## A Review on Recommendation Systems: Context-aware to Social-based

#### S.M. Mahdi Seyednezhad

Department of Computer Engineering and Sciences Florida Institute of Technology Melbourne, Florida, USA. sseyednezhad2013@my.fit.edu

#### Kailey Nobuko Cozart

Department of Mathematics and Computer Science
Whitworth University
Spokane, Washington, USA
cozart21@my.whitworth.edu

#### John Anthony Bowllan

Department of Mathematics Middlebury College Middlebury, Vermont, USA jbowllan@middlebury.edu

#### Anthony O. Smith

Department of Computer Engineering and Sciences Florida Institute of Technology Melbourne, Florida, USA. anthonysmith@fit.edu

November 30, 2018

## Contents

$\mathbf{A}$	bstra	ct	3
1	Intr	roduction	4
2	Rec	ommender System Methods and Evaluation	8
	2.1	Demographic Filtering	9
	2.2	Content-based Filtering	9
		2.2.1 Keyword-based vector space model	11
	2.3	Collaborative Filtering	13
	2.4	Hybrid Methods	15
	2.5	Evaluation Criteria	17
		2.5.1 Quality of the prediction	17
		2.5.2 Quality of the set of recommendations	18
		2.5.3 Quality of the list of recommendations	20
		2.5.4 Novelty and diversity	20
3	Cor	atext-aware Recommender Systems	<b>22</b>
	3.1	Context in Recommender Systems	23
	3.2	Obtaining contextual information	26
	3.3	Utilizing Context in Recommender Systems	28
		3.3.1 Contextual Pre-filtering	28
		3.3.2 Contextual Post-filtering	30
		3.3.3 Contextual Modeling	32
4	$\mathbf{Soc}$	ial-based Recommender Systems	34
	4.1	· ·	35
	4.2		38
	4.3		39

4.4	Immediate Friend Inference	40
4.5	Link Prediction for Social Networks	43
Acknow	wledgments	45
Bibliog	graphy	45

### Abstract

The number of Internet users has grown rapidly enticing companies and cooperations to make full use of recommendation infrastructures. Consequently, online advertisement companies emerged to aid us in the presence of numerous items and users. Even as a user, you may find yourself drowned in a set of items that you think you might need, but you are not sure if you should try them. Those items could be online services, products, places or even a person for a friendship. Therefore, we need recommender systems that pave the way and help us making good decisions. This paper provides a review on traditional recommendation systems, recommendation system evaluations and metrics, context-aware recommendation systems, and social-based recommendation systems. While it is hard to include all the information in a brief review paper, we try to have an introductory review over the essentials of recommendation systems. More detailed information on each chapter will be found in the corresponding references. For the purpose of explaining the concept in a different way, we provided slides available on Slideshare.

## Chapter 1

### Introduction

It is unbelievable that the human brain has evolved to deal with our complex world. However, this same world has recently been augmented by technology and humans are dependent on such technology to perform daily tasks. This vicious cycle means that humans now require outside help to make sense of the world, in particular input from others regarding their previous experience; that is, their recommendations.

Surprisingly, during ancient civilizations (4000 - 1200 BC), humans needed recommendations. They could be used for problems such as what crops to cultivate, the appropriate time of cultivation, what religion to follow, etc. After that, in old times (and probably nowadays) families used to recommend acquaintances to each other for arranged marriages. Currently, people ask for recommendations regarding many aspects of modern life such as travel, music, movies, etc. The idea of recommendation systems began to be more important after the industrial revolution in which the number of available goods grew enormously and it became vital when computers changed the global market [50].

By 2015, the number of Internet users had grown from 738 million in 2000 to 3.2 billion [54], meaning that 43% of the world population was using online services, enticing companies and cooperations to make full use of

recommendation infrastructures. Consequently, online advertisement companies emerged to aid us in the presence of numerous items and users. Even as a user, you may find yourself drowned in a set of items that you think you might need, but you are not sure if you should try them. Those items could be online services, products, places or even a person for a friendship. Therefore, we need recommender systems that pave the way and help us making good decisions.

Recommender systems have attracted the attention of a significant number of popular Internet sites, such as Amazon.com, YouTube, Netflix, Spotify, LinkedIn, Facebook, Tripadvisor and IMDB [31]. Particularly, many media companies offer practical recommender systems to their subscribers. Based on the type of applications, there are various purposes for a recommender system, including but not limited to, increasing the number of items sold, selling more diverse items, and increasing user satisfaction and fidelity [43].

Recommender systems (RSs) collect information from users' preference for a set of items and predict the best desired items for them [8]. The information can be obtained explicitly by recording users' ratings or implicitly by observing users' behavior. Generating a recommender system depends on a set of considerations, such as type of available data, filtering algorithms, models, techniques, sparsity level of data and desired quality [8]. Some recommender systems are designed for a specific task. For instance, Guan et al. [23] introduced a recommender system for apparel. Recommender systems found their way to becoming an independent research area in the mid 1970s at Duke University [50].

Recommender systems are used with a lot of information about items, users, and ratings. In an information filtering system, unwanted information is removed by using computerized methods prior to presentation to the users. Its main goal is the to manage the information overload and to increase the semantic signal-to-noise ratio [25]. In fact, a recommender system needs to filter information in order to find more relevant items for the users.

Demographic filtering, content-based filtering, collaborative filtering and hybrid methods are the main four methods of recommender systems [10, 2]. Among them, collaborative filtering (CF) and the methods combining with it are the most popular ones because they are based on user ratings [45, 28]. Content-based filtering is based on content of the items that the users liked in the past [55]. For example, if the user tried a science fiction movie in the past, the recommender system will most likely recommend a recent science fiction movie [42]. This method is popular for websites such as IMDB, Rotten Tomatoes, and Pandora. On the other hand, in demographic filtering, the recommender system observes the common attributes of the users (gender, age, location) and suggests items to the users with similar attributes. This is based on the principle that people with some specific common attributes may have common interest.

When we want to take into account user ratings, we should make use of collaborative filtering. These systems try to predict the utility of items for a specific user based on the items that the user previously rated [2]. These days, massive online companies such as Amazon, Facebook, Twitter, LinkedIn, Spotify, Google News and Last.fm employ this technique. The most widely used algorithm for collaborative filtering is k Nearest Neighbors (kNN) [7]; its application is based on two main approaches: user to user and item to item. In the user to user version, the kNN algorithm first tries to determine the k neighborhood for the user; then, it aggregates users based on their ratings and finally predicts based on the aggregated information. Gong [22] uses both items and users to implement a bi-cluster method for a new recommender system. The major pitfalls of using the kNN algorithm for recommender systems are the high level of sparsity in RS datasets and its low scalability [34].

Cold start is one of the main challenges that almost all recommender systems face when the initial ratings or any knowledge about user experience is not sufficient. There are three major cold-start problems [8]: new commu-

nity, new user and new item. In the new community problem, the RS suffers from a lack of sufficient data when it is initialized for a new community. This becomes even harder when the RS uses a pure collaborative filtering based on community preferences [47]. In Chapter 2, we provide more details about recommender system filtering methods. In Chapter 3, we talk about the concept of context in recommender systems and the major approaches for deploying the contextual information in a recommender system. The emergence of online social networks raises the concept of context. In Chapter 4, we provide additional information on recommender systems when the social information plays an important role.

