ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/329398751

User Preference Elicitation, Rating Sparsity and Cold Start: Algorithms, Practical Challenges and Applications

Chapter · November 2018

DOI: 10.1142/9789813275355_0008

CITATIONS
11

READS
432

4 authors:

Mehdi Elahi
University of Bergen
85 PUBLICATIONS **1,490** CITATIONS
SEE PROFILE

Matthias Braunhofer
Microsoft
28 PUBLICATIONS **628** CITATIONS
SEE PROFILE

Tural Gurbanov
ivi.ru
9 PUBLICATIONS **47** CITATIONS
SEE PROFILE

Francesco Ricci
Free University of Bozen-Bolzano
284 PUBLICATIONS **12,743** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Music recommender systems View project

Create new project "ONE Project" View project

All content following this page was uploaded by Mehdi Elahi on 21 March 2019.

The user has requested enhancement of the downloaded file.

Chapter 1

User preference elicitation, rating sparsity and cold start

Mehdi Elahi, Matthias Braunhofer, Tural Gurbanov, Francesco Ricci
Free University of Bozen - Bolzano,
Piazza Domenicani, 3
Italy - 39100, Bozen-Bolzano

{*meelahi,mbraunhofer,tgurbanov,fricci*}*@unibz.it*

1.1 Abstract

A prerequisite for implementing collaborative filtering recommender systems is the availability of users' preferences data. This data, typically in the form of ratings, is exploited to learn the tastes of the users and to serve them with personalized recommendations. However, there may be a lack of preference data, especially at the initial stage of the operations of a recommender system, i.e., in the *Cold Start* phase. In particular, when a new user has not yet rated any item, the system would be incapable of generating relevant recommendations for this user. Or, when a new item is added to the system catalogue and no user has rated it, the system cannot recommend this item to any user.

This chapter discusses the cold start problem and provides a comprehensive description of techniques that have been proposed to address this problem. It surveys algorithmic solutions and provides a summary of their performance comparison. Moreover, it lists publicly available resources (e.g., libraries and datasets) and offers a set of practical guidelines that can be adopted by researchers and practitioners.

1.2 Introduction

The research on Recommender Systems (RSs) has seen a continuous growth, starting from the mid-90s when early works have been published [Resnick *et al.* (1994); Shardanand and Maes (1995)]. One of the most popular approaches in recommender systems is Collaborative Filtering (CF), which is the main focus of this chapter. In collaborative filtering a set of preferences provided by a community of users, the relationships and similarities among the user preferences, are exploited to generate personalized recommendations for a target user. A variety of collaborative filtering techniques, which are typically classified as user-based [Herlocker *et al.* (1999)], item-based [Linden *et al.* (2003)], or Matrix Factorization (MF) [Koren and Bell (2011)], have been proposed and evaluated in both industry and academia [Koren and Bell (2015); Shi *et al.* (2014)].

Moreover, various types of preference data can be exploited by collaborative filtering recommender systems, and they can roughly divided in two wide categories: *Explicit Feedback* and *Implicit Feedback* (see chapter **??**). Explicit feedback, which is typically considered as a more informative signal of user's preference, refers to item evaluations that the user explicitly reports, e.g., five star ratings for movies in Netflix, or like/dislikes for posts in Facebook [Koren and Bell (2015); Stern *et al.* (2009); Agarwal and Chen (2009)]. Despite their usefulness, eliciting explicit feedback requires some user effort [Elahi *et al.* (2014)] and still might not completely reveal actual users' needs [Neidhardt *et al.* (2014)].

Hence, implicit feedback, i.e., actions performed by the users on the items, such as viewing an item, have been used to infer preference information [Oard *et al.* (1998); Hu *et al.* (2008a); Gurbanov and Ricci (2017)]. This data is much more abundant and simpler to collect than explicit feedback. For example, the purchase or browsing history of a user in Amazon.com can be used by the system to predict additional interests of the user and ultimately generate recommendations for her. A user who frequently purchases books of a specific author will likely be interested in that author.

It is worth noting that despite the clear usefulness of the implicit feedback, the usage of this type of data has some limitations: for instance, it is easier to infer a "positive" feedback from a user action rather than a "negative" one. Most importantly, the system can only guess the actual user interests from the tracked behaviour of a user. In fact, purchasing an item may not necessarily indicate that the user was satisfied about it, as the user may have regret purchasing that item. But, repeated actions of the

users on items may increase the confidence in such an inferred preference [Hu *et al.* (2008a)].

Preference data can be also classified into *Absolute* or *Relative* . The former type of preference is the most popular; these are expressed in the form of absolute evaluations in a predefined scale, e.g., the five star ratings used by Netflix.com [Linden *et al.* (2003); Sarwar *et al.* (2001); Koren and Bell (2015)]. However, this type of preference data have limitations. For example, a user who loves two items, but prefers one over another, might be pushed to rate them alike, or to penalize one, due to the limited number of points in the rating scale. This, and other problems, could be resolved by eliciting relative preferences, i.e., pairwise comparisons: "I prefer Blade Runner to Indiana Jones". One can also express how much an item is preferred (or considered inferior) to another, hence generating pair scores (positive or negative). The larger the score the more one item is preferred to the other item in the pair [Kalloori *et al.* (2016); Blédaité and Ricci (2015); Jones *et al.* (2011); Rendle *et al.* (2009)].

Regardless of the type of the data used, collaborative filtering recommender systems commonly suffer from a challenging problem, i.e., *Cold Start*. This problem occurs when the system has not yet acquired sufficient preference data (e.g., ratings or pairwise scores) to generate relevant recommendations for users. The most common example of the cold start problem occurs when the system is not capable of properly recommending any item to a new user (*New User* problem) or recommending a new item (not yet evaluated by any user) to a user (*New Item* problem) [Adomavicius and Tuzhilin (2005); Schein *et al.* (2002)]. In extreme cases, both problems may take place, e.g., when a recommender system has been recently launched and the system database contains very limited or no information about the preferences of users on items [Su and Khoshgoftaar (2009)].

In addition to these problems, preference data *Sparsity* may be observed. This occurs when the system has only collected a small percentage of all the potential preferences, which, for instance, in a collaborative filtering system is the full set of ratings of all the users for all the items. In fact, even in a mature system the users have rated a small percentage of the available items. Sparsity becomes a significant problem when the number of available ratings of each user is extremely smaller than the overall number of items [Adomavicius and Tuzhilin (2005)]. In general, preference data sparsity makes it very challenging for the recommender to generate accurate recommendations.

The cold start problem is even more challenging in Context-Aware Rec-

ommender Systems , known as CARSs (see chapter **??**). These systems generate recommendations which are adapted not only to the user's preferences but also to the contextual situation [Adomavicius and Tuzhilin (2011)]. Here, there is a cold start problem when the system is requested to generate recommendations for contextual situations under which the observed users did not express preferences (*New Context* problem) [Braunhofer (2015)]. Indeed, CARS needs to collect preference data that are augmented with the indication of the contextual situation of the user when experiencing the evaluated item. For that reason, in CARS, it is not only important to have ratings, but also to acquire them in several different contextual situations, so that the system can learn the users' contextually dependent preferences. Hence, the cold start problem in CARS occurs more frequently, compared to traditional recommender systems; there is often a large number of possible alternative contextual situations that may be observed in realistic scenarios [Braunhofer *et al.* (2014)].

Various approaches for addressing the cold start problem and improving collaborative filtering have been proposed in the literature. Here we briefly introduce them, and in the other sections of this chapter, we discuss them in details.

A popular approach to cope with the cold start problem consists of implementing *hybrid* recommendation techniques (see chapter **??**), e.g., to combine collaborative and content-based filtering [Nicholas and Nicholas (1999); Burke (2002); Ge *et al.* (2015); Adomavicius and Tuzhilin (2005); Ricci *et al.* (2015); Vartak *et al.* (2017); Kim *et al.* (2016)]. It is also possible to address the cold-start problem with cross-domain recommendation techniques. These techniques aim at improving the recommendations in a target domain by making use of information about the user preferences in an auxiliary domain [Fernández-Tobías *et al.* (2016); Cantador and Cremonesi (2014)]. In this case, knowledge of the preferences of the user is transferred from an auxiliary domain to the target domain. More complete and accurate user models and item recommendations in the target domain can then be built even when not much preference data is available in this domain. For example, by using the knowledge of ratings and tags assigned by the users to items in a movie domain (auxiliary) it is possible to better learn the user preferences in a book domain (target) [Enrich *et al.* (2013)].

Another approach to cold-start consists of complementing the rating data with other sources of information about the items. For example, in multimedia recommender systems, it has been shown that audio-visual features, e.g., variation of colour, camera and object motion, and lighting,

can be automatically extracted from movies in order to solve the cold start problem and to effectively generate relevant recommendations [Elahi *et al.* (2017); Deldjoo *et al.* (2016, 2018)]. As an additional example, in a food recommender, it has been shown that one can exploit the information brought by tags assignments of the users to recipes to improve the recommendation performance of a system that is using only ratings for recipes [Ge *et al.* (2015); Massimo *et al.* (2017)]. Or, in a restaurant RS, it is possible to use information about the restaurant cuisine or location [Burke (2000); Adomavicius and Tuzhilin (2005); Ricci *et al.* (2015)].

In an alternative group of approaches the cold start problem is tackled by better profiling the users with additional information about them, such as their personality [Pu *et al.* (2012)]. In fact, studies conducted on user personality characteristics have shown that it is useful to exploit this information in collaborative filtering [Hu and Pu (2011, 2009); Tkalcic *et al.* (2013); Schedl *et al.* (2018); Elahi *et al.* (2013)]. Another example in this group of approaches, is discussed in [Trevisiol *et al.* (2014)]. The authors propose a graph-based method to solve the cold start problem in news recommendation. This method uses referral link of the new user as well as the currently visited page in order to generate recommendations for new users.

The cold-start problem can also attacked by directly enlarge the size of the preference data set by programming the system to selectively acquire new users' preferences with *Active Learning* [Elahi *et al.* (2016)]. In fact, Active learning tackles the cold start problem at the root, by identifying high quality data that better represents a user's preferences and improves the performance of the system. This can be done in various forms, e.g., by requesting the user to assess items one-by-one [Rashid *et al.* (2008a); Elahi *et al.* (2014)] or alternatively to evaluate a set of them altogether (Ekstrand *et al.*, 2014; Loepp *et al.*, 2014).

In one seminal paper on this subject [Rashid *et al.* (2002)], the authors present several active learning techniques that have been tested on the well-known movie recommender system MovieLens. The goal was to create an effective sign up procedure that can help the system to build the initial user profile by acquiring specific ratings. Hence, when a new user registers to the system, the user is requested to rate movie items that are estimated to be more informative. Experimental results have shown the effectiveness of the approach.

Active learning has been used in a Points of Interest (POI) CARS named South Tyrol Suggests (STS) [Braunhofer (2015)]. In the early stages of the deployment, STS was in an extreme cold-start situation where only a few

hundred of contextual ratings were provided by users for nearly 27,000 POIs stored in the system database. To solve this problem, the system acquired the users' personality to identify which items they could have visited and hence requested to provide their contextual ratings for those items [Elahi *et al.* (2013)].

