, Gokhale, and Miranda (1999) combine the two predictions using an adaptive weighted average, where the weight of the collaborative component increases as the number of users accessing an item increases.

Melville et al. (2002) proposed a general framework for *content-boosted collaborative filtering*, where content-based predictions are applied to convert a sparse user ratings matrix into a full ratings matrix, and then a CF method is used to provide recommendations. In particular, they use a Naïve Bayes classifier trained on documents describing the rated items of each user, and replace the unrated items by predictions from this classifier. They use the resulting *pseudo ratings matrix* to find neighbors similar to the active user, and produce predictions using Pearson correlation, appropriately weighted to account for the overlap of actually rated items, and for the active user's content predictions. This approach has been shown to perform better than pure collaborative filtering, pure content-based systems, and a linear combination of the two. Within this content-boosted CF framework, Su, Greiner, Khoshgoftaar, and Zhu (2007) demonstrated improved results using a stronger content-predictor, TAN-ELR, and unweighted Pearson collaborative filtering.

Several other hybrid approaches are based on traditional collaborative filtering, but also maintain a content-based profile for each user. These content-based profiles, rather than co-rated items, are used to find similar users. In Pazzani's approach (Pazzani, 1999), each user-profile is represented by a vector of weighted words derived from positive training examples using the Winnow algorithm. Predictions are made by applying CF directly to the matrix of user-profiles (as opposed to the user-ratings matrix). An alternative approach, Fab (Balabanovic & Shoham, 1997), uses relevance feedback to simultaneously mold a personal filter along with a communal "topic" filter. Documents are initially ranked by the topic filter and then sent to a user's personal filter. The user's relevance feedback is used to modify both the personal filter and the originating topic filter. Good et al. (1999) use collaborative filtering along with a number of personalized information filtering agents. Predictions for a user are made by

applying CF on the set of other users and the active user's personalized agents.

Several hybrid approaches treat recommending as a classification task, and incorporate collaborative elements in this task. Basu, Hirsh, and Cohen (1998) use *Ripper*, a ►rule induction system, to learn a function that takes a user and movie and predicts whether the movie will be liked or disliked. They combine collaborative and content information, by creating features such as *comedies liked by user* and *users who liked movies of genre X*. In other work, Soboroff and Nicholas (1999) multiply a *term-document matrix* representing all item content with the user-ratings matrix to produce a *content-profile matrix*. Using latent semantic Indexing, a rank-k approximation of the content-profile matrix is computed. Term vectors of the user's relevant documents are averaged to produce a user's profile. Then, new documents are ranked against each user's profile in the LSI space.

Some hybrid approaches attempt to directly combine content and collaborative data under a single probabilistic framework. Popescul, Ungar, Pennock, and Lawrence (2001) extended Hofmann's *aspect model* (Hofmann, 1999) to incorporate a three-way co-occurrence data among users, items, and item content. Their generative model assumes that users select latent topics, and documents and their content words are generated from these topics. Schein, Popescul, Ungar, and Pennock (2002) extend this approach, and focus on making recommendations for items that have not been rated by any user.

Evaluation Metrics

The quality of a recommender system can be evaluated by comparing recommendations to a test set of known user ratings. These systems are typical measured using *predictive accuracy metrics* (Herlocker, Konstan, Terveen, & Riedl, 2004), where the predicted ratings are directly compared to actual user ratings. The most commonly used metric in the literature is ►*Mean Absolute Error* (MAE) – defined as the average absolute difference between predicted ratings and actual ratings, given by:

$$MAE = \frac{\sum_{\{u,i\}} |p_{u,i} - r_{u,i}|}{N} \quad (7)$$

Where $p_{u,i}$ is the predicted rating for user u on item i, $r_{u,i}$ is the actual rating, and N is the total number of ratings in the test set.

A related commonly used metric, ►*Root Mean Squared Error* (RMSE), puts more emphasis on larger absolute errors, and is given by:

$$RMSE = \sqrt{\frac{\sum_{\{u,i\}} (p_{u,i} - r_{u,i})^2}{N}} \quad (8)$$

Predictive accuracy metrics treat all items equally. However, for most recommender systems the primary concern is accurately predict the items a user will like. As such, researchers often view recommending as predicting *good*, that is, items with high ratings versus *bad* or poorly rated items. In the context of information retrieval (IR), identifying the good from the background of bad items can be viewed as discriminating between "relevant" and "irrelevant" items; and as such, standard IR measures, like ►*Precision*, ►*Recall* and ►*Area Under the ROC Curve (AUC)* can be utilized. These, and several other measures, such as *F1-measure*, *Pearson's product-moment correlation*, *Kendall's* τ, *mean average precision*, *half-life utility*, and *normalized distance-based performance measure* are discussed in more detail by Herlocker et al. (2004).

Challenges and Limitations

This section, presents some of the common hurdles in deploying recommender systems, as well as some research directions that address them.