## Chapter 2

# Recommender System Methods and Evaluation

As we mentioned in Chapter 1, there are three traditional main recommendation methods: demographic filtering, content based filtering and collaborative filtering [42]; among them, demographic and content-based filtering have been the most popular ones. However, most of the big companies primarily use a hybrid approach which is a combination of the aforementioned methods [8]. In this chapter, we explain these methods in more detail and discuss each one's virtues and drawbacks.

In general, the information about an active user's feedback is crucial for recommender systems and it is obtained explicitly or implicitly. Explicit feedback is the users' ratings on items, and it can be considered as direct information about feedback. The main advantage of this type of feedback is its simplicity; nonetheless, the drawback is the need for active users to rate items. Unfortunately, some users do not rate items. On the other hand, implicit feedback is extracted by monitoring user behavior and analyzing user activity. For example, if a user tries action movies frequently, the implicit information implies that the user's rating on action movies could be high. In the case that an item is a document, then, printing, saving, reading or

bookmarking could be the reflect of user interest in that document. The distinct advantage of this method is that there is no need for an active user to rate items; however, sometimes biasing can happen. For example, if you are interrupted by a phone call while you are opening a document, the recommender system may judge you as a fan of that document, which may not be true.

#### 2.1 Demographic Filtering

This type of recommender systems suggests items based on the demographic profile of users. It can be used to identify the taste of users that belong to a certain community. Therefore, to design these systems, we need some information about users to categorize them into groups. Then, if some users in a particular group like or order an item, it is possible that the other users of this group tend to do the same. It should be noted that although it might be better to use more structured information about users, there is a trade-off between the computational complexity and the quality of demographic filtering. Pazzani [42] ran an experiment based on demographic filtering on data about restaurants and he claimed that on average, 57.5% of the top three recommended restaurants were liked by users. Table 2.1 shows an example of people who rated a specific restaurant. It tells us that a female in an area in which the code is 714 is probably going to like the restaurant.

#### 2.2 Content-based Filtering

Content-based methods make recommendations based on the description of the items. Nowadays, it is combined with other methods and use more information about items and users. However, several algorithms have been proposed to analyze the content of a document. For example, we consider the case in which a recommender system is designed with the content-based

Table 2.1: Demographic information on the users who rated a specific restaurant.

User	Gender	Age	Area Code	Education	Employed	Rate
Karen	F	15	714	HS	F	+
Lynn	F	17	714	HS	$\mathbf{F}$	_
Chris	$\mathbf{F}$	27	714	$^{\mathrm{C}}$	${ m T}$	+
Mike	${ m M}$	40	714	$^{\mathrm{C}}$	${ m T}$	_
Jill	$\mathbf{F}$	10	714	${ m E}$	$\mathbf{F}$	?

method to recommend movies to users. We may assume that the movie description has been already extracted. If the movie is an action film and a user liked it, the recommender system will recommend another action movie to the user.

Content-based recommender systems (CBRS) consist of three major parts from a high level architectural point of view [35]; first it does the preprocessing on items with a *content analyzer*, then a *profile learner* learns about users. Finally, the *filtering component* finds a set of appropriate recommendations. More details for these three parts are provided as follows:

- Content Analyzer For any decision making problem, the raw data should be pre-processed to extract featured information. Here, the output of this pre-processing part is the structured relevant information. The content analyzer prepares information for the next step. It transforms information from its original format to one that is more abstract and useful. For example, it may receive a web page as input and convert it to a vector of keywords.
- **Profile Learner** This module is specifically designed for the user side. It receives the pre-processed information from the content analyzer and generalizes them to construct the user preferences. The generalization step models the user interest based on the user's past ratings

of items. For example, a profile learner in a web page recommender system may combine the vector of positive and negative examples to construct a *prototype* item vector that represents the user profile [46].

• Filtering Component — This is the final part that finds the relevant items based on the user profile and recommends them to the user. It uses a similarity measure (e.g. cosine similarity) between items and the user prototype.

Content-based recommender systems are mainly used where the item is either a document or a text used to describe an item. Thus, text mining methods play an important role in content-based recommender systems. The traditional methods are highly sensitive to the way that documents are represented [42]. However, the following technique is widely used to analyze the content of a text document and turn it into a vector.

#### 2.2.1 Keyword-based vector space model

The Vector Space Model (VSM) is one of the spatial representations of text documents. It transforms a text document to a n-dimensional space (i.e. a vector with n elements) in which each dimension (element) is a term in the given document collection. This method needs to weight the terms and calculate the similarity of documents based on those weights. The most commonly used weighting method for terms is TF-IDF (Term Frequency Inverse Document Frequency) weighting, which is based on information extracted from text.

In TF-IDF, terms that are frequently found in one text (TF), but rarely in other documents (IDF), will be possibly more related to the topic of that text [35]. Additionally, the weight is normalized to give the same chance of being retrieved to both small and large documents. Equation 2.1 shows TF

based on term frequencies.

$$TF(t_k, d_j) = \frac{f_{k,j}}{max\{f_{z,j}\}}$$
 (2.1)

where  $t_k$  denotes the kth term in the dictionary of terms  $T = \{t_1, t_2, t_3, ..., t_n\}$ ,  $d_j$  is a document from the document collection  $D = \{d_1, d_2, d_3, ..., d_N\}$ ,  $f_{k,j}$  is the frequency of term  $t_k$  in document  $d_j$ , and  $max\{f_{z,j}\}$  is the maximum of all the frequencies of all terms in document  $d_j$ . Moreover, we have equation 2.2 to calculate the IDF based on the size of the collection and the documents with a particular term  $t_k$ .

$$IDF(t_k) = \log \frac{N}{n_k} \tag{2.2}$$

where N is the number of documents (collection size), and  $n_k$  is the number of documents in which the term  $t_k$  has been seen at least once.

Equation 2.3 uses the obtained TF and IDF to calculate the TF-IDF for each term in each document.

$$TF\text{-}IDF(t_k, d_j) = TF(t_k, d_j) \cdot IDF(t_k)$$
(2.3)

Now, we need to normalize the weights to be in [0,1] and to have the vectors with the same length. Equation 2.4 does the cosine normalization for this purpose.

$$w_{k,j} = \frac{TF\text{-}IDF(t_k, d_j)}{\sqrt{\sum_{s=1}^{|T|} TF\text{-}IDF(t_k, d_j)^2}}$$
(2.4)

where  $w_{k,j}$  denotes the weight corresponding to term  $t_k$  in document  $d_j$ .

One similarity measure is needed to test the closeness of two documents. Equation 2.5 calculates the similarity between documents  $d_i$  and  $d_j$  using

cosine similarity, which is common in this field.

$$sim(d_i, d_j) = \frac{\sum_k w_{k,i} \cdot w_{k,j}}{\sqrt{\sum_k w_{k,i}^2} \cdot \sqrt{\sum_k w_{k,j}^2}}$$
(2.5)

In content-based filtering methods that use VSM, both user profiles and items are represented by vectors of weighted terms [35]. Recently, semantic aware methods have attracted the attention of scholars who have been working on content based recommender systems.