In addition to the above-mentioned methods, it is also possible to use stereotypes as a mechanism for generating recommendations for users in the cold start situation. In this case, the system recommends items that matches a generic profile of that user, built upon the available data for a group the target user belongs to [Adomavicius and Tuzhilin (2005)]. For example, the systems attempts to recommend items that are popular within a certain age group or gender [Braunhofer *et al.* (2015a)]. In the extreme cold start scenario, if absolutely no data is available, the system may recommend popular items or items with the highest average ratings. Although the recommendations are non-personalized, still a satisfactory level of recommendation quality might be achieved [Fernandez Tobias *et al.* (2016)].

The rest of the chapter is structured as the following. Section 1.3 reviews the algorithmic solutions to tackle with the cold start and sparsity problem. Section 1.4 introduces the available resources such as libraries and datasets applicable in the research on cold start. Section 1.5 compares the discussed solutions and provides an overview of their performances. In section 1.6, a set of practical guidelines are provided that can be used by researchers and practitioners. Finally, section 1.7 concludes the chapter and discusses the directions of the future works.

1.3 Algorithmic solutions

As we have discussed in the introduction, various and different approaches to alleviate the cold-start problem have been proposed in the literature. We present them by grouping them into two broad classes according to which kind of knowledge is mainly used. The approaches in the first and more important class exploit in the recommendation process additional knowledge sources about either the users, the items or the contextual situations. The five approaches identified in this class are: active learning, cross-domain recommendation, recommendation based on implicit feedback, content-based recommendation and demographic-based recommendation. On the other hand, the approaches in the second class try to better process and leverage existing knowledge rather than acquiring new one.

1.3.1 *Active Learning*

Active learning (AL) applied to recommender systems refers to technologies aimed at eliciting the most informative preference data. When the system is using ratings, AL tries to elicit from the user those ratings that best reveal the user's interests, and hence should better improve the quality of subsequent recommendations [Rubens *et al.* (2015)]. [Rashid *et al.* (2002)] discusses six different AL rating elicitation strategies:

Entropy where items with the largest rating entropy are asked to be rated by the user;

Random which randomly selects items to be rated;

Popularity which measures the number of already acquired ratings for the items, and requests the user to rate the most frequently rated ones;

Popularity * Entropy where items that are both popular and have diverse ratings are requested to be rated by the user;

Log(popularity) * Entropy which is similar to the previous strategy except that it considers the log of the number of ratings before computing popularity;

Item-Item Personalized where the items are selected randomly until a first rating is acquired. Then a recommender is used to predict the items that the user is likely to have seen; these items are requested to the user to rate.

The results of off line and on line experiments conducted on the MovieLens dataset demonstrated that *log(popularity) * entropy* improves more the performance of the recommender system in terms of rating prediction accuracy.

In [Rashid *et al.* (2008a)], the authors extend their early work [Rashid *et al.* (2002)] by proposing additional AL strategies, namely:

Entropy0 which differs from the entropy strategy, as described above, in that it treats missing ratings as a separate category, in addition to the acquired ratings on the 1-5 scale;

HELF Harmonic mean of Entropy and Logarithm of Frequency selects the items with the largest harmonic mean of the logarithm of the rating frequency and the entropy;

IGCN Information Gain through Clustered Neighbours constructs a decision tree where the nodes are the items to rate and a node's branches according to the different ratings that a user can give to

the item.

The authors evaluated these strategies in a preliminary off line experiment. Then, in an on line experiment, they used the MovieLens web-based movie recommender and every new user, during the sign-up process, was asked to rate 20 movies selected by one of the AL strategies (training set). After the sign-up process the new users were asked to complete a user satisfaction survey and to provide some more movie ratings, which were used to check the prediction accuracy (testing set). Based on the obtained results, the authors concluded that, overall, IGCN and entropy0 are the best-performing strategies.

Additional AL strategies can be found in [Elahi *et al.* (2014, 2011)], including:

Binary Prediction which first transforms the original rating matrix into a binary matrix, by mapping null entries to 0 and not null entries to 1, and then learns a predictive model that identifies the items the user is likely to have experienced and thus is able to rate;

Highest Predicted where the items with the highest predicted ratings are asked to be rated by the user;

Lowest Predicted which, differently from Highest Predicted, requests the user to rate the items with the lowest predicted ratings;

Highest and Lowest Predicted where items with either extreme low or extreme high predicted ratings are selected;

Voting which combines together the lists of top candidate items for rating elicitation from different strategies to produce a single list.

The authors have evaluated the considered strategies for their system-wide effectiveness implementing a simulation loop that models the gradual process of rating elicitation and rating dataset growth. The procedure is initiated by splitting the matrix of ratings into three different matrices with the equal number of rows and columns [Elahi *et al.* (2014)]:

- K: contains the ratings which are known to the recommender system at a certain point of time;
- X: contains the ratings which are known by the users but not by the recommender system. These ratings are iteratively acquired, i.e., they are transferred into K if the recommender system requests the (simulated) users to rate them;
- T: contains a subset of the ratings that are known by the users but

are withheld from X for the evaluation purpose.

In every *iteration* of the experiment, the system carefully selects a set of items from the item catalog, according to a specific active learning strategy. The selected items are checked to find those with ratings available in X. The ratings of these items are assigned to the corresponding entries in K. Then the assigned ratings are removed from the X. Finally, the evaluation metrics are measured on the ratings in T, and the prediction model of the system is trained again using the new set of ratings in K. This process is repeated for 170 iterations, till almost all the ratings are elicited and the system performance gets stabilized.

Figure 1.1 illustrates the comparison of the rating prediction accuracy (in terms of Mean Absolute Error) of the described active learning strategies on MovieLens dataset. As it can be seen, there are two separable groups of active learning strategies [Elahi *et al.* (2014)]:

(1) Monotone error reduction strategies which include lowest-highest predicted, lowest predicted, voting and random strategies.
(2) Non-monotone error reduction strategies that include binary predicted, highest predicted, popularity, log(popularity)*entropy and variance strategies.

The strategies in the first group have a better performance, specially in the middle phase of the rating elicitation process. However, at the very beginning and at the end, the strategies in the second group outperform the first ones. Indeed, at the very beginning, the binary-predicted and the voting strategies (both from the first group) perform the best. Then, the random and the lowest-highest-predicted strategies (both from the second group) have good performance. Finally, at the ending phase, the MAE stabilizes for most of the strategies since they are not able to elicit anymore ratings.

The non-monotone behavior of the strategies in the second group occurs since they have a strong selection bias. the highest predicted strategy is perhaps the best known since it is the default strategy in recommender systems (people typically provide ratings to the recommendations). At the beginning of the rating elicitation process, this strategy elicits primarily items that have received high ratings. This leads to acquire significantly more high ratings than low ratings and ultimately biasing the recommender towards overestimating the predicted ratings. This is the reason why the error increases in the middle stage and drops afterward, as the items with

high ratings are finished and the strategy begins to select also items with lower ratings.

Overall, all the conducted experiments have shown that each strategy has its own strengths and weaknesses. Prediction-based strategies are ineffective to deal with new users or new items, whereas voting, popularity and log(popularity) * entropy can select items for new users, though not for new items. Moreover, some strategies, such as, highest predicted and lowest predicted may bring a system-wide bias and increase the system error as they try to add only ratings with certain values.

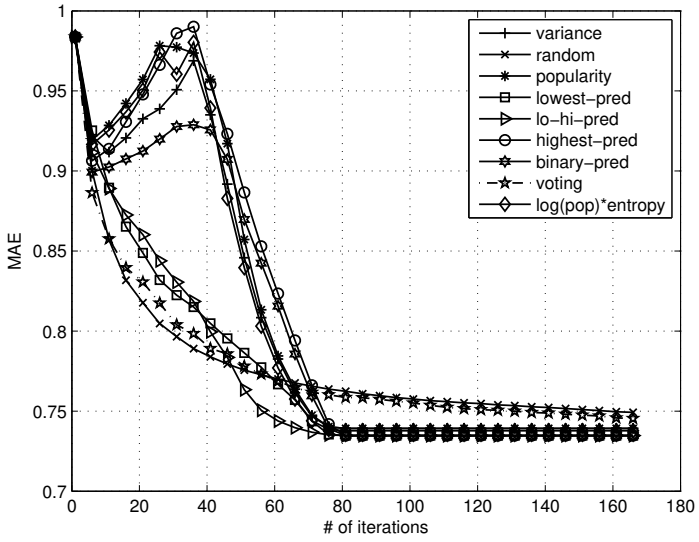

Fig. 1.1: Performance comparison of some well known Active Learning strategies in terms of prediction accuracy. [Elahi *et al.* (2014)]

It is worth noting that all the aforementioned AL strategies have been designed to work on traditional, two-dimensional user-item rating matrices, hence they are only suitable for attacking the new user and new item problems. In case of multidimensional user-item-context matrices, it is necessary to modify these strategies so that they output not only a list of items to be rated but also the contextual factors which characterize the situation of

the items' consumption. Identifying and acquiring exactly those contextual information that matter ensures that (i) the user effort in specifying contextual information is kept to a minimum, and (ii) the system's performance is not negatively impacted by irrelevant information.

A solution to that problem can be found in [Baltrunas *et al.* (2012)], where the authors present a survey-based approach to identify the most useful contextual factors and conditions, as well as, to capture data about how the context influences user ratings. In their approach, they first estimated the dependency of the user preferences from an initial candidate set of contextual factors. This was achieved through a web tool, in which users were requested to evaluate if a particular contextual condition (e.g.,"it is a cold day") has a positive, negative or no influence on the user's rating of a particular type of POI (e.g., spa). Using the obtained data, they were able to establish the most important contextual factors for different types of POI.

The same problem mentioned above, i.e., identifying the contextual factors that are truly relevant for a particular recommender system, was addressed by Odić et al. [Odić *et al.* (2012)]. They developed two approaches: the first one is called "assessment" and it is based on surveying the users, while the second is called "detection" of the context relevance and is performed by mining the rating data. These two approaches are very different in terms of when each one could be used (e.g., assessment can be used before rating acquisition, whereas detection cannot), what information is needed (e.g., detection requires a substantial number of in-context ratings, while assessment does not) and whether they rely on real situations (as in detection) or hypothetical situations (as in assessment). In order to determine which of the two is better, the authors used real rating data and a survey data set to construct two (possibly different) lists of relevant and irrelevant contextual factors. Then, they considered all the contextual information one by one as input to a contextualized matrix factorization model, and finally counted for both approaches the number of times a contextual factor estimated as relevant led the system to obtain a higher prediction accuracy than using a factor estimated as irrelevant. Based on the obtained results, they concluded that the detection method performs better than the assessment one for identifying the contextual factors that should be exploited in the rating prediction model.

In a related paper [Odić *et al.* (2013)], the same authors investigate in more detail the "detection" approach and provide several statistical measures for relevant-context detection, i.e., unalikeability, entropy, variance,

χ^2 test and Freeman-Halton test. Among these measures, they identify the Freeman-Halton test as the most useful and flexible measure to distinguish relevant from irrelevant contextual factors. Moreover, the authors show that the rating prediction performance was significantly better when using contextual factors detected as relevant than when using contextual factors detected as irrelevant.

Another example of selecting the most relevant contextual factors can be found in [Vargas-Govea *et al.* (2011)]. In this paper, the authors focus on a context-aware recommender system for restaurants, and show that its efficiency and predictive accuracy can be improved by using a reduced subset of contextual factors. To select contextual factors, the Las Vegas Filter (LVF) algorithm was chosen. LVF repeatedly generates random subsets of factors, computes their evaluation measure based on an inconsistency criterion, which tests the extent to which a reduced subset can still predict the rating values, and finally returns the subset yielding the best evaluation measure.