Sparsity: Stated simply, most users do not rate most items and, hence, the user ratings matrix is typically very sparse. This is a problem for collaborative filtering systems, since it decreases the probability of finding a set of users with similar ratings. This problem often occurs when a system has a very high item-to-user ratio, or the system is in the initial stages of use. This issue can be mitigated by using additional domain information (Melville et al., 2002; Su et al., 2007) or making assumptions about the data generation process that allows for high-quality imputation (Su et al., 2008).

The Cold-Start Problem: New items and new users pose a significant challenge to recommender systems. Collectively these problems are referred to as the *cold-start problem* (Schein et al., 2002). The first of these problems arises in collaborative filtering systems, where

an item cannot be recommended unless some user has rated it before. This issue applies not only to new items, but also to obscure items, which is particularly detrimental to users with eclectic tastes. As such the *new-item problem* is also often referred to as the *first-rater problem*. Since content-based approaches (Mooney & Roy, 2000; Pazzani & Billsus, 1997) do not rely on ratings from other users, they can be used to produce recommendations for *all* items, provided attributes of the items are available. In fact, the content-based predictions of similar users can also be used to further improve predictions for the active user (Melville et al., 2002).

The *new-user problem* is difficult to tackle, since without previous preferences of a user it is not possible to find similar users or to build a content-based profile. As such, research in this area has primarily focused on effectively selecting items to be rated by a user so as to rapidly improve recommendation performance with the least user feedback. In this setting, classical techniques from ►*active learning* can be leveraged to address the task of item selection (Harpale & Yang, 2008; Jin & Si, 2004).

Fraud: As recommender systems are being increasingly adopted by commercial websites, they have started to play a significant role in affecting the profitability of sellers. This has led to many unscrupulous vendors engaging in different forms of fraud to game recommender systems for their benefit. Typically, they attempt to inflate the perceived desirability of their own products (*push attacks*) or lower the ratings of their competitors (*nuke attacks*). These types of attack have been broadly studied as *shilling attacks* (Lam & Riedl, 2004) or *profile injection attacks* (Burke, Mobasher, Bhaumik, & Williams, 2005). Such attacks usually involve setting up dummy profiles, and assume different amounts of knowledge about the system. For instance, the *average attack* (Lam & Riedl, 2004) assumes knowledge of the average rating for each item; and the attacker assigns values randomly distributed around this average, along with a high rating for the item being *pushed*. Studies have shown that such attacks can be quite detrimental to predicted ratings, though *item-based* collaborative filtering tends to be more robust to these attacks (Lam & Riedl, 2004). Obviously, content-based methods, which only rely on a users past ratings, are unaffected by profile injection attacks.

While pure content-based methods avoid some of the pitfalls discussed above, collaborative filtering still has some key advantages over them. Firstly, CF can perform in domains where there is not much content associated with items, or where the content is difficult for a computer to analyze, such as ideas, opinions, etc. Secondly, a CF system has the ability to provide serendipitous recommendations, that is, it can recommend items that are relevant to the user, but do not contain content from the user's profile.

Further Reading

Good surveys of the literature in the field can be found in Adomavicius and Tuzhilin (2005); Bell et al. (2009); Su and Khoshgoftaar (2009). For extensive empirical comparisons on variations of Collaborative Filtering refer to Breese (1998), Herlocker (1999), Sarwar et al. (2001).

Recommended Reading

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering, 17*(6), 734–749.

Balabanovic, M., & Shoham, Y. (1997). Fab: Content-based, collaborative recommendation. *Communications of the Association for Computing Machinery, 40*(3), 66–72.

Basu, C., Hirsh, H., & Cohen, W. (July 1998). Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the fifteenth national conference on artificial intelligence (AAAI-98), Madison, Wisconsin* (pp. 714–720).

Bell, R., Koren, Y., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *IEEE Computer 42*(8):30–37.

Billsus, D., & Pazzani, M. J. (1998). Learning collaborative information filters. In *Proceedings of the fifteenth international conference on machine learning (ICML-98), Madison, Wisconsin* (pp. 46–54). San Francisco: Morgan Kaufmann.

Breese, J. S., Heckerman, D., & Kadie, C. (July 1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the fourteenth conference on uncertainty in artificial intelligence, Madison, Wisconsin*.

Burke, R., Mobasher, B., Bhaumik, R., & Williams, C. (2005). Segment-based injection attacks against collaborative filtering recommender systems. In *ICDM '05: Proceedings of the fifth IEEE international conference on data mining* (pp. 577–580). Washington, DC: IEEE Computer Society. Houston, Texas

Claypool, M., Gokhale, A., & Miranda, T. (1999). Combining content-based and collaborative filters in an online newspaper. In *Proceedings of the SIGIR-99 workshop on recommender systems: algorithms and evaluation*.

Cotter, P., & Smyth, B. (2000). PTV: Intelligent personalized TV guides. In *Twelfth conference on innovative applications of artificial intelligence, Austin, Texas* (pp. 957–964).

Goldberg, D., Nichols, D., Oki, B., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the Association of Computing Machinery, 35*(12), 61–70.