#### 2.3 Collaborative Filtering

We consider a recommender system for movies. The RS may face a situation in which we do not know a particular movie's features, but we know how some specific users rated it. Now, if two users named "Marcos" and "Diego" like a movie titled A, and later Marcos watches another movie titled B and likes it, then we can recommend this movie to Diego. This approach is adopted from the collaborative filtering method.

In collaborative filtering, the recommender system looks for similarity between users to make predictions. In several cases, the pattern of ratings of users is a useful feature to determine similarity [42]. Normally, collaborative filtering recommendation methods use patterns of ratings or usage to recommend items specified for users without need for extra information about either users or items [60]. Similarly to other recommendation methods, CF methods must relate items and users which are two essential different entities. The *Neighborhood approach* is a technique that concentrates on how items or users are related among themselves. For instance, in an item-item approach, the RS models the preference of a user to an item with respect to the previous rating of the same user to a similar item. Another technique is the *latent factor model*, such as matrix factorization, which transforms both items and users to the same latent factor space.

Additionally, there has been a different point of view to categorize the collaborative filtering techniques by dividing it into memory-based and *model-based* methods. Memory-based methods act only on a user-item rating matrix and can easily be adapted to use all the ratings before the filtering process; thus, its results are updated. On the other hand, a model based system, like a neural network, generates a model that learns from the information of user-item ratings and recommends new items [44].

In memory-based CF methods, measuring the similarity plays a significant role, because the RS either tries to find the similarity between items or the similarity between users [52]. It needs to find the similarity between items to see what a user's opinion is of items and what the closest new/unseen/unknown item is to the items that the user has already liked, following which a recommendation can be made. Likewise, it needs the similarity between users to see what are the close users, and if a user tries a new item, the RS recommends it to the users close to her.

Among various similarity measures, we mention the *Pearson Correlation* measure, which reveals the information on how much two variables are linearly related to each other. Equation 2.7 calculates the Pearson correlation between user u and v which gives us the information about the similarity of users who both rated the same item.

$$w_{u,v} = \frac{\sum_{i \in I_{u,v}} (r_{u,i} - \bar{r}_u) \cdot (r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{u,v}} (r_{u,i} - \bar{r}_u)^2} \cdot \sqrt{\sum_{i \in I_{u,v}} (r_{v,i} - \bar{r}_v)^2}}$$
(2.6)

where  $I_{u,v}$  is the set of items that both users u and v rated,  $\bar{r}_u$  is the average rating of items rated by user u in  $I_{u,v}$ , and  $r_{u,i}$  is the rating of user u on item i.

In a similar fashion, by using Pearson Correlation in equation 2.7, we can

calculate the similarity of items i and j that were rated by users:

$$w_{i,j} = \frac{\sum_{u \in U_{i,j}} (r_{u,i} - \bar{r}_i) \cdot (r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in U_{i,j}} (r_{u,i} - \bar{r}_i)^2} \cdot \sqrt{\sum_{u \in U_{i,j}} (r_{u,j} - \bar{r}_j)^2}}$$
(2.7)

where  $\bar{r}_j$  is the average rating of item i, and  $U_{i,j}$  is the set of users that rated both items i and j.

After computing the item-item and user-user similarity, the RS job is to predict a rating on a particular item from a certain user. In a neighborhood-based model, a nearest neighbor should be picked to be involved in predicting the ratings. If we assume that we have an active user a and that the RS needs to predict the user's rating on item i. In weighted sum of others' ratings, the predicted rating is calculated by equation 2.8.

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u \in U} (r_{u,i} - \bar{r}_u) \cdot w_{a,u}}{\sum_{u \in U} |w_{a,u}|}$$
(2.8)

where  $P_{a,i}$  is the predicted rating of user a for item i,  $\bar{r}_u$  is the average rating of user u,  $\bar{r}_a$  is the average rating of item a, and  $w_{a,u}$  is the weight between those two users calculated by equation 2.7. Additionally, we can define a threshold for  $w_{a,u}$  to avoid participating the considerably small weights.

#### 2.4 Hybrid Methods

Each recommendation method has its own virtues and drawbacks. This fact has led scholars to combine them in order to have a recommender system that benefits from those virtues and be able to overcome most of the drawbacks. Finally, researchers came up with the idea of using a hybrid method for recommender systems. A hybrid filtering method may use a combination of collaborative filtering with demographic filtering or collaborative filtering with content-based filtering to have boosted results. For instance, Balabanovic et al. [5] created a recommender system named Fab which extracts user profile

of interest on web pages by content filtering techniques and uses that information for collaborative filtering. Moreover, the hybrid method can involve different recommender systems based on the confidence that they have on predicted ratings or recommendations [9]. Additionally, different techniques from one method can be combined together to create a new hybrid RS.

Predominantly, collaborative filtering has been combined with content-based filtering to make a hybrid method. Babodilla et al. [8] categorized them in four different groups that is shown in figure 2.1. Figure 2.1a indicates the methods that combines CF and CBF with a weighting method. It may rank the items from both and recommend the top best items from them. Figure 2.1b shows the methods that use CBF methods to extract features and send it to CF to make the final recommendation. The example we mentioned from Balabanovic [5] used this technique. Furthermore, the prediction from CBF can be an input of CF as well. In figure 2.1c, a unified model is depicted that utilizes CF and CBF to have their output for another classifier, such as rule based classifier or a probability model. Figure 2.1d depicts a model that uses output from CF for CBF. For example user ratings can help CBF characterize users better.

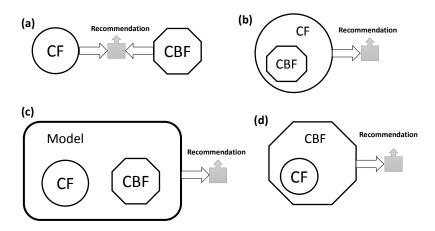


Figure 2.1: Different methods of combining CF and CBF

#### 2.5 Evaluation Criteria

Recommender systems should be evaluated for many reasons, such as comparison of quality of techniques. Evaluation metrics play a significant role in comparing several solutions for the same problem. Evaluation metrics are categorized in four main groups [8]: (a) prediction metrics, such as accuracy (b) set recommendation metrics, such as Precision and Recall (c) rank recommendation metrics like half-life and (d) diversity and novelty.

#### 2.5.1 Quality of the prediction

Quality of prediction is the first criterion that has been used to compare recommender systems. One of the widely used prediction metrics is Mean Absolute Error (MAE) and other metrics derived from it such as mean square error, root mean square error (RMSE), and normalized mean absolute error.