Finally, in [Braunhofer *et al.* (2015b); Braunhofer and Ricci (2016)], a context acquisition algorithm is proposed, that given a user-item pair, parsimoniously and adaptively identifies the most useful contextual factors, i.e., those that when elicited together with the user ratings improve more the quality of future recommendations, both for that user and for other users of the system. "Parsimonious" means that it selectively requests and possibly elicits only the most relevant contextual factors, whereas "adaptive" means that it personalizes the selection of the most relevant contextual factors to each individual user and item.

1.3.2 *Cross-Domain Recommendation*

Another option for collecting preference data in the form of ratings comes from collecting them in an auxiliary and possibly better known domain with the plan to transfer the knowledge brought by this preference data to the target domain [Cremonesi *et al.* (2011)]. Cross-domain approaches are about that, and are classified into two classes: those that aggregate knowledge from various auxiliary domains to perform recommendations in a target domain, and those that link and transfer knowledge between domains to support recommendations.

An example of knowledge aggregation for cross-domain recommendation is described in [Berkovsky *et al.* (2007)]. Here, the authors present four aggregation-based methods: (i) centralized prediction – the aggregated

knowledge consists of user preferences (i.e., ratings), (ii) distributed peer identification – the aggregated knowledge is composed of user similarities, (iii) distributed neighbourhood formation – the aggregated knowledge is made up of user neighbourhoods, and (iv) distributed prediction – the aggregated knowledge is composed of single-domain recommendations. The authors show that the proposed methods can improve the accuracy of target domain recommendations in case of data sparsity, however, these methods can only be applied if the involved recommenders share some users or items and secondly, they must use the same recommendation technique.

One example of a knowledge transfer approach for cross-domain recommendations is presented in [Enrich *et al.* (2013)]. This paper introduces three novel cross-domain rating prediction models (UserItemTags, UserItemRelTags and ItemRelTags), which are based on the SVD matrix factorization model [Koren *et al.* (2009)]. They leverage the preference knowledge contained in tag assignments in order to improve the rating prediction task. The underlying hypothesis is that the information about how users tag items in an auxiliary domain can be exploited to improve the rating prediction accuracy in a different target domain, provided that there is an overlap between the set of tags used in the two domains. All the proposed models add tag factor vectors to the latent item vectors, which are then combined with the latent user features to compute rating predictions. The difference between these models lies in the set of tags used for rating prediction. UserItemTags and UserItemRelTags predict a target user rating by exploiting the tags this user has attached to the target item, whereas, ItemRelTags considers all the tags assigned by any user on the target item to predict ratings. To evaluate whether the exploitation of user tags collected in one domain could be useful to improve the rating prediction accuracy in a completely different domain, the authors carried out a series of off line experiments using the MovieLens and LibraryThing datasets. The obtained results indicate that the usage of tags is beneficial and their proposed models outperform the traditional matrix factorization model which only uses ratings [Koren *et al.* (2009)].

In another paper, Fernández-Tobías et al. [Fernández-Tobías *et al.* (2016)] have studied the quality of cross-domain recommendations in terms of accuracy, diversity and catalog coverage, by evaluating a number of algorithms (i.e., popular items, user-based nearest neighbours, item-based nearest neighbours, matrix factorization for positive-only feedback) on two datasets with positive-only feedback. Their results show an increased ranking accuracy in cold-start situations when cross-domain information is used.

The results for diversity varied across the two datasets, and hence the authors conclude that in general the results depend on the target domain. Finally, the results also indicate that a greater item diversity on the source profile can harm the perform in the target domain.

1.3.3 *Recommendation Based on Implicit Feedback*

In many real-world scenarios explicit feedback can be difficult to obtain or unavailable (e.g., news portals). In these cases, recommender systems can use implicit feedback. Indeed, they can monitor the users' behaviour, such as noting which items they browse or purchase, the duration of time spent viewing an item and the search terms they use, and use these observations in order to infer the user preferences regardless of the user's willingness to actively provide explicit evaluations, such as, ratings. In this scenario, recommenders have been built either by leveraging implicit feedback data only [Hu *et al.* (2008a); Rendle *et al.* (2009); Gurbanov *et al.* (2016)] or by extending recommender models based on explicit feedback. A notable example of this last category of approaches is SVD++ [Koren (2008)], which extends the standard matrix factorization model by adding a new set of item factor vectors in order to leverage preference information coming from the items for which users provided implicit feedback. In particular, in this model each user is characterized not only by a user factor vector, but with additional factor vectors which represent the contribution of the implicit feedback to the model of the user in the factors space.

We conclude this section by pointing out that approaches based on implicit and explicit feedback are only applicable if feedback information from users is available, which is not the case for newly registered users. Hence, the applicability of these approaches in the new user scenario is in both cases severely restricted.

1.3.4 *Content-Based Recommendation*

A classical approach for addressing the cold-start problem, and in particular the new item problem suffered from standard collaborative filtering-based recommenders, is to rely on preference information brought by the features associated with the liked items (content). For example, if a user has positively rated a POI of type"hiking trails" then the system can predict that other POIs of the same type too will be liked.

Among the many systems where content information is used, we mention the work of Manzato [Manzato (2013)]. The proposed gSVD++ model

shows how content metadata can be directly incorporated into matrix factorization and in particular into the SVD++ model proposed by Koren [Koren (2008)]. This is achieved by introducing a new set of latent factor vectors for content metadata (e.g., the type of a POI, the actors or the genre of a movie) that are combined with the item's latent factor vector. To evaluate gSVD++, the author used the MovieLens 100k dataset, and have found that it has lower MAE and RMSE than SVD++ and a previous model proposed in [Manzato (2012)], which also performs factorization on item's metadata.

Fernández-Tobias and Cantador [Fernández-Tobías and Cantador (2014)] adapted gSVD++ by enhancing the items' latent factor vectors with additional latent factor vectors for the tags that were applied to them. Likewise, the user's latent factor vector is extended with additional latent factor vectors representing the tags of the user, to better capture these interests. These enhancements tackle the new item problem and the new user problem. Moreover, it suffices that a tag is available for an item, but not necessarily a rating, in order to make the item recommendable. This happens in many situations, for instance, in the social bookmarking website Delicious[1], in which users can apply tags to their bookmarks, but are not asked to rate the bookmarked website.

[Shi *et al.* (2014)] provides a comprehensive overview of collaborative filtering recommender systems that integrate rich side information about items and users as well. The authors mostly focused on social networks and user-contributed information, such as user tags, geotags, multimedia content and reviews/comments, which has become widely available since the introduction of Web 2.0 technologies. Based on this focus, they have identified two types of challenges for recommender system research: 1) new conditions and tasks (e.g., social recommendation, group recommendation, long tail recommendation, cross-domain collaborative filtering); and 2) challenges due to the interaction between recommender systems and other areas of research (e.g., search, interaction and economics).

To conclude, by incorporating content information, a recommender system is capable of recommending items not yet rated by any user and hence can overcome the new item problem. Nevertheless, it does not solve the new user problem, since still enough ratings have to be collected before the system can really understand the user preferences and provide accurate recommendations.

[1] https://del.icio.us/

1.3.5 *Leveraging User Descriptions*

A notable way to solve the new user problem is to utilize user attributes (e.g., demographic data and personality characteristics) in the recommendation process. In [Vozalis and Margaritis (2004)], the authors enhance traditional collaborative filtering by integrating in to the model demographic information (i.e., age, gender, job and zip code). Specifically, when generating rating predictions, the authors propose to use not only the ratings-based correlation, but also the demographic correlation between the active user and a neighbour one. Demographic correlation is calculated by representing each user with a vector of 27 demographic features and applying vector similarity on the generated demographic vectors.

The same idea is used also in the approach presented in [Koren *et al.* (2009)]. Here, the authors extend their original SVD++ algorithm [Koren (2008)] by incorporating known user attributes. In practice, they consider Boolean attributes and a user u is described by the subset of their attributes $A(u)$. Attributes can, for instance, describe the gender, age group, zip code, income level, and so on. Then, the proposed algorithm computes a distinct factor vector $y_a \in \mathbb{R}^f$ for each attribute of the user.

User personality has also been used in collaborative filtering [Fernandez Tobias *et al.* (2016)]. In this article, the authors investigate three different approaches: (a) personality-based collaborative filtering, which directly improves the recommendation prediction model by incorporating user personality information; (b) personality-based active learning, which exploits personality information to first identify useful user preference data to be elicited in a target domain, and then improve the recommendation prediction model; and (c) personality-based cross-domain recommendation, which utilizes personality information to enhance the usage of preference data from auxiliary domains in order to compensate the lack of preference data in the target domain. The results of their experiments on the myPersonality dataset [Bachrach *et al.* (2012)] show that the proposed approaches are viable methods for improving collaborative filtering to tackle the new user problem.

To conclude, we note that the exploitation of user-related information can help to generate better recommendations for new users, i.e., for which none or a limited number of ratings is available. However, in general, recommendations based only on user features are less accurate than those based on user preferences. In fact, for instance, demographic factors account only for a little part of the variance of the rating behaviour.

1.3.6 *Better Using Existing Preference Knowledge*

Instead of acquiring new information about either the user or the application domain, as in the approaches illustrated in the previous sections, one can simply try to better use the available information.

A popular solution moving in this direction is offered by hybrid recommender systems. These systems combine two or more recommendation techniques in order to improve their performance and to overcome the limitations of the combined techniques [Burke (2007)]. For instance, the new item problem, which is an issue for pure Collaborative Filtering (CF), can be overcome by hybridizing CF with a content-based component. According to [Burke (2002)], it is possible to identify seven hybridization techniques:

Weighted The scores of different recommendation components are combined into a single weighted score.

Switching The system selects and applies a single recommender from among its constituents based on the current recommendation situation.

Mixed Recommendations generated by different recommendation components are presented side-by-side in a combined list.

Feature Combination Features derived from different recommenders are combined and injected into a single recommendation algorithm.

Cascade Recommendation components are strictly hierarchically ordered from the strongest to the weakest, with the ties observed by using a technique are broken by the following one.

Feature Augmentation A feature or set of features computed by one recommendation technique are passed as input to the next technique.

The idea of better using existing preference data is especially important in context-aware recommenders. In fact, since these systems need significantly more preference data than context-free ones, for CARS all the available preference data should be exploited as much as possible. Hence, some techniques have been proposed in the CARS literature aiming at better exploiting the (usually sparse) set of contextually-tagged ratings. An approach worth to be mentioned is the generalized pre-filtering approach proposed by Adomavicius et al. [Adomavicius and Tuzhilin (2015)]. It exploits hierarchical relationships between different contexts in order to generalize a target context when the number of ratings acquired in that particular context is small. For example, if we would like to identify which POIs to recommend on a Monday, we may use not only the ratings for POIs collected on Mondays, but also those obtained on other weekdays

(but not in the weekend). This results in a significantly better usage of the existing rating dataset, and allows to build more robust and general rating prediction models.

A similar idea to deal with the cold-start problem in context-aware recommenders is used by Zheng et al. [Zheng *et al.* (2012)]. They also try to increase the number of ratings used to generate predictions by defining a relaxed notion of match between the target context and the contexts associated with ratings.

The main limitation of the two aforementioned approaches is that it is necessary to choose the right level of generalization for contexts. This might be easy for datasets with relatively dense in-context ratings, but for datasets where only a small amount of in-context ratings is available, finding the proper level of generalization remains problematic.