Good, N., Schafer, J. B., Konstan, J. A., Borchers, A., Sarwar, B., Herlocker, J., et al. (July 1999). Combining collaborative filtering with personal agents for better recommendations. In *Proceedings of the sixteenth national conference on artificial intelligence (AAAI-99), Orlando, Florida* (pp. 439–446).

Harpale, A. S., & Yang, Y. (2008). Personalized active learning for collaborative filtering. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval, Singapore* (pp. 91–98). New York: ACM.

Herlocker, J., Konstan, J., Borchers, A., & Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proceedings of 22nd international ACM SIGIR conference on research and development in information retrieval, Berkeley, California* (pp. 230–237). New York: ACM.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems, 22*(1), 5–53.

Hofmann, T. (1999). Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, Stockholm, Sweden, July 30-August 1, 1999 Morgan Kaufmann

Hofmann, T. (2004). Latent semantic analysis for collaborative filtering. *ACM Transactions on Information Systems, 22*(1), 89–115.

Jin, R., & Si, L. (2004). A Bayesian approach toward active learning for collaborative filtering. In *UAI '04: Proceedings of the 20th conference on uncertainty in artificial intelligence, Banff, Canada* (pp. 278–285). Arlington: AUAI Press.

Lam, S. K., & Riedl, J. (2004). Shilling recommender systems for fun and profit. In *WWW '04: Proceedings of the 13th international conference on World Wide Web, New York* (pp. 393–402). New York: ACM.

Lang, K. (1995). NewsWeeder: Learning to filter netnews. In *Proceedings of the twelfth international conference on machine learning (ICML-95)* (pp. 331–339). San Francisco. Tahoe City, CA, USA Morgan Kaufmann, ISBN 1-55860-377-8

Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature, 401*, 788.

Linden, G., Smith, B., & York, J. (2003). Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing, 7*(1), 76–80.

Melville, P., Mooney, R. J., & Nagarajan, R. (2002). Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the eighteenth national conference on artificial intelligence (AAAI-02), Edmonton, Alberta* (pp. 187–192).

Mooney, R. J., & Roy, L. (June 2000). Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on digital libraries, San Antonio, Texas* (pp. 195–204).

Nakamura, A., & Abe, N. (1998). Collaborative filtering using weighted majority prediction algorithms. In *ICML '98: Proceedings of the fifteenth international conference on machine learning* (pp. 395–403). San Francisco: Morgan Kaufmann. Madison, Wisconsin

Pan, R., & Scholz, M. (2009). Mind the gaps: Weighting the unknown in large-scale one-class collaborative filtering. In *15th ACM SIGKDD conference on knowledge discovery and data mining (KDD), Paris, France.*

Pazzani, M. J. (1999). A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review, 13* (5–6), 393–408.

Pazzani, M. J., & Billsus, D. (1997). Learning and revising user profiles: The identification of interesting web sites. *Machine Learning, 27*(3), 313–331.

Popescul, A., Ungar, L., Pennock, D. M., & Lawrence, S. (2001). Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of the seventeenth conference on uncertainity in artificial intelligence*. University of Washington, Seattle

Rennie, J., & Srebro, N. (2005). Fast maximum margin matrix factorization for collaborative prediction. In *International conference on machine learning, Bonn, Germany*.

Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P., & Reidl, J. (1994a). GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 computer supported cooperative work conference, New York*. New York: ACM. ACM Press New York, NY, USA

Resnick, P., Neophytos, I., Bergstrom, P., Mitesh, S., & Riedl, J. (1994b). Grouplens: An open architecture for collaborative filtering of netnews. In *CSCW94 – Conference on computer supported cooperative work, Chapel Hill* (pp. 175–186). Addison-Wesley.

Sarwar, B., Karypis, G., Konstan, J., & Reidl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the tenth international conference on World Wide Web* (pp. 285–295). New York: ACM. Hong Kong

Schein, A. I., Popescul, A., Ungar, L. H., & Pennock, D. M. (2002). Methods and metrics for cold-start recommendations. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 253–260). New York: ACM. Tampere, Finland

Soboroff, I., & Nicholas, C. (1999). Combining content and collaboration in text filtering. In T. Joachims (Ed.), *Proceedings of the IJCAI'99 workshop on machine learning in information filtering* (pp. 86–91).

Srebro, N., & Jaakkola, T. (2003). Weighted low-rank approximations. In *International conference on machine learning (ICML)*. Washington DC

Su, X., Greiner, R., Khoshgoftaar, T. M., & Zhu, X. (2007). Hybrid collaborative filtering algorithms using a mixture of experts. In *Web intelligence* (pp. 645–649).

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence, 2009*, 1–20.

Su, X., Khoshgoftaar, T. M., Zhu, X., & Greiner, R. (2008). Imputation-boosted collaborative filtering using machine learning classifiers. In *SAC '08: Proceedings of the 2008 ACM symposium on applied computing* (pp. 949–950). New York: ACM.