We define U as the set of users of RS, and I as the set of RS items. Then,  $p_{u,i}$  is the prediction of item i on user u,  $r_{u,i}$  is the rating of user u on item i and " $\bullet$ " means user u has not rated item i. Let  $O_u = \{i \in I \mid p_{u,i} \neq \bullet \land r_{u,i} \neq \bullet \}$  be the set of items rated by user u with prediction values. The error is defined as the difference between prediction and real value:  $|p_{u,i} - r_{u,i}|$ . Equations 2.9 and 2.10 show how we should calculate the mean absolute error (MAE) and the root mean square error (RMSE) respectively.

$$MAE = \frac{1}{\#U} \sum_{u \in U} \left( \frac{1}{\#O} \sum_{i \in O_u} |p_{u,i} - r_{u,i}| \right)$$
 (2.9)

where " $\#\{*\}$ " means the number of elements of the set  $\{*\}$  or cardinality of it.

$$RMSE = \frac{1}{\#U} \sum_{u \in U} \sqrt{\frac{1}{\#O} \sum_{i \in O_u} (p_{u,i} - r_{u,i})^2}$$
 (2.10)

Another metric is coverage which can be interpreted as the capacity of predicting from a particular metric [18]. It calculates the percentage of sit-

uations in which at least one out of the k neighbors of each active user rates an item that has not been rated yet by that active user [8]. The total coverage of a recommender system equals to the average of all users' coverage. We define  $K_{u,i}$  as the set of user  $u \in U$  which have rated the item i,  $C_u = \{i \in I \mid r_{u,i} = \bullet \land K_{u,i} \neq \emptyset\}$  as the set of items that have not been rated by user u and at least one of the neighbors rated it, and  $D_u = \{i \in I \mid r_{u,i} = \bullet\}$  as the set of items that have not been rated by user u. We have equation 2.11 to calculate coverage:

$$coverage = \frac{1}{\#U} \sum_{u \in U} \frac{1}{\#O} \sum_{i \in O_u} \left( 100 \times \frac{\#C_u}{\#D_u} \right)$$
 (2.11)

#### 2.5.2 Quality of the set of recommendations

For some users, having a reduced set of items is more important that having one item recommended. Precision, recall and F1 are the most important metrics to evaluate the quality of the set of recommendations. Precision indicates the rate of relevant recommended items to all of the recommended items. Recall is about the rate of relevant recommended items to all of the relevant items and F1 is a combination of precision and recall. We consider  $X_u$  as the set of recommendations to user u, and  $z_n$  as the set of  $z_n$  recommendations to user  $z_n$  we calculate the aforementioned metrics by making  $z_n$  test recommendation to user  $z_n$ . By considering a  $z_n$  as threshold, we have equation 2.12 for Precision.

$$precision = \frac{1}{\#U} \sum_{u \in U} \frac{\#\{i \in Z_u \mid r_{u,i} \ge \theta\}}{n}$$
 (2.12)

Equation 2.12 sums over the number of recommendations and normalizes them. Moreover, equation 2.13 calculates Recall.

$$recall = \frac{1}{\#U} \sum_{u \in U} \frac{\#\{i \in Z_u \mid r_{u,i} \geqslant \theta\}}{\#\{i \in Z_u \mid r_{u,i} \geqslant \theta\} + \#\{i \in Z_u^c \mid r_{u,i} \geqslant \theta\}}$$
(2.13)

where  $\#\{i \in Z_u^c \mid r_{u,i} \ge \theta\}$  denotes the number of relevant items that have not been recommended.

The F1 measure is calculated in equation 2.14.

$$recall = \frac{2 \times precision \times recall}{precision + recall}$$
 (2.14)

In figure 3.2 we depict the role of recall and precision for evaluating recommender systems.

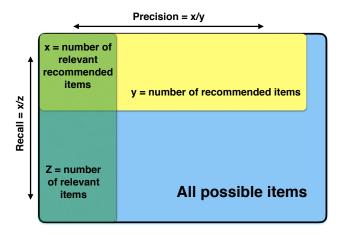


Figure 2.2: Precision and recall with respect to all of the items.

#### 2.5.3 Quality of the list of recommendations

When we have a considerable number of recommendations, users give attention to the first items. Consequently, if the RS makes mistake in the first options, it is going to be a serious mistake. From information retrieval studies, we adopt the ranking measures that have been used in information retrieval can be applied here; half-life and discounted cumulative gain are the most popular measures for recommender systems. Equation 2.15 shows how we should calculate half-life. It assumes that users loses their interest of the following items in the list exponentially.

$$Hl = \frac{1}{\#U} \sum_{u \in U} \sum_{i=1}^{N} \frac{max(r_{u,p_i} - d, 0)}{2^{(i-1)/(\alpha - 1)}}$$
(2.15)

where  $p_1, ..., p_n$  represents the recommendation list,  $r_{u,p_i}$  is the true rating of user u for item  $p_i$ , d is the default rating and  $\alpha$  is the number of items that have 50% chance to be reviewed by user.

Similar to half-time, discounted cumulative gain (DCG) considers a logarithmic decay in users' interest.

$$DCG = \frac{1}{\#U} \sum_{u \in U} \left( r_{u,p_1} + \sum_{i=2}^{k} \frac{r_{u,p_1}}{\log_2(i)} \right)$$
 (2.16)

where k is the rank of recommended items in the list of recommendations.

#### 2.5.4 Novelty and diversity

In some applications the RS needs to recommend novel items, because companies want to sell their new items as well. Further, some users may want to explore a new type of items. Therefore, there should be a metric to compare recommender systems based on this criterion. In this case, we want to know up to what extent a RS can recommend diverse items. Novelty and diversity are two main metrics that are useful here. There is no standard way

to define these metrics and scholars tend to use different ways to calculate them. However, numerous authors used equation 2.17 and 2.18 to calculate diversity and novelty respectively [29]:

$$diversity_{Z_u} = \frac{1}{\#Z(\#Z - 1)} \sum_{i \in Z_u} \sum_{j \in Z_u, j \neq i} [1 - sim(i, j)]$$
 (2.17)

where sim(i, j) denotes item to item collaborative filtering similarity measures, and  $Z_u$  is the set of n recommendations to user u. In equation 2.17, diversity is calculated by summing over the similarity between pairs of recommended items and normalizing it.

$$novelty_i = \frac{1}{\#Z - 1} \sum_{j \in Z_u} [1 - sim(i, j)], \quad i \in Z_u$$
 (2.18)

Equation 2.18 shows how to obtain novelty for each recommended item. It returns the normalized similarity between item i and all other recommended items in  $Z_u$ . Note that, sometimes, novelty is vital, because there are some items which most of the users do not buy frequently (like refrigerator). Thus, if a user buy one of them, most likely he or she will not buy it again in the near future. Then, the RS should not continue to recommend it to the user. However, if the user tries to buy them again, the RS should learn that and include them in the set of recommended items. There are some other metrics that might or might not be important for an RS designer. For example, stability in the RS prediction. It implies that the set of recommendations should not be changed drastically through the time [3].