One possible solution, is offered by the Differential Context Weighting (DCW) approach [Zheng *et al.* (2013)], which relies on the concept of weighting vectors and similarity of contexts in order to weight the contribution of individual contextual factors and ratings in the rating prediction algorithm, respectively. More specifically, weighting vectors are used to indicate the influential power of each contextual dimension in the various components involved in the recommendation process, whereas similarities are used to assess how much to weight a rating under a particular target context. This allows to consider all (weighted) ratings in the rating prediction process instead of completely filtering them out when they have non-matching contexts.

Similarly, also Codina et al. [Codina *et al.* (2013)] show that, in the recommendation process, instead of using only contextual ratings that exactly match the target context, it is possible to reuse also those that were provided in similar contexts. In their approach, two contexts are deemed as similar if they are influencing the items' (or users') ratings in a similar way.

Yet another example is the Contextual Sparse LInear Method (CSLIM) approach of [Zheng *et al.* (2014, 2015)]. It also reuses ratings collected in contexts that do not match the target context for rating prediction, i.e., by weighting them via the learned contextual rating deviation or similarity (correlation) terms.

1.4 Tools & Datasets

In this section, we discuss tools and datasets that can be used to build RSs that are resistant to the cold start and data sparsity problems. While it is impossible to provide an exhaustive survey of the ever-growing number of available resources, this selection should present a proper initial selection and where tools and datasets might be headed next.

1.4.1 *Tools*

In general, there are two ways of integrating a RS into a product, either to use a tool providing the RS as a service, i.e., Software as a Service (SaaS) RS, or to use a software framework providing functionality that can be incorporated into the source code of the product.

Yusp[2] (former *Gravity*) provides one of the most well-known SaaS solutions for RSs. The company offers to their customers a recommendation engine that generates recommendations by collecting and utilizing data of a customer's users through an API. Customers can manage the engine settings using a special dashboard. Similarly to many other SaaS products, this recommendation engine is closed-source. To solve the cold start and the data sparsity problems, Yusp collects knowledge from different resources[3] and uses hybridization techniques[4]. The available hybridization strategies, which are here called *logics*, can be selected from a list of the pre-defined ones or can be created from scratch using the dashboard tool. The weighting, switching or cascade strategies are available, but there are some more [Burke (2007)]. For example, the "optimized more like this" strategy attempts to make recommendations based on collaborative filtering, however, if there is not enough behavioural data (e.g., clicks or views) for an item, it "falls back" to content-based models. In addition, Yusp can distinguish the users who come to the service only to browse from those who know what they are looking for specific items[5]. Knowing the users' intent the recommendation method or the recommended items are further adapted. For instance, if someone clicks on everything from phone cases to real estate within a short period of time, the system will assume she is only there for browsing and will not use their click history for recommendations.

[2] http://www.yusp.com/
[3] http://support.yusp.com/support/solutions/articles/5000717574-actions-what
[4] http://support.yusp.com/support/solutions/articles/5000712803-select-a-recommendation-logic
[5] http://www.yusp.com/blog/cold-start-problem-recommender-systems/

Other companies, such as *RetailRocket*[6], *nosto*[7] or *Relap*[8], provide recommendation functionality similar to those included in Yusp. However, little or nothing is known about the recommendation techniques and methods used by these companies. The target application domains for SaaS RSs are usually e-commerce, media and news.

In comparison to SaaS solutions, almost all RS software frameworks are open-source and well-documented, that makes them more attractive for developers and researchers. The frameworks provide generic functionality that can be further tailored to the application specific goals by additional code written on top of the framework. Some of the frameworks, such as rrecsys [Çoba and Zanker (2016)], SurPRISE[9], LibRec [Guo *et al.* (2015)], and Hi-Rec [10] [Elahi *et al.* (2017)] have been designed for rapid prototyping of recommendation algorithms. These frameworks contain only basic recommendation models. Others, such as *Turi*[11], *Apache Mahout*[12], *Mortar*[13], *Seldon*[14] and *Oryx*[15], can be used for building large scale fully-fledged recommendation engines that can cope with the cold start and the data sparsity problems. In this section, we will consider only frameworks from the second group.

To overcome the cold start and the data sparsity problems, frameworks, as well as SaaS RSs, combine data from different knowledge sources. This is done either by utilizing recommendation models that can work with multiple data sources or by employing hybridization techniques. For example, Apache Mahout generates recommendations by leveraging information about actions of different types performed by users while interacting with the product (clicks, views, purchases, etc.)[16]. The open-source recommender engine Mortar uses a graph-based approach that can blend collaborative filtering with content-based recommendations and other signals[17]. Finally, the Seldon recommender engine allows the product owners to spec-

[6] https://retailrocket.net/
[7] http://www.nosto.com/
[8] https://relap.io/
[9] http://surpriselib.com
[10] https://fmoghaddam.github.io/Hi-Rec/
[11] https://turi.com/
[12] http://mahout.apache.org/
[13] https://github.com/mortardata/mortar-recsys
[14] https://www.seldon.io/
[15] http://oryx.io/
[16] https://mahout.apache.org/users/algorithms/intro-cooccurrence-spark.html
[17] http://help.mortardata.com/data_apps/recommendation_engine/recommendation_engine_basics

ify in a configuration file the set of models that must be used, the order in which these models should be applied, and the priority of each model[18] (i.e., hybridize several recommendation models).

Table 1.1 summarizes the properties and features of the aforementioned frameworks. The *Specialization* column contains a brief description of the most specific feature of the frameworks. It can be noted that only Mortar has been developed specifically for solving RS tasks. All the other frameworks, even though they provide a range of prediction tasks employed in RSs, they have been developed as scalable Machine Learning (ML) frameworks for general purposes. The *Components* row shows additional tools and frameworks that are used to support the target RS framework and let it run on a computer cluster. In fact, all the frameworks can be run on a computer cluster. To employ a framework some *Minimum Requirements* should be met. For example, Oryx can be deployed only on Hadoop cluster[19], while Seldon requires Kubernetes[20]. The recommendation techniques supported by the frameworks are listed in the *RS models* row. Turi provides the largest set of techniques, including, matrix factorization (MF) [Koren *et al.* (2009)], implicit feedback MF (IMF) [Hu *et al.* (2008a)], SVD++ [Koren (2008)], item-based CF [Deshpande and Karypis (2004)], content-based and popularity-based models.

All the frameworks, except Oryx, allow filtering items using *meta-parameters*, such as tags and content information (genres, categories, etc.). Mortar and Apache Mahout can leverage information about user actions of different types, but do not provide out-of-the-box HTTP API and cannot provide recommendations for anonymous users.

When using Turi and Oryx the request to the RS can specify the set of items that a user has observed or has interacted with[21, 22].

These items are used by the framework to generate the user's profile on-the-fly and, therefore, to generate recommendations for *anonymous* users (often are cold users). Knowing the similarity between items, Oryx and Mortar build recommendations containing diverse items, that can be useful to acquire more information about users' preferences. Moreover, they provide functionality that can be used to explain the recommenda-

[18] http://docs.seldon.io/advanced-recommender-config.html
[19] http://hadoop.apache.org/
[20] https://kubernetes.io/
[21] http://oryx.io/apidocs/com/cloudera/oryx/app/serving/als/RecommendToAnonymous.html
[22] https://github.com/turi-code/userguide/blob/master/recommender/making-recommendations.md#making-recommendations-for-new-users

	Frameworks				
	Apache Mahout	Mortar	Oryx	Seldon	Turi
Specialization	Scalable algorithms	Personalized recom. at scale	Real-time large scale ML	Scalable ML with deployment	End-to-end large-scale data analysis and data product development
Components	Apache Hadoop, Apache Spark, A search engine	Mortar Development Framework	Apache Spark, Apache Kafka, Apache Hadoop, Apache Zookeper	Apache Spark, MySQL, Kubernetes	Graphlab Create, Turi Distributed framework
Minimum requirements	-	Mortar account, Github account, AWS account	Hadoop cluster	Kubernetes cluster	Turi license
Main language	Java	Pig	Java	Scala/Python	Python
RS models	Item-based CF, MF with ALS, IMF, SVD++, Content-based	Graph-based	MF with ALS, Popularity-based	MF with ALS, User-Clustered MF, Popularity-based	Item-based CF, Content-based, MF with ALS/SGD/adagrad, IMF, SVD++, Popularity-based
Actions of multiple types	Yes	Yes	No	No	No
Recommendations delivery	Hadoop command-line interface	Stores all the recommendations in Dynamo DB	HTTP API	Seldon command-line interface, HTTP API	Graphlab SDK, HTTP API
Anonymous user	No	No	Yes, user is defined by a set of items she interacted with	Yes, by creating a hybrid model	Yes, user is defined by a set of items she interacted with
Diversity	No	Yes	Yes	No	Yes
Explanations	No	Yes	Yes	No	No

Table 1.1: Properties and features of the frameworks

tions. One framework only supports hybridization (cascading), this is Seldon. And only one framework supports active learning, this is Oryx. It provides the "surprise me" functionality, which recommends items that a user least-likely will interact with.

To sum up, though numerous RS tools and frameworks is currently available, only few of them are equipped with functionality capable to mitigate the cold start and the data sparsity problems. The problems are usually treated by using additional knowledge sources about the users, the items or the contextual situations. For example, Apache Mahout, Turi and Oryx use algorithmic solutions based on implicit feedback data, while Yusp and Mortar employ content-based filtering. Beyond that, Yusp and Seldon provide hybridization and basic active learning techniques. Yusp offers weighting, switching and cascading hybridization approaches, while Seldon supports the lowest predicted (a. k.a. "surprise me") AL strategy. Whilst these solutions allow reducing the problems we believe that the efficient implementation of more advanced hybridization and AL techniques described in this chapter can significantly improve the existing tools.

1.4.2 *Datasets*

There are many publicly available datasets that can be used to conduct research in the field of recommender systems; in particular, for analyzing, testing and comparing existing recommenders. In this section, we will mention datasets that can be employed to train models resistant to the cold start and the data sparsity problems. Such datasets contain information about interactions between users and items and satisfy at least one of the following conditions:

- the dataset contains user and item metadata;
- the dataset contains additional interaction information (i.e., user actions of multiple types);
- the dataset reflects the real-time (or near real-time) changes in the user behaviour.

The first two conditions are essential for building and testing systems which are based on hybridization or use additional knowledge sources, while the third one might be helpful in building an active learning, rating elicitation, process.

One of the most popular recommender system data sets is MovieLens [Harper and Konstan (2015)]. The dataset comprises 5-star ratings

and free-text tags collected by a movie recommendation service. Using tags assigned by users to movies as well as movie genres one can build a content-based recommender system. Other data sets containing item and user metadata are Amazon product data [He and McAuley (2016)] and Yelp dataset[23]. The first one includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs) from Amazon. The second one is a subset of Yelps businesses, reviews, and user data that incorporates over 1.2 million business attributes like hours, parking, availability, and ambiance. It also aggregates check-ins over time for each of the 174,000 businesses.