## Chapter 3

## Context-aware Recommender Systems

A wide range of recommender system techniques concentrate on the most relevant item based on user ratings. However, there is other useful information that can be collected in order to help the recommender system. This information may consist of time, place, job or any other beneficial information about the user or a group of users. As a result, in addition to the two traditional components of a recommender system, i.e. item and user, we have other information as well. This information is referred to "contextual information" and can be applied in special circumstances. For example, information about time can help us in recommending a travel package, or a web page. Additionally, mobile recommender systems attract attention, because a significant number of users have mobile devices and information such as location and time can be extracted from those devices in order to help the recommender system to understand the context better.

This topic leads us to a wider area of information that should be taken into account concerns user behavior in different circumstances, because a recommender system with more contextual information can be more accurate. For instance, a music recommender can be more accurate if it considers places of interest, in-car music, music while reading, and even the mood of the listener [19]. Another example is the Netflix recommender system that uses locational contextual variables such as city or zip code and time to provide context specific recommendations. Reed Hasting, the CEO of Netflix, claimed that they can improve the performance of their recommender system up to 3% when considering such contextual information<sup>1</sup>. In general, context-aware recommender systems consist of three main parts, pre-filtering, post filtering, and modeling, which we explain in more detail in the following sections.

#### 3.1 Context in Recommender Systems

Before we talk about context in recommender systems, we should know what the context is in general. The definition of context in Webster's dictionary is: "the parts of a discourse that surround a word or passage and can throw light on its meaning; condition or circumstances which effect something; the interrelated conditions in which something exists or occurs: environment, setting the historical context of the war". As can be seen, the definition is not precise, and that suggests that the concept of context is a multidisciplinary concept that can have a different definition in each field of study. In computer sciences, and specifically in recommender system studies, a context is the information that can improve the performance of the system and cannot be measured just by tracking user rating or item rates.

The traditional methods, particularly collaborative and content-based filtering, use two important fundamental elements of a recommender system, i.e. item and user, to predict the ratings. Therefore, we can assume that a recommender system is a function that takes users and items and returns ratings:

<sup>&</sup>lt;sup>1</sup>Watch the video at 44:40 minute https://www.youtube.com/watch?v=8FJ5DBLSFe4 <sup>2</sup>https://www.merriam-webster.com/dictionary/context?

$$R: User \times Item \rightarrow Rating$$

In this function, the input is actually in 2 dimensions since it only considers users and items. However, when we add the concept of context in our recommender system, it becomes another input parameter to the rating function. Then we have:

$$R: User \times Item \times Context \rightarrow Rating$$

Context can be considered as a vector that contains different contextual information. There have been two main representational approaches for context: hierarchical and tensor representation. Hierarchical representation is introduced by Palmisano et al. [39] suggesting granular information as contextual dimensions. In their model, contextual information is defined as a set of contextual dimension  $\mathbf{K}$ , so that each contextual dimension k is a set of q attributes  $k = \{k^1, ..., k^q\}$  and these attributes have a hierarchical structure to capture different types of context.  $k^q$  is the finer or more granular level of information, while the  $k^1$  defines the coarser or less granular level of contextual information.

As can be seen in figure 3.1, which is an example from [19], the root contains the coarsest level of information (all of the database). Then the next level is the information about whether the merchandise is for personal use or a gift; thus, we have  $k^1 = \{Personal, Gift\}$ . The next finer level of the hierarchy could be the values of either "Personal" or "Gift". Subsequently, we have the next finer level  $k^2 = \{PersonalWork, PersonalOther, GiftPartner/Friend, GiftParent/Other\}.$ 

A different way to represent context is the way that mathematicians work with tensors. If we let  $D_1, D_2, \ldots, D_n$  be dimensions the input vector for the R function:

$$R: D_1 \times D_2 \times \ldots \times D_n \to Rating$$

Two of these dimensions are item and user, and the rest are contexts. In the tensor representation method, each dimension  $D_i$  is a Cartesian product of

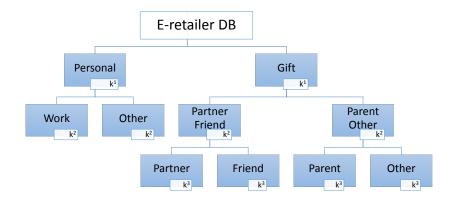


Figure 3.1: contextual information represented by hierarchical structure.

some attributes  $A_{ij}$ ,  $(j = 1, 2, ..., k_i)$ ; that is,  $D_i \subseteq A_i 1 \times A_i 2 \times ... A_{ik_i}$ . For more illustrations, we can consider a recommender system in which a user has the information such as UName and Address and Age. This can then be shown as  $User \subseteq UName \times Address \times Age$ ; likewise, the item dimension could be  $Item \subseteq IName \times Type \times Price$ , and if we consider the Time as our context, it could be  $Time \subseteq Year \times Month \times Day$ . Figure 3.2 shows this tensor model for the stated example.

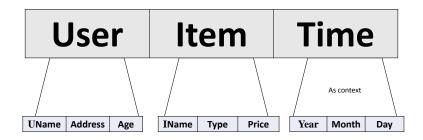


Figure 3.2: Precision and recall with respect to all of the items.

#### 3.2 Obtaining contextual information

Before working on a recommender system that uses context, we need to know how the context is going to be obtained, albeit there are some context-aware recommender systems (CARS) that assume the contextual dimensions have already been provided. The availability of information about the items, users and some other circumstances related to them or the interaction between them plays a significant role in content acquisition. In general, there are three ways to obtain context [19]:

- Explicitly As we mentioned in Chapter 2, in the explicit approach, the information is gained directly from entities. For example, a website or a company may provide a survey for users and ask them to fill it in. Likewise, the information of location or time can be extracted from the users' device.
- Implicitly This type of information needs a monitoring system to observe the users and interactions. It should be noted that the source of information is accessed directly. For example, frequent changes in the GPS of a user extracted from her device is implicit information about the user that suggests the user may not stay for a long time in a specific location.
- Inferring In this approach, the recommender system should infer information from other data that has already been extracted. The information here is hidden and requires special algorithms to be revealed. For example, if a machine learning method recognizes the type of person who is watching TV at home, it can help the recommender system to recommend better TV shows which are more desirable for that specific person.

All of the aforementioned methods should be performed as part of the data collection process, because the recommender system relies on the data and predicts rates based on them. Another important issue here is the relevance of the extracted context. For example, a book store (either online or traditional) can capture information from a buyer regarding their purpose in buying the book, the planned reading time, and general information about the stock market at the time of buying. However, the information about stock market may not be applicable at all. Hence, the relevance of the information is important, and it becomes crucial in context-aware recommender systems because they work with larger databases than usual. Here it is necessary to have an expert in the domain of the application. Another example is a mobile recommender system that needs physical context such as time and position, and social context regarding whether the user is alone or not; interaction media context such as the type of device is also important context. In the case of technology enhanced learning (TEL), computing context, user context, and physical context are all important [56].

Besides using an expert or a manual approach to define the essential relevance context, there are some machine learning and data mining algorithms that help us to detect contexts automatically [37]. Adomavicius et. al. in [2] suggested that an expert should suggest some contextual features as candidate; then, by statistical methods, the most relevant one is extracted. For example, they did a pairwise t-test among candidate features. Another common way to assess the relevance of a context is stated by Baltrunas et al. [6] which it is suggested that some hypothetical contextual preferences should be offered to users as a survey. Then they ask users to respond to survey, and in this way they collect useful contextual information. They show that their system outperforms a recommender system that does not use context.

## 3.3 Utilizing Context in Recommender Systems

In order to utilize context in recommender systems, we should take into account two major approaches to using this information: (i) context-driven querying and (ii) contextual preference elicitation and estimation. The context-driven approach suggests that the recommender system should rely only on contextual information and try to relate the items and users based on the contextual information. Some scholars use it to create a mobile tourist recommender system [12].

On the other hand, the contextual preference elicitation and estimation methods have engaged more context-aware recommender system researchers. Unlike the previous method, this one encourages us to learn the context and reinforce the collaborative or content-based filtering by using it. It should be noted that it is possible to design a recommender system that uses a combination of both general methods. We may recall that recommender systems are created based on partial user preferences (i.e. some ratings from some users), and the input record of recommended systems are a subset of < user, item, rating >. In context-aware recommender systems we have a new element known as "context" that changes the records to a new tuple which is < user, item, context, rating >. Now it is important to decide how and where in the recommender system we should use contextual features. Generally, we can use context either before selecting data records, after selecting them, or in the recommending process. Figure 3.3 illustrates these methods, and we will explain them in more detail in the following sections.

#### 3.3.1 Contextual Pre-filtering

In this recommendation paradigm (figure 3.3a), the information about a certain context c is used to select or filter relevant data; then it is fed to the conventional 2D (i.e.  $User \times Item$ ) methods such as collaborative or content-

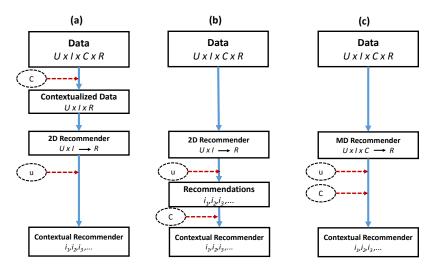


Figure 3.3: Paradigms of using context in recommender systems. (a) Prefiltering (b) post-filtering (c) contextual modeling.

based filtering. For instance, context c is considered as a query to find relevant ratings data [19]. For a more detailed illustration, if we assume that a viewer wants to watch a movie on Saturday, the recommender system first picks all the Saturday movies' ratings and feeds them to a collaborative filter to find the closest user, and then recommends the best items for the viewer. This method is called exact pre-filtering, because the data filtering is based on an exact value of a context. It can be seen that we turned a 3D input recommendation problem into a 2D one; after all, the collaborative filtering part of the recommender system does not deal with context anymore. It can be easily implemented by a selection and a projection over the database as following:

$$\forall (u,i,c) \in U \times I \times T, R_{user \times item \times context}^{D}(u,i,t) = R_{user \times item}^{D[Context = c](User,Item,Rating)}(u,i)$$

The downside of this method is the narrow context that it returns. For example, in a case where the context is c = (Partner, Theater, Saturday), the recommender may not find the best movie that is playing in a good theater on Saturday that is good for the viewer to watch with a partner. In

order to avoid this problem, Adomavicius et al. [2] suggest using generalized pre-filtering which uses aggregated information and tries to generalize the contextual information. If we recall from our previous example, the recommender system could aggregate Saturday and Sunday together and show them by a new aggregated value named "Weekend", in which case it would find more options. We let  $S_c$  be a segment of data that aggregates context, i.e.  $c \in S_c$ ; then, in this method, the selection and projection steps would be:

$$R_{user \times item \times context}^{D}(u,i,t) = R_{user \times item}^{D[Context \in S_c](User,Item,Aggregate(Rating))}(u,i)$$

Moreover, it is even possible to use more than one aggregated context and to filter the data based on them.

The aggregation reformation brings another problem, which is the need to find a "right" level of granularity. One may think about using an expert person for that, but to have an adaptive system for big data we need a computational automatic approach. In [30], the authors investigated different levels of generalization and compared the prediction accuracy of the recommender system in order to find the best level of generalization.

Another issue is the locality problem that happens because the recommendations come from the data that is pre-filtered by aggregated context from a specific segment of contextual information. For example, if you want to go out and have fun by watching a movie in a theater, then a pre-filtering context-aware recommender system may generate better recommendations for you. However, if you are at home and want to watch a movie on your TV, then it might be better to have a simple 2D recommender system.

#### 3.3.2 Contextual Post-filtering

In this method (figure 3.3b), contextual information is not considered until the last step of the recommendation. It means that the system takes the whole 3D database and makes decisions on this data; then, at the end, right

before making the final list of recommended items, the contextual information is applied to adjust the final list. There are two main approaches to modify the final list based on the contextual information: *Filter out* the irrelevant items or *reorder* the items in the recommended list. Furthermore, the post-filtering technique is classified into heuristic (memory-based) and model-based ones.

In the heuristic method, the post-filter part of the recommender system searches for common item features for a given context and adjusts the list based on their quantity. For example, if you like some movies with specific actors, it will adjust the recommendation list to include more of those actors. This adjustment can be done by filtering (dumping) out the movies that do not have a specific number of those actors, or it can be accomplished via ranking the movies in the list based on the number of desired actors involved in them.

In the model-based method, the post-filter can learn the probability of the popularity of a movie based on the its context. For instance, it may learn the likelihood of choosing a movie with a certain director. Then it uses that probability to adjust the recommendation list. This adjustment operation could filter out the items which have the relevance probability less than a certain threshold. Similar to the heuristic model, it can also rank the final list by weighting the items in it using the calculated probability. Panniello et al. [41] compare post-filtering and pre-filtering methods on two databases of an e-commerce and Amazon<sup>3</sup>. Their results suggest that weighted post-filtering performs better than the pre-filtering method, and the pre-filtering outperforms the filter post-filtering.

<sup>&</sup>lt;sup>3</sup>Their dataset consists of some items purchased by students containing contextual information

#### 3.3.3 Contextual Modeling

In contextual modeling (figure 3.3c), the contextual information is used in the process of finding the unknown ratings. One common method is to deploy the context directly in the process of user rating prediction. In contradiction to the pre-filtering and post-filtering methods, this method uses the 3D recommendation function. That means that it operates like Rating = R(User, Item, Context) where R is a prediction function that predicts each user's rating on a target item. A similarity function can be used to find the similarity between the < user, item, context > tuples. The unknown ratings are predicted with respect to those tuples that have rates on items. Moreover, the ratings involved in this calculation are inversely related to the similarity metrics. Equation 3.1 shows the a prediction method for an unknown rate  $r_{u,i,c}$  for < u,i,c > which is a tuple in the database:

$$r_{u,i,c} = k \sum_{(u',i',c')\neq(u,i,c)} W((u',i',c'),(u,i,c)) \times r_{u',i',c'}$$
(3.1)

where k in a normalization factor, and W((u',i',c'),(u,i,c)) is the "weight" of the rating  $r_{u',i',c'}$  participating in calculating the prediction rate which can be the inverse of the Euclidean distance between (u',i',c') and (u,i,c). In other research [1] the aggregated information of the context shows better performance; furthermore, the authors consider the distance equal to zero wherever the the context in two tuples is not the same (i.e. if  $c \neq c'$  then dist((u',i',c'),(u,i,c)) = 0).