In addition to user and item metadata, datasets can include user (behavioral) data, such as information about personalty or actions of different types performed by the users while interacting with the system. For instance, My Personality dataset [24] and STS dataset [25] both contain the personality of the users together with other types of data. The RetailRocket [26] as another dataset that contains logs of users' actions, such as, clicks, add to carts, and transactions, that were collected from the e-commerce website over a period of 4.5 months. Another good example is XING's dataset [Abel *et al.* (2017)] which contains the logs of interactions performed by users on job postings (see chapter **??**). These interactions include clicks, bookmarks, replies, and deletes. It also includes reciprocal interactions, e.g., whether or not a recruiter has showed his interest into a user by clicking on a candidate user's profile. It is worth noting that the available XING dataset is a semi-synthetic sample of another data set, and it is enriched with artificial users whose presence contributes to the anonymization. Moreover, some noise was added to the data: not all the users' interactions are contained in the dataset while some of the interactions are artificial (have actually not been performed by the users). Four types of actions are contained in the dataset: clicks, bookmarks, replies and deletes.

Moreover, recently a set of datasets have been published, called Mise-en-scene [Deldjoo *et al.* (2016)] [27] and MPEG-7 visual datasets [28] [Deldjoo

[23] https://www.yelp.com/dataset/
[24] https://mypersonality.org
[25] https://www.researchgate.net/publication/305682479_Context-Aware_Dataset_STS_-_South_Tyrol_Suggests_Mobile_App_Data
[26] https://www.kaggle.com/retailrocket/ecommerce-dataset
[27] https://www.researchgate.net/publication/305682388_Mise-en-Scene_Dataset_Stylistic_Visual_Features_of_Movie_Trailers_description
[28] https://goo.gl/ZYij61

et al. (2018)].

These datasets contain various types of visual features extracted from movie trailers. The idea is that, for the new movie items in the extreme cold start situation, where there is no data available, visual features can be used to generate personalized recommendation. These features are descriptive of the aesthetic elements (e.g., color, light, and motion), encapsulated within the movies. The features are automatically extracted by analyzing frame-by-frame of each movie and aggregating them over the entire movie.

Finally, *Satori*[29] and *webhose.io*[30] are services that allow collecting data about social activity, e-commerce reviews, and news in near real-time. The data can be used to explore the influence of different events on the users' preferences. Using this knowledge one can separate the reviews which reflect more stable user preferences from those that were mostly influenced by some exogenous event.

1.5 Performance Comparison

In this section, we will summarize the experimental results contained in research studies that focused on collaborative filtering systems in cold start scenarios. However, preliminary, we would like to make a few important observations.

First of all, most of the studies conducted in this particular research area have adopted an off line evaluation setting and measured the rating prediction error (e.g., Mean Absolute Error - MAE, or Root Mean Squared Error - RMSE). Hence, we do not have any knowledge about how the surveyed techniques may perform with respect to other important metrics, such as those measuring ranking quality (e.g., Normalized Discounted Cumulative Gain - NDCG, or Mean Average Precision - MAP). Moreover, the majority of the surveyed studies have focused on the movie recommendation domain and used the well-known MovieLens and Netflix datasets.

Most importantly, in spite of the similarities among the adopted evaluation methods, still there are some substantial differences, and hence, it is almost impossible to draw any final conclusion, as well as to generalize the results obtained in the experiments.

As summarized in Table 1.2 and described in more detail in this section, the surveyed techniques for tackling the cold-start problem have their own

[29] https://www.satori.com/
[30] https://webhose.io/

advantages and drawbacks and have been evaluated on different datasets[31].

Please note that *HLU* refers to *Half Life Utility* and it measures the utility of a recommendation list for every user, assuming that the likelihood that she will notice and choose a recommended item decays exponentially with respect to the ranking of the item's Breese *et al.* (1998); Pan *et al.* (2008); Schedl *et al.* (2018)]. Hence, greater value of *HLU* may correspond to a superior recommendation performance. Additionally, *MPR* refers to *Mean Percentile Ranking* and it is a metric that measures the user satisfaction of the items in a ranked list of recommendations, and it is computed as the average of the percentile ranking of the test items within the ranked list of recommendations for every test user Hu *et al.* (2008b); Schedl *et al.* (2018)]. It is worth noting that the smaller *MPR*, the better recommendation performance.

First, let us consider Active Learning – it has been shown in offline and online experiments that it can be used to effectively tackle the new user, new item and new context problem by selecting informative items for users to rate. More specifically, experimental results obtained in previous research have provided evidence that a proper choice of an active learning elicitation strategy allows to improve the performance of a recommender system in terms of rating prediction and ranking accuracy for new users, new items and new contextual situations Rashid *et al.* (2002, 2008a); Elahi *et al.* (2014); Odić *et al.* (2012); Braunhofer and Ricci (2016)]. However, users may perceive active learning as tedious or even unpleasant, as it takes time and effort for them to browse the proposed items and rate them.

Also cross-domain recommendation can be a compelling approach to solve the cold-start problem. It has been shown that by exploiting cross-domain recommendation techniques it is possible to improve the prediction accuracy Berkovsky *et al.* (2007); Enrich *et al.* (2013)] as well as the ranking accuracy [Fernández-Tobías *et al.* (2016)] in the target domain in various cold-start situations. Cross-domain recommendation techniques, however, require the co-existence of users, items and contextual situations across the considered and different domains, which is not always the case. Otherwise, wrong conclusions about the user preferences might be transferred from the

[31] MovieLens Harper and Konstan (2016)], Netflix Koren (2009)], STS Braunhofer *et al.* (2014)], EachMovie [GroupLens (2018)], MyPersonality [Kosinski *et al.* (2015)], LibraryThing [Librarything (2018)], 7TV [Gurbanov *et al.* (2016)], Rossmann Rendle *et al.* (2009)], Adom [Adomavicius *et al.* (2005)], CoMoDa [Košir *et al.* (2011)], InCarMusic [Baltrunas *et al.* (2011)], TripAdvisor [TripAdvisor (2018)] , IMDB [IMDB (2018)]

auxiliary domain to the target domain.

A similar comment holds for recommendation approaches that incorporate implicit feedback. Implicit feedback data is generally easier to collect than explicit ratings data, and it has been shown that by extending rating-based recommender models using implicit feedback it is possible to achieve a higher recommendation performance [Koren (2008)]. However, recommendation approaches based on implicit data fail on users, items and contextual situations which were just entered into the system and for which no implicit feedback is yet available.

The other techniques, i.e., content-based recommendation and demographic-based recommendation, deal with just one specific type of cold-start problems. For instance, content-based techniques can lead to a higher prediction and ranking accuracy for new items (new item problem) [Manzato (2013); Fernández-Tobías and Cantador (2014)], whereas demographic-based techniques can improve the prediction and ranking recommendation accuracy for new users (new user problem) [Koren *et al.* (2009); Fernandez Tobias *et al.* (2016)].

As we have already mentioned in Section 1.3.6, the observed advantages and drawbacks of these techniques motivated the development of hybrid approaches that exploit simultaneously more than one technique, and adaptively use them for recommendation depending on their strengths and weaknesses in a given (cold-start) situation.

1.5.1 *Caveats*

There are several important issues related to the evaluation of recommendation techniques to deal with the cold-start problem that should also discussed. We list these aspects in the rest of this section.

Direct comparison of cold-start solutions. Different cold-start solutions have been typically evaluated in isolation with each other, i.e., new active learning solutions are compared to state-of-the-art active learning solutions, newly proposed cross-domain recommendation algorithms are confronted with other cross-domain recommendation algorithms, and so on. It is still an open and important question which one is more effective, e.g., in terms of recommendation accuracy, effort required from the users or simplicity of implementation.

User-centric evaluation studies. Off line evaluations are the most common evaluation methods for cold-start solutions. Although they allow for low-cost simulation of the behaviour of users when interacting with different

Approach	Cold-start problem			Metric			Evaluation		Dataset
	New user	New item	New context	RMSE, MAE	Precision, MAP	NDCG, HLU, MPR	Offline	Online	
Active Learning	✓	✓	✓	✓	✓	✗	✓	✓	CoMoDa, MovieLens, Netflix, STS
Cross-domain recommendation	✓	✓	✓	✓	✓	✓	✓	✗	EachMovie, MyPersonality, MovieLens, LibraryThing
Implicit feedback	✗	✗	✗	✓	✓	✓	✓	✗	7TV, MyPersonality, Netflix, Rossmann
Content-based recommendation	✗	✓	✗	✓	✓	✓	✓	✗	MovieLens, LibraryThing
Demographic-based recommendation	✓	✗	✗	✓	✗	✗	✓	✗	MovieLens
Hybrid recommendation	✓	✓	✓	✓	✓	✗	✓	✗	IMDb, MovieLens, Netflix
Context-aware models optimized for data sparsity	✗	✗	✓	✓	✓	✓	✓	✗	Adom, CoMoDa, InCarMusic, TripAdvisor, STS

Table 1.2: Summary of the performance of various cold-start solutions on different data sets.

algorithms, they can not substitute on line evaluations. On line evaluations are more costly, however, as they are carried out with real users in almost real-life settings, they allow to draw more reliable conclusions about the true merits of an algorithm.

Definition of benchmark results, evaluation procedures and public large-scale datasets. The evaluation of cold-start solutions for recommender systems is complex for several reasons: (1) it requires large feedback datasets with user and item attributes; otherwise it is difficult or even impossible to test the effects of using user and item attributes in the prediction models; (2) the evaluation must cover multiple perspectives, e.g., it must consider new users, new items, new contextual situations, mixtures of elementary cold-start cases, different degrees of coldness and different types of user and item attributes available; and finally (3) the evaluations and results of the cold-start solutions proposed so far are difficult to compare as there exists no standard or reference evaluation procedure as well as metrics.

1.6 Guidelines

In this section, we suggest a number of practical guidelines that can be adopted for the design and development of an operational collaborative filtering system, resistant to cold start problem.

Algorithms should adapt to the time evolution of user/item profiles: one of the important factors that could impact on the behavior of collaborative filtering systems, is the temporal dynamics of the user preferences which could be dependent on the application domain. As an example, the ratings that the users provide to TV programmes or books are more stable than the ones provided to news items, as the news items change more rapidly [Picault *et al.* (2011)]. This is important to take into account when designing the algorithms that can better adapt to the dynamic nature of the reality of collaborative filtering systems. Adaptive algorithms can tackle this issue by learning the temporal evolution of the user/item profile recognizing the different evolution patterns, e.g., stable profiles, progressive profiles, fast changing profiles, etc. [Picault and Ribiere (2008); Hawalah and Fasli (2015)]

Scalability of the recommender algorithm should be taken into account: the computation load performed by collaborative filtering algorithms can increase dramatically with the growth of the dimensions of the rating matrix, i.e., the number of users and items in the dataset [Park *et al.* (2012)]. It will also increase the sparsity of the data and hence create severe cold start

problem. Consequently, a recommendation algorithm that could operate with a limited volume of users and items may fail to generate recommendations in a reasonable time if a considerable number of new users and new items are added to the dataset. Hence, in real world recommender systems, with huge databases (big data), it is crucial to adopt algorithms that are capable of scaling up (see chapters **??**, **??**,**??**, and **??**). For instance, collaborative filtering algorithms based on Dimensionality Reduction methods (e.g., Singular Value Decomposition - SVD) are able to significantly reduce the computational cost of a recommendation and still to generate highly accurate recommendations [Koren and Bell (2015); Isinkaye *et al.* (2015)].