Additionally, Oku et al. [38] use the additional context in the 3D database and use the support vector machine (SVM) classification, which looks into the items and corresponding ratings as two sets of "like" and "dislike" and creates the hyperplane based on the support vectors, then recommending the items that fall on the like side of the hyperplane.

## Chapter 4

## Social-based Recommender Systems

The emergence of social networks and their drastic growth suggests that the tremendous information within them could be helpful in many applications including recommender systems. Moreover, the overload of resources (i.e. items and data in general) makes the process of making decisions even harder for social media users. Therefore, we need a social media-based system that channels the resources in social media. For example, by learning from the new types of data extracted from online social networks such as tags and relationships we can help a recommender system to find similar users in a better way.

In general, social information is useful for three main reasons [8]. First it can be deployed to improve the quality of prediction. For example, the RS may infer that since two users are friends in a social network, it is possible for them to have the same taste for items. This can help collaborative filtering methods. In [58] the authors show that social information enhances the result of collaborative filtering. Second, it can even be used to create a new recommender system. Here, the goal is not to improve a pre-designed recommender system but to propose a new way to generate an RS based on

social information. Siersdorfer and Sergei in [51] used the multi-dimensional social environment of a specific user to create a social recommender that suggests users, items or groups to that specific user. The third purpose of social filtering is just to analyze the relationships between social information and collaborative entities. For example, correlation between recommender and recommendee may be important for decision-making problems.

Initially it was thought that social information could be used to create a trust network for recommender systems, but weak generalization led the scholars to have a wider overview on information from social networks [53]. Moreover, some researchers believe that content recommendation is an important subject that should be considered in social-based recommender systems [24].

#### 4.1 Recommendation related to contents

In social media, content plays a vital role, whether it is going to be recommended to users to use, or to generate new content. For example, a social recommender for Facebook users may recommend news or video to the users to read or watch. On the other hand, it may recommend topics based on the trends or tags to users to post a text about it. The content could be the comments of a user, the tags used, or the votes or ratings (e.g. like and dislike). These contents in addition to the relations among users, can give us an invaluable opportunity to have a more effective recommender system. Golbeck [21] uses membership forms from the "FilmTrust" system, which is a web-based social network and has a movie rating and review system. The author uses trust between individuals as the weight of their mutual rating on an item, then estimates the unknown rating based on the weighted known ratings. Her results show that this information can improve movie recommendations.

Guy [24] mentions the important "key domains" in social recommender

systems. The first important one is the *blog* which is one of the classic social media. A blog owner (a person or company) writes about a topic on the blog and it creates a blog post. The owner or users of the blog can add posts about the topic or interpret it in the comments. The blog itself can be an item to recommend. Moreover, the content of the posts and the reactions to them can be considered as a context or extra information to help the recommender system.

The next key content is *multimedia*, which is quite challenging since extracting the actual content in an audio file or a video is computationally expensive, and it returns an enormous amount of content. The most famous social media for multimedia purposes is YouTube. Davidson et al. [14] use covisited video counting and associated rule planning [61] to predict the score of a video. They suggest that the YouTube recommender system should recommend fresh and diverse videos with respect to the video that the user has recently watched or reacted to. Beyond that, they state that the user should understand why a video was recommended to them.

Question and answer is the next important content in special Q&A websites like StackOverFlow and Yahoo Answers. The main issue here is to recommend other relevant questions and/or appropriate answers. Another content related to online social network is news. Social news broadcasters such as Digg, Reddit, or Google Readerlet try their best to recommend the most relevant and popular news to the readers. Research from Google [33] creates a distribution of user clicks over a year tallied for each month. Then they use this information for computing the distance and then similarity to feed to the collaborative filtering part of their recommender system. They improved the pure collaborative filtering method by 7% via this technique. It should be noted that recommending the freshest and most recent post is extremely crucial in both question and answer and news social networks.

The Other content is about 'jobs. LinkedIn is the best known website one in this area. Additionally, ResearchGate is recommending jobs and op-

portunity. The significant impact of this subject on people's lives make it an attractive one for recommender system scholars. Figure 4.1 shows a profile on ResearchGate that recommends some job opportunities based on the user profile; the user can interactively purify the recommendations by leaving feedback on the recommended option.

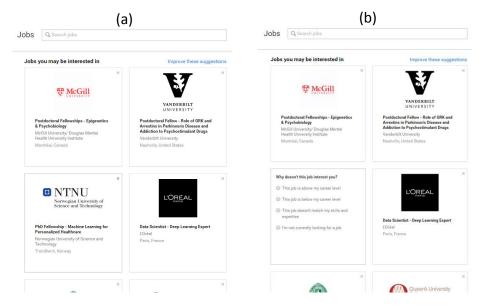


Figure 4.1: Example of job recommendation by ResearchGate (a) Recommend opportunities (b) User can delete an opportunity from the recommended list and leave feedback.

Microblogs are among the other contents that have become popular with Twitter. Here, the concept of "follower" and "follower" can help us to recommend tweets or people to users to follow. Most of algorithms use one of the following aspects on Twitter to recommend content: candidate selection, topic relevance or social voting. Social voting is about the number of user's followes that follow the user and also follows the posts that the users tweeted or followed. Comprehensive research [13] on different algorithms of recommending URL on Twitter shows that social voting works better than topic relevance.

#### Social information in movie domain recommendation

Carrer-Neto et al. [11] use semantic knowledge extracted from movie meta data along with the data extracted from the profile of the user. They assume that in the database they have, the user defined her "social aperture" by choosing one of these options: "Moderate", "Liberal" or "Conservative".

If the user is moderate, they use 25% of her friends' ratings to calculate her rating. If the user is liberal, both her own ratings and her friends' ratings will be considered equally and if she is conservative, then they only use the ratings calculated for the user. Their results show that using this social information outperforms the case where they did not use it.

### 4.2 People Recommendation

One of the main duty of a social recommender system is recommending people to each other. Social media websites must use an algorithm to suggest relevant or similar people to each other. Guy [24] discusses that relation between people on social networks has different dimensions. For instance, in Facebook, you and another user may become friends to each other, but in Twitter you may follow a user without having followed back by that user. Here we have the issue of "symmetric" versus "asymmetric" relations. Moreover, in this example, you need to send an invitation to become somebody's friend, but on Twitter you can follow a user without her "confirmation". Thus, you may face different social network either with or without confirmation. Sometimes a relation is a temporary one just to organize a meeting or an event. But in other cases, your relation is permanent. For example you may be in an online group of researchers from you laboratory. This indirect relation is considered permanent.

An example of recommending people has been studied by Geil et al. [20]. They use the "Who To Follow" (WTF) algorithm on GPU. The algorithm first finds the circle of trust (CoT) of the user, which is the 1000 nodes closest

to the user, and then creates a bipartite graph of individuals from the CoT on one side and the ones they follow on the other side. Then it uses Twitter's Money algorithm and assigns a similarity and relevance number to all nodes, after which it propagates the similarity value to followers and the relevance to followers. Finally, it recommends people with the highest relevance scores.