Ratings on recommended items can cause serious system bias: recommender systems typically obtain ratings on items with the highest predicted ratings (recommendations). This is due to the fact that it is more likely that the users assess and rate recommended items which are supposed to be interesting to them. However, it has been shown [Elahi *et al.* (2014)] that ratings given for such items may inject in the data set a large number of high ratings, which will ultimately cause a serious bias of the prediction algorithm for high ratings (see chapter **??**). Therefore, it is also important that the system adopts certain strategies in order to obtain ratings on items that the users may dislike and would rate low.

Natural acquisition of ratings can strongly impact the system performance: in operational recommender systems the ratings are added to the system database through two different processes [McNee *et al.* (2003); Carenini *et al.* (2003); Rashid *et al.* (2008b); Elahi *et al.* (2014)]: (i) the system explicitly requests the user to rate a set of selected items (active learning), or (ii) the user voluntary explores the item catalog and provides some ratings (natural acquisition of ratings). The majority of the research works have studied the performance of collaborative filtering systems in cold start settings, under the assumption that the new ratings are added only as a response to the system requests. However, in reality, user ratings are generated by both processes with diverse impact on the system performance. Hence, it is important to consider a *mixed initiative* scenario [Rashid *et al.* (2008b); Pu *et al.* (2012)], by considering both rating elicitation sources. This may provide a better prediction of the temporal evolution of the system performance and a more realistic scenario [Elahi *et al.* (2014, 2012)].

User ratings on random items could be beneficial: supporting users in exploring items and providing ratings on random items has shown to be useful in reducing the system bias. This could be more effective partic-

ularly when the system contains very popular items that receive almost all the ratings from the users. For systems that include rating elicitation in the sign-up process, a small portion of randomly selected items can be included in the lists the users are proposed to rate [Zhao *et al.* (2013); Christakopoulou *et al.* (2016)].

Conversational interaction models could improve user profiling: the standard preference elicitation model of current collaborative filtering systems, is mainly supporting the generation of the initial user profiling, during the sign-up process; the system builds the user profile by requesting the user to rate a set of items [Carenini *et al.* (2003)]). However, the user should be able to update her profile (by ratings more items) whenever she likes. This approach is analysed in [Carenini *et al.* (2003); Sun *et al.* (2013)]. Here the user rates items (selected by the system) in the sign-up process, and later on, she still gets notifications from the system motivating her to rate more items, which are selected by the system. This process includes explanations which clarify the benefits of providing more ratings. Therefore, the control is still in the user's hands, while a sense of co-operation between the user and the system is formed.

Elicitation of contextual ratings are necessary for CARS: in context aware recommender systems (CARS), the context of the user plays a significant role in recommendation accuracy. While there are several types of contextual information that can be automatically obtained from sensors (e.g., weather, temperature, location, daytime, season, and weekday), there are also contextual information that have to be specified by the user (e.g., budget, companion, mood, and transport mean). However, not all the contextual factors are equally useful for the system to improve the recommendation accuracy. Hence, actively selecting the contextual factors that are the most informative to explain the rating given by the user to an item is important and can potentially improve the system accuracy more while being easier for the user to provide them. So, the application of active learning in context-aware recommender systems is in order [Baltrunas (2011)].

Hybridization can result in robustness: while collaborative filtering approaches are considered powerful and effective in generating relevant recommendations, they have several limitations, which are impacting on their performance in cold start situations. For instance, if two items have synonymous names, e.g., espresso machine and coffee maker, a collaborative filtering systems may not consider these items similar until they are similarly rated by the users [Isinkaye *et al.* (2015)]. Another example is rec-

ommendation in the news domain where the items (news articles) may get outdated quickly and the users may not be willing to read them any more. As a consequence, there would be low overlap among the ratings of users, and the recommendation quality may drop considerably. In both the above-mentioned problems, hybridizing collaborative filtering with content-based could provide an effective solution [Burke (2007)]. This is probably the reason why almost all real-world recommender systems employ hybridization techniques in order to increase system robustness.

1.7 Conclusions and Future Directions

As was mentioned previously, collaborative filtering suffer from the cold start problem which occurs when the system is not capable of identifying items that could be recommended to a new user or users that may receive a recommendation for a new item. Moreover, when the system has collected only a small percentage of all the potential ratings, the rating sparsity problem may be observed. These problems are even worse in Context-Aware Recommender Systems (CARSs) where the recommendations can be requested for contextual situations under which the users did not rated any item.

Various approaches for addressing the cold start and data sparsity problems and improving collaborative filtering have been proposed. These approaches can be split into two major groups. Solutions in the first group exploit additional knowledge sources about the users, the items or the contextual situations, and incorporates active learning (AL), cross-domain, implicit feedback, content-based and demographic-based approaches. While solutions in the second group leverage hybrid approaches that try to make a better use of the available information by combining two or more recommendation techniques.

However, no solution is fully solving the considered problem and each of them has drawbacks. For instance, one cannot leverage implicit feedback data in the new user scenario, because even this type of user/item interactions are missing. Similar limitations can be found for the content-based approach where a sufficient number of ratings have to be collected before the system can understand the user preferences and provide accurate recommendations. The demographic-based approaches are easier to implement but less accurate than personalized recommendations, and the cross-domain approaches require the co-existence of users, items and contextual situations across different domains, which is not always the case.

Meanwhile, users do not like to enter ratings requested by the system, as this is tedious and takes time and effort. Finally, only basic active learning and hybridization techniques are present in the existing tools and frameworks for building RSs. Thus, by taking into account all these aspects, we would like to close this chapter by briefly indicating possible future directions for addressing the cold start and data sparsity problems.

As IT-technology gets more advanced, it becomes possible to collect and process more information from different sources. For example, "Internet-of-Things" devices, are capable of sensing their environment and collect users' contextual and behavioural information [Tu *et al.* (2016)]. The availability of data of different types (collected from various sources) entails the creation of more advanced hybrid approaches.

Another direction is related to the recent development of artificial intelligence and its application in dialog systems (e.g., chatbots or smart assistants). This type of technologies can transform conversational recommender systems from a "bothersome process" to an enjoyable one, satisfying both the system objectives and the user at the same time [Rubens *et al.* (2015)].

Finally, the efficient implementation of the advanced hybridization and active learning techniques in the existing systems tools and frameworks may stimulate the research community to find even better solutions to solve to the cold start and data sparsity problems.

Bibliography

Abel, F., Deldjoo, Y., Elahi, M., and Kohlsdorf, D. (2017). Recsys challenge 2017: Offline and online evaluation, in *Proceedings of the Eleventh ACM Conference on Recommender Systems* (ACM), pp. 372–373.

Adomavicius, G., Sankaranarayanan, R., Sen, S., and Tuzhilin, A. (2005). Incorporating contextual information in recommender systems using a multidimensional approach, *ACM Transactions on Information Systems (TOIS)* **23**, 1, pp. 103–145.

Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, *IEEE Trans. on Knowl. and Data Eng.* **17**, 6, pp. 734–749, doi:10.1109/TKDE.2005.99.

Adomavicius, G. and Tuzhilin, A. (2011). Context-aware recommender systems, in *Recommender Systems Handbook* (Springer), pp. 217–253.

Adomavicius, G. and Tuzhilin, A. (2015). Context-aware recommender systems, in *Recommender Systems Handbook* (Springer), pp. 191–226, doi:10.1007/978-1-4899-7637-6_6.

Agarwal, D. and Chen, B.-C. (2009). Regression-based latent factor models, in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09 (ACM, New York, NY, USA), ISBN 978-1-60558-495-9, pp. 19–28, doi:10.1145/1557019.1557029, http://doi.acm.org/10.1145/1557019.1557029.

Bachrach, Y., Kosinski, M., Graepel, T., Kohli, P., and Stillwell, D. (2012). Personality and patterns of facebook usage, in *Proceedings of the 4th Annual ACM Web Science Conference*, WebSci '12 (ACM, New York, NY, USA), ISBN 978-1-4503-1228-8, pp. 24–32, doi:10.1145/2380718.2380722, http://doi.acm.org/10.1145/2380718.2380722.

Baltrunas, L. (2011). *Context-Aware Collaborative Filtering Recommender Systems*, Ph.D. thesis, The Department of Computer Science, Free University of Bozen - Bolzano.

Baltrunas, L., Kaminskas, M., Ludwig, B., Moling, O., Ricci, F., Aydin, A., Lüke, K.-H., and Schwaiger, R. (2011). Incarmusic: Context-aware music recommendations in a car, in *International Conference on Electronic Commerce*

and Web Technologies (Springer), pp. 89–100.

Baltrunas, L., Ludwig, B., Peer, S., and Ricci, F. (2012). Context relevance assessment and exploitation in mobile recommender systems, *Personal Ubiquitous Comput.* **16**, 5, pp. 507–526, doi:10.1007/s00779-011-0417-x, http://dx.doi.org/10.1007/s00779-011-0417-x.

Berkovsky, S., Kuflik, T., and Ricci, F. (2007). Distributed collaborative filtering with domain specialization, in *Proceedings of the 2007 ACM Conference on Recommender Systems*, RecSys '07 (ACM, New York, NY, USA), ISBN 978-1-59593-730–8, pp. 33–40, doi:10.1145/1297231.1297238, http://doi.acm.org/10.1145/1297231.1297238.

Blédaité, L. and Ricci, F. (2015). Pairwise preferences elicitation and exploitation for conversational collaborative filtering, in *Proceedings of the 26th ACM Conference on Hypertext & Social Media*, HT '15 (ACM, New York, NY, USA), ISBN 978-1-4503-3395-5, pp. 231–236, doi:10.1145/2700171.2791049, http://doi.acm.org/10.1145/2700171.2791049.

Braunhofer, M. (2015). *Techniques for Context-Aware and Cold-Start Recommendations*, Ph.D. thesis, Free University of Bozen-Bolzano.

Braunhofer, M., Elahi, M., and Ricci, F. (2014). Techniques for cold-starting context-aware mobile recommender systems for tourism, *Intelligenza Artificiale* **8**, 2, pp. 129–143, doi:10.3233/IA-140069.

Braunhofer, M., Elahi, M., and Ricci, F. (2015a). User personality and the new user problem in a context-aware point of interest recommender system, in *Information and Communication Technologies in Tourism 2015* (Springer), pp. 537–549.

Braunhofer, M., Fernández-Tobías, I., and Ricci, F. (2015b). Parsimonious and adaptive contextual information acquisition in recommender systems, in *IntRS@RecSys*, *CEUR Workshop Proceedings*, Vol. 1438 (CEUR-WS.org), pp. 2–8.

Braunhofer, M. and Ricci, F. (2016). Contextual information elicitation in travel recommender systems, in *Information and Communication Technologies in Tourism 2016* (Springer), pp. 579–592.

Breese, J. S., Heckerman, D., and Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering, in *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann Publishers Inc), pp. 43–52.

Burke, R. (2000). Knowledge-based recommender systems, in *Encyclopedia of library and information science*, 32 (CRC Press), pp. 181–201.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments, *User Modeling and User-Adapted Interaction* **12**, 4, pp. 331–370, doi:10.1023/A:1021240730564, http://dx.doi.org/10.1023/A:1021240730564.

Burke, R. (2007). The adaptive web, in P. Brusilovsky, A. Kobsa, and W. Nejdl (eds.), *The Adaptive Web: Methods and Strategies of Web Personalization*, chap. Hybrid Web Recommender Systems (Springer-Verlag, Berlin, Heidelberg), ISBN 978-3-540-72078-2, pp. 377–408, http://dl.acm.org/citation.cfm?id=1768197.1768211.