## 4.3 Group Recommendation

Another issue in a social recommender system is "group recommendation". It is important to determine that whether the recommender system is going to recommend items to a group or only to an individual. For example, in a case where the recommender system is going to recommend a TV show, if a group like a family wants to watch it, the system may recommend different items in comparison to the situation in which it deals with just one person. Consequently, some other questions that matter are what kinds of groups we are going to make our recommendation to, and how similar the members of the groups are to each other. One application here could be recommending some music to a group of people who are working out in a gym. Profile aggregation and recommendation aggregation are the most common approaches in this field.

In order to aggregate the rating of a group, we need to consider the type of strategies used to obtain the group rating. For this purpose, group recommending take into account three main strategies [17]:

• Average satisfaction which assumes equal importance for all the members of the group. Let  $GR_i$  be the group rating on the item i, then in order to calculate it, we use equation 4.1 that simply calculates the average.

$$GR_i = average(r_{u,i}) = \frac{\sum_{u=1}^{n} r_{u,i}}{n}$$
(4.1)

where  $r_{u,i}$  is the rating of user u on item i and n is the number of

members in the group.

- Minimum misery is used when we want to give special attention to the members of the group that rate an item very low. In this case, the group average is the minimum rating of all members, i.e.  $GR_i = min\{r_{u,i}\}$ .
- Maximum satisfaction which is concerned with the members that rate an item higher than other members of the group. Then the group rating is the maximum of the rating of the members, i.e.  $GR_i = max\{r_{u,i}\}$ .

Nevertheless, the aforementioned strategies are not accurate enough to describe the aggregated group rating on an item. Hence, Gartell et al. [17] try to use social information in order to have a better group descriptor. They define a social weight  $w_{u,v}$  as the contact frequency over a specific time. We can generalize it to a proportion of the number of tags two connected users have in common. Equation 4.2 obtains this descriptor:

$$S(G) = \frac{2 \cdot \sum_{u,v \in G} w_{u,v}}{|G| \cdot (|G| - 1)}$$
(4.2)

Then they use it to define how much they should rely on min, max or average rating of the group. Basically, they say that if the social descriptor is not high or low, the average is desired, but if it is high or low, the maximum or minimum should be considered accordingly.

## 4.4 Immediate Friend Inference

If the access to social data about users is provided, we can involve the friends of a user to recommend the best suitable item to her. The impact of immediate friends (i.e. the friends with one hop distance) and a probability-based inference is discussed in [27] by He and Chu. They assume that the ratings are integers; then, they try to find out what is the probability of rating of user u on the item i, i.e.  $R_{u,i}$ , given the set of attributes  $a_u$  of user, set of

attributes  $b_i$  of item and the rating of the neighbors, i.e.  $R_{v,i}$ , for that item. They use naive Bayesian assumption and reach the equation 4.3.

$$P(R_{u,i} = k | B = b_i, A = a_u, \{R_{v,i} = r_{v,i} : \forall v \in U_i \cap N_u\})$$

$$= \frac{1}{Z} P(R_{u,i} = k | B = b_i) \times P(R_{u,i} = k | A = a_u)$$

$$\times P(R_{u,i} = k | \{R_{v,i} = r_{v,i} : \forall v \in U_i \cap N_u\})$$
(4.3)

where B is the random variable standing for the set attribute of item i,  $b_i$  is the set of values of attribute of item i,  $A_u$  is the random variable standing for the set attribute of user u,  $a_u$  is the set of values of attributes of user u, v is a neighbor of user v, v is the set of users that rated item v, v is the set of the neighbors (friends) of user v, and v, v is the ratings of neighbors of user v on item v.

Now we need to calculate each probability independently. The probability of the rating of user u given set of attributes for item i is the user preference. This means that in order to calculate the user preference we should find the probability  $P(R_{u,i} = k|B = b_i)$ ; with the naive Bayesian assumption we have equation 4.4.

$$P(R_{u} = k | B = b_{i}) = \frac{P(R_{u} = k) \times P(B_{1}, B_{2}, \dots, B_{n} | R_{u} = k)}{P(B_{1}, B_{2}, \dots, B_{n})}$$

$$= \frac{P(R_{u} = k) \times \prod_{j=1}^{j=n} P(B_{j} | R_{u} = k)}{P(B_{1}, B_{2}, \dots, B_{n})}, B_{j} \in \{B_{1}, B_{2}, \dots, B_{n}\}$$

$$(4.4)$$

where  $P(R_u = k)$  is the prior probability that the user u gives a rating k, and  $P(B_j|R_u = k)$  is the conditional probability that each item with attribute  $B_j$  in B gets the value  $b_j$  given u rated it with k. For example,  $P(actor = AlPacino|R_u = 5) = 0.9$  means that the probability that Al Pacino plays in a movie given the movie received the rate 5 equals to 0.9. Equation 4.5 and equation 4.6 calculate the two nominator probabilities in the previous equation by a simple counting over the database.

$$P(R_u = k) = \frac{|I(R_u = k)| + 1}{|I(u)| + n}$$
(4.5)

$$P(B_j = b_j | R_u = k) = \frac{|I(B_j = b_j, R_u = k)| + 1}{|I(R_u = k)| + m}$$
(4.6)

where |I(u)| is the number of items that the user u rated,  $|I(R_u = k)|$  is the number of items that the user u gives the rating equal to k, and  $|I(B_j = b_j, R_u = k)|$  is the number of ratings k that the user u gave to items with the attribute of  $b_j$ . We add one in the numerator and n as the range of ratings and m as the range of attribute value in the denominators, because of the Laplace estimate that helps us in avoiding strong probabilities.

Subsequently, we need to find the item acceptance probability which is  $P(R_i = k | A = a_u)$ . It implies the general acceptance of item i from users like user u. For example, if two reviewers are similar to each other and one of them rated "The Godfather" 5, we want to know how likely is that the other one gives the same rating. Again, by naive Bayesian assumption, we have equation 4.7.

$$P(R_{i} = k | A = a_{i}) = \frac{P(R_{i} = k) \times P(A_{1}, A_{2}, \dots, A_{n} | R_{i} = k)}{P(A_{1}, A_{2}, \dots, A_{n})}$$

$$= \frac{P(R_{i} = k) \times \prod_{j=1}^{j=m} P(A_{j} | R_{i} = k)}{P(A_{1}, A_{2}, \dots, A_{n})}, A_{j} \in \{A_{1}, A_{2}, \dots, A_{m}\}$$

$$(4.7)$$

where  $P(R_i = k)$  is the prior probability that item i receives a rating value k, and  $P(A_j|R_i = k)$  is the conditional probability that a user has attribute  $A_j$  equal to  $a_j$  given that she rates item i as k. Note that in the previous equations, both  $P(B_1, B_2, \ldots, B_n)$  and  $P(A_1, A_2, \ldots, A_m)$  are normalizing constants.

Finally, the influence from immediate