Cantador, I. and Cremonesi, P. (2014). Tutorial on cross-domain recommender

systems, in *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys'14 (ACM, New York, NY, USA), ISBN 978-1-4503-2668-1, pp. 401–402, doi:10.1145/2645710.2645777.

Carenini, G., Smith, J., and Poole, D. (2003). Towards more conversational and collaborative recommender systems, in *Proceedings of the 8th international conference on Intelligent user interfaces*, IUI '03 (ACM, New York, NY, USA), ISBN 1-58113-586-6, pp. 12–18, doi:10.1145/604045.604052.

Christakopoulou, K., Radlinski, F., and Hofmann, K. (2016). Towards conversational recommender systems, in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16 (ACM, New York, NY, USA), ISBN 978-1-4503-4232-2, pp. 815–824, doi:10.1145/2939672.2939746, http://doi.acm.org/10.1145/2939672.2939746.

Çoba, L. and Zanker, M. (2016). rrecsys: An r-package for prototyping recommendation algorithms, in *RecSys Posters*.

Codina, V., Ricci, F., and Ceccaroni, L. (2013). Exploiting the semantic similarity of contextual situations for pre-filtering recommendation, in *User Modeling, Adaptation, and Personalization* (Springer), pp. 165–177.

Cremonesi, P., Tripodi, A., and Turrin, R. (2011). Cross-domain recommender systems, in *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on* (IEEE), pp. 496–503.

Deldjoo, Y., Elahi, M., Cremonesi, P., Garzotto, F., Piazzolla, P., and Quadrana, M. (2016). Content-based video recommendation system based on stylistic visual features, *Journal on Data Semantics* , pp. 1–15doi:10.1007/s13740-016-0060-9.

Deldjoo, Y., Elahi, M., Quadrana, M., and Cremonesi, P. (2018). Using visual features based on mpeg-7 and deep learning for movie recommendation, *International Journal of Multimedia Information Retrieval* .

Deshpande, M. and Karypis, G. (2004). Item-based top-n recommendation algorithms, *ACM Trans. Inf. Syst.* **22**, 1, pp. 143–177, doi:10.1145/963770.963776, http://doi.acm.org/10.1145/963770.963776.

Ekstrand, M. D., Harper, F. M., Willemsen, M. C., and Konstan, J. A. (2014). User perception of differences in recommender algorithms, in *Proceedings of the 8th ACM Conference on Recommender systems* (ACM), pp. 161–168.

Elahi, M., Braunhofer, M., Ricci, F., and Tkalcic, M. (2013). Personality-based active learning for collaborative filtering recommender systems, in *Congress of the Italian Association for Artificial Intelligence* (Springer), pp. 360–371.

Elahi, M., Deldjoo, Y., Bakhshandegan Moghaddam, F., Cella, L., Cereda, S., and Cremonesi, P. (2017). Exploring the semantic gap for movie recommendations, in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17 (ACM, New York, NY, USA), ISBN 978-1-4503-4652-8, pp. 326–330, doi:10.1145/3109859.3109908, http://doi.acm.org/10.1145/3109859.3109908.

Elahi, M., Ricci, F., and Repsys, V. (2011). System-wide effectiveness of active learning in collaborative filtering, in *Proceedings of the International Workshop on Social Web Mining, Co-located with IJCAI, Barcelona, Spain (July 2011)*.

Elahi, M., Ricci, F., and Rubens, N. (2012). Adapting to natural rating acquisition with combined active learning strategies, in *ISMIS'12: Proceedings of the 20th international conference on Foundations of Intelligent Systems* (Springer-Verlag, Berlin, Heidelberg), ISBN 978-3-642-34623-1, pp. 254–263, doi:http://dx.doi.org/10.1007/978-3-642-34624-8_30.

Elahi, M., Ricci, F., and Rubens, N. (2014). Active learning strategies for rating elicitation in collaborative filtering: A system-wide perspective, *ACM Transactions on Intelligent Systems and Technology* **5**, 1, pp. 13:1–13:33, doi:10.1145/2542182.2542195.

Elahi, M., Ricci, F., and Rubens, N. (2016). A survey of active learning in collaborative filtering recommender systems, *Comput. Sci. Rev.* **20**, C, pp. 29–50, doi:10.1016/j.cosrev.2016.05.002, http://dx.doi.org/10.1016/j.cosrev.2016.05.002

Enrich, M., Braunhofer, M., and Ricci, F. (2013). Cold-start management with cross-domain collaborative filtering and tags, in *E-Commerce and Web Technologies - 14th International Conference, EC-Web 2013, Prague, Czech Republic, August 27-28, 2013. Proceedings*, pp. 101–112.

Fernandez Tobias, I., Braunhofer, M., Elahi, M., Ricci, F., and Ivan, C. (2016). Alleviating the new user problem in collaborative filtering by exploiting personality information, *User Modeling and User-Adapted Interaction (UMUAI)* **26**, Personality in Personalized Systems, doi:10.1007/s11257-016-9172-z.

Fernández-Tobías, I. and Cantador, I. (2014). Exploiting social tags in matrix factorization models for cross-domain collaborative filtering, in *CBRecSys@RecSys*, pp. 34–41.

Fernández-Tobías, I., Tomeo, P., Cantador, I., Di Noia, T., and Di Sciascio, E. (2016). Accuracy and diversity in cross-domain recommendations for cold-start users with positive-only feedback, in *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16 (ACM, New York, NY, USA), ISBN 978-1-4503-4035-9, pp. 119–122, doi:10.1145/2959100.2959175, http://doi.acm.org/10.1145/2959100.2959175

Ge, M., Elahi, M., Fernaández-Tobías, I., Ricci, F., and Massimo, D. (2015). Using tags and latent factors in a food recommender system, in *Proceedings of the 5th International Conference on Digital Health 2015*, DH '15 (ACM, New York, NY, USA), ISBN 978-1-4503-3492-1, pp. 105–112, doi:10.1145/2750511.2750528.

GroupLens (2018). Eachmovie, https://grouplens.org/datasets/eachmovie/

Guo, G., Zhang, J., Sun, Z., and Yorke-Smith, N. (2015). Librec: A java library for recommender systems, in *UMAP Workshops*.

Gurbanov, T. and Ricci, F. (2017). Action prediction models for recommender systems based on collaborative filtering and sequence mining hybridization, in *Proceedings of the Symposium on Applied Computing*, SAC '17 (ACM, New York, NY, USA), ISBN 978-1-4503-4486-9, pp. 1655–1661, doi:10.1145/3019612.3019759, http://doi.acm.org/10.1145/3019612.3019759

Gurbanov, T., Ricci, F., and Ploner, M. (2016). Modeling and predicting user actions in recommender systems, in *Proceedings of the 2016 Conference*

on User Modeling Adaptation and Personalization, UMAP '16 (ACM, New York, NY, USA), ISBN 978-1-4503-4368-8, pp. 151–155, doi:10.1145/2930238.2930284, http://doi.acm.org/10.1145/2930238.2930284.

Harper, F. M. and Konstan, J. A. (2015). The movielens datasets: History and context, *ACM Trans. Interact. Intell. Syst.* **5**, 4, pp. 19:1–19:19, doi:10.1145/2827872, http://doi.acm.org/10.1145/2827872.

Harper, F. M. and Konstan, J. A. (2016). The movielens datasets: History and context, *ACM Transactions on Interactive Intelligent Systems (TiiS)* **5**, 4, p. 19.

Hawalah, A. and Fasli, M. (2015). Dynamic user profiles for web personalisation, *Expert Syst. Appl.* **42**, 5, pp. 2547–2569, doi:10.1016/j.eswa.2014.10.032, http://dx.doi.org/10.1016/j.eswa.2014.10.032.

He, R. and McAuley, J. (2016). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering, in *Proceedings of the 25th International Conference on World Wide Web*, WWW '16 (International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland), ISBN 978-1-4503-4143-1, pp. 507–517, doi:10.1145/2872427.2883037, https://doi.org/10.1145/2872427.2883037.

Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. (1999). An algorithmic framework for performing collaborative filtering, in *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99 (ACM, New York, NY, USA), ISBN 1-58113-096-1, pp. 230–237, doi:10.1145/312624.312682, http://doi.acm.org/10.1145/312624.312682.

Hu, R. and Pu, P. (2009). A comparative user study on rating vs. personality quiz based preference elicitation methods, in *Proceedings of the 14th international conference on Intelligent user interfaces*, IUI '09 (ACM, New York, NY, USA), ISBN 978-1-60558-168-2, pp. 367–372, doi:10.1145/1502650.1502702.

Hu, R. and Pu, P. (2011). Enhancing collaborative filtering systems with personality information, in *Proceedings of the fifth ACM conference on Recommender systems*, RecSys '11 (ACM, New York, NY, USA), ISBN 978-1-4503-0683-6, pp. 197–204, doi:10.1145/2043932.2043969.

Hu, Y., Koren, Y., and Volinsky, C. (2008a). Collaborative filtering for implicit feedback datasets, in *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08 (IEEE Computer Society, Washington, DC, USA), ISBN 978-0-7695-3502-9, pp. 263–272, doi:10.1109/ICDM.2008.22, http://dx.doi.org/10.1109/ICDM.2008.22.

Hu, Y., Koren, Y., and Volinsky, C. (2008b). Collaborative filtering for implicit feedback datasets, in *Proceedings of the 8th IEEE International Conference on Data Mining* (IEEE), pp. 263–272.

IMDB (2018). Internet movie database: Movies, tv and celebrities, http://www.imdb.com/.

Isinkaye, F., Folajimi, Y., and Ojokoh, B. (2015). Recommendation systems: Principles, methods and evaluation, *Egyptian Informatics Journal* **16**, 3, pp. 261–273.

Jones, N., Brun, A., and Boyer, A. (2011). Comparisons instead of ratings: Towards more stable preferences, in *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '11 (IEEE Computer Society, Washington, DC, USA), ISBN 978-0-7695-4513-4, pp. 451–456, doi:10.1109/WI-IAT.2011.13, http://dx.doi.org/10.1109/WI-IAT.2011.13.

Kalloori, S., Ricci, F., and Tkalcic, M. (2016). Pairwise preferences based matrix factorization and nearest neighbor recommendation techniques, in *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16 (ACM, New York, NY, USA), ISBN 978-1-4503-4035-9, pp. 143–146, doi:10.1145/2959100.2959142, http://doi.acm.org/10.1145/2959100.2959142.

Kim, D., Park, C., Oh, J., Lee, S., and Yu, H. (2016). Convolutional matrix factorization for document context-aware recommendation, in *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16 (ACM, New York, NY, USA), ISBN 978-1-4503-4035-9, pp. 233–240, doi:10.1145/2959100.2959165, http://doi.acm.org/10.1145/2959100.2959165.

Koren, Y. (2008). Factorization meets the neighborhood: A multifaceted collaborative filtering model, in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08 (ACM, New York, NY, USA), ISBN 978-1-60558-193-4, pp. 426–434, doi:10.1145/1401890.1401944, http://doi.acm.org/10.1145/1401890.1401944.

Koren, Y. (2009). The bellkor solution to the netflix grand prize, *Netflix prize documentation* **81**, pp. 1–10.

Koren, Y. and Bell, R. (2011). *Advances in Collaborative Filtering* (Springer US, Boston, MA), ISBN 978-0-387-85820-3, pp. 145–186, doi:10.1007/978-0-387-85820-3_5, https://doi.org/10.1007/978-0-387-85820-3_5.

Koren, Y. and Bell, R. (2015). Advances in collaborative filtering, in *Recommender Systems Handbook* (Springer), pp. 77–118, doi:10.1007/978-1-4899-7637-6_3.

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

Kosinski, M., Matz, S. C., Gosling, S. D., Popov, V., and Stillwell, D. (2015). Facebook as a research tool for the social sciences: Opportunities, challenges, ethical considerations, and practical guidelines. *American Psychologist* **70**, 6, p. 543.

Košir, A., Odic, A., Kunaver, M., Tkalcic, M., and Tasic, J. F. (2011). Database for contextual personalization, *Elektrotehniški vestnik* **78**, 5, pp. 270–274.

Librarything (2018). Catalog your books online 2018, https://www.librarything.com/.

Linden, G., Smith, B., and York, J. (2003). Amazon.com recommendations: Item-to-item collaborative filtering, *IEEE Internet Computing* **7**, 1, pp. 76–80, doi:10.1109/MIC.2003.1167344.

Loepp, B., Hussein, T., and Ziegler, J. (2014). Choice-based preference elicitation for collaborative filtering recommender systems, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (ACM), pp. 3085–3094.

Manzato, M. G. (2012). Discovering latent factors from movies genres for enhanced recommendation, in *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12 (ACM, New York, NY, USA), ISBN 978-1-4503-1270-7, pp. 249–252, doi:10.1145/2365952.2366006, http://doi.acm.org/10.1145/2365952.2366006

Manzato, M. G. (2013). gsvd++: Supporting implicit feedback on recommender systems with metadata awareness, in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13 (ACM, New York, NY, USA), ISBN 978-1-4503-1656-9, pp. 908–913, doi:10.1145/2480362.2480536, http://doi.acm.org/10.1145/2480362.2480536

Massimo, D., Elahi, M., Ge, M., and Ricci, F. (2017). Item contents good, user tags better: Empirical evaluation of a food recommender system, in *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization* (ACM), pp. 373–374.

McNee, S. M., Lam, S. K., Konstan, J. A., and Riedl, J. (2003). Interfaces for eliciting new user preferences in recommender systems, in *Proceedings of the 9th international conference on User modeling*, UM'03 (Springer-Verlag, Berlin, Heidelberg), ISBN 3-540-40381-7, pp. 178–187, http://dl.acm.org/citation.cfm?id=1759957.1759988

Neidhardt, J., Schuster, R., Seyfang, L., and Werthner, H. (2014). Eliciting the users' unknown preferences, in *Proceedings of the 8th ACM Conference on Recommender systems* (ACM), pp. 309–312.

Nicholas, I. S. C. and Nicholas, C. K. (1999). Combining content and collaboration in text filtering, in *In Proceedings of the IJCAI'99 Workshop on Machine Learning for Information Filtering*, pp. 86–91.

Oard, D. W., Kim, J., *et al.* (1998). Implicit feedback for recommender systems, in *Proceedings of the AAAI workshop on recommender systems*, pp. 81–83.

Odić, A., Tkalčič, M., Tasič, J. F., and Košir, A. (2012). Relevant context in a movie recommender system: Users' opinion vs. statistical detection, *ACM RecSys 2012, Proceedings of the 4th International Workshop on Context-Aware Recommender Systems (CARS 2012)* .

Odić, A., Tkalčič, M., Tasič, J. F., and Košir, A. (2013). Predicting and detecting the relevant contextual information in a movie-recommender system, *Interacting with Computers* , p. iws003.

Pan, R., Zhou, Y., Cao, B., Liu, N. N., Lukose, R., Scholz, M., and Yang, Q. (2008). One-class collaborative filtering, in *Proceedings of the 8th IEEE International Conference on Data Mining* (IEEE), pp. 502–511.

Park, D. H., Kim, H. K., Choi, I. Y., and Kim, J. K. (2012). A literature review and classification of recommender systems research, *Expert Systems with Applications* **39**, 11, pp. 10059–10072.

Picault, J. and Ribiere, M. (2008). An empirical user profile adaptation mechanism that reacts to shifts of interests, in *Proceedings of the European conference in artificial intelligence (ECAI)*.

Picault, J., Ribiere, M., Bonnefoy, D., and Mercer, K. (2011). How to get the recommender out of the lab? in *Recommender Systems Handbook* (Springer), pp. 333–365.

Pu, P., Chen, L., and Hu, R. (2012). Evaluating recommender systems from the user's perspective: survey of the state of the art, *User Modeling and User-Adapted Interaction* **22**, 4-5, pp. 317–355, doi:10.1007/s11257-011-9115-7.

Rashid, A. M., Albert, I., Cosley, D., Lam, S. K., McNee, S. M., Konstan, J. A., and Riedl, J. (2002). Getting to know you: Learning new user preferences in recommender systems, in *Proceedings of the 7th International Conference on Intelligent User Interfaces*, IUI '02 (ACM, New York, NY, USA), ISBN 1-58113-459-2, pp. 127–134, doi:10.1145/502716.502737, http://doi.acm.org/10.1145/502716.502737

Rashid, A. M., Karypis, G., and Riedl, J. (2008a). Learning preferences of new users in recommender systems: An information theoretic approach, *SIGKDD Explor. Newsl.* **10**, 2, pp. 90–100, doi:10.1145/1540276.1540302, http://doi.acm.org/10.1145/1540276.1540302

Rashid, A. M., Karypis, G., and Riedl, J. (2008b). Learning preferences of new users in recommender systems: an information theoretic approach, *SIGKDD Explor. Newsl.* **10**, pp. 90–100, doi:10.1145/1540276.1540302.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. (2009). Bpr: Bayesian personalized ranking from implicit feedback, in *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence* (AUAI Press), pp. 452–461.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews, in *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94 (ACM, New York, NY, USA), ISBN 0-89791-689-1, pp. 175–186, doi:10.1145/192844.192905, http://doi.acm.org/10.1145/192844.192905

Ricci, F., Rokach, L., and Shapira, B. (2015). Recommender systems: Introduction and challenges, in *Recommender Systems Handbook* (Springer US), pp. 1–34, doi:10.1007/978-1-4899-7637-6_1.

Rubens, N., Elahi, M., Sugiyama, M., and Kaplan, D. (2015). Active learning in recommender systems, in *Recommender Systems Handbook - chapter 24: Recommending Active Learning* (Springer US), pp. 809–846, doi:10.1007/978-1-4899-7637-6_24.

Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms, in *Proceedings of the 10th international conference on World Wide Web* (ACM), pp. 285–295.

Schedl, M., Zamani, H., Chen, C.-W., Deldjoo, Y., and Elahi, M. (2018). Current challenges and visions in music recommender systems research, *International Journal of Multimedia Information Retrieval* doi:10.1007/s13735-018-0154-2, https://doi.org/10.1007/s13735-018-0154-2

Schein, A. I., Popescul, A., Ungar, L. H., and 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* (ACM, New York, NY, USA), ISBN 1-58113-561-0, pp. 253–260, doi:10.1145/564376.564421.

Shardanand, U. and Maes, P. (1995). Social information filtering: Algorithms for

automating “word of mouth”, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95 (ACM Press/Addison-Wesley Publishing Co., New York, NY, USA), ISBN 0-201-84705-1, pp. 210–217, doi:10.1145/223904.223931, http://dx.doi.org/10.1145/223904.223931

Shi, Y., Larson, M., and Hanjalic, A. (2014). Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges, *ACM Comput. Surv.* **47**, 1, pp. 3:1–3:45, doi:10.1145/2556270, http://doi.acm.org/10.1145/2556270

Stern, D. H., Herbrich, R., and Graepel, T. (2009). Matchbox: Large scale online bayesian recommendations, in *Proceedings of the 18th International Conference on World Wide Web*, WWW '09 (ACM, New York, NY, USA), ISBN 978-1-60558-487-4, pp. 111–120, doi:10.1145/1526709.1526725, http://doi.acm.org/10.1145/1526709.1526725

Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques, *Adv. in Artif. Intell.* **2009**, pp. 4:2–4:2, doi:10.1155/2009/421425, http://dx.doi.org/10.1155/2009/421425

Sun, M., Li, F., Lee, J., Zhou, K., Lebanon, G., and Zha, H. (2013). Learning multiple-question decision trees for cold-start recommendation, in *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13 (ACM, New York, NY, USA), ISBN 978-1-4503-1869-3, pp. 445–454, doi:10.1145/2433396.2433451.

Tkalcic, M., Kosir, A., and Tasic, J. (2013). The ldos-peraff-1 corpus of facial-expression video clips with affective, personality and user-interaction metadata, *Journal on Multimodal User Interfaces* **7**, 1-2, pp. 143–155, doi:10.1007/s12193-012-0107-7.

Trevisiol, M., Aiello, L. M., Schifanella, R., and Jaimes, A. (2014). Cold-start news recommendation with domain-dependent browse graph, in *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14 (ACM, New York, NY, USA), ISBN 978-1-4503-2668-1, pp. 81–88, doi:10.1145/2645710.2645726.

TripAdvisor (2018). Read reviews, compare prices and book, 2018, https://www.tripadvisor.com/

Tu, M., Chang, Y.-K., and Chen, Y.-T. (2016). A context-aware recommender system framework for iot based interactive digital signage in urban space, in *Proceedings of the Second International Conference on IoT in Urban Space*, Urb-IoT '16 (ACM, New York, NY, USA), ISBN 978-1-4503-4204-9, pp. 39–42, doi:10.1145/2962735.2962736, http://doi.acm.org/10.1145/2962735.2962736

Vargas-Govea, B., González-Serna, G., and Ponce-Medellın, R. (2011). Effects of relevant contextual features in the performance of a restaurant recommender system, *ACM RecSys* **11**.

Vartak, M., Thiagarajan, A., Miranda, C., Bratman, J., and Larochelle, H. (2017). A meta-learning perspective on cold-start recommendations for items, in *Advances in Neural Information Processing Systems*, pp. 6907–6917.

Vozalis, M. and Margaritis, K. G. (2004). Collaborative filtering enhanced by demographic correlation, in *in Proceedings of the AIAI Symposium on Professional Practice in AI, part of the 18th World Computer Congress*, pp. 293–402.

Zhao, X., Zhang, W., and Wang, J. (2013). Interactive collaborative filtering, in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, CIKM '13 (ACM, New York, NY, USA), ISBN 978-1-4503-2263-8, pp. 1411–1420, doi:10.1145/2505515.2505690, http://doi.acm.org/10.1145/2505515.2505690.

Zheng, Y., Burke, R., and Mobasher, B. (2012). Differential context relaxation for context-aware travel recommendation, in *E-Commerce and Web Technologies* (Springer), pp. 88–99.

Zheng, Y., Burke, R., and Mobasher, B. (2013). Recommendation with differential context weighting, in *User Modeling, Adaptation, and Personalization* (Springer), pp. 152–164.

Zheng, Y., Mobasher, B., and Burke, R. (2014). Cslim: Contextual slim recommendation algorithms, in *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14 (ACM, New York, NY, USA), ISBN 978-1-4503-2668-1, pp. 301–304, doi:10.1145/2645710.2645756, http://doi.acm.org/10.1145/2645710.2645756.

Zheng, Y., Mobasher, B., and Burke, R. (2015). Integrating context similarity with sparse linear recommendation model, in *User Modeling, Adaptation and Personalization* (Springer), pp. 370–376.

View publication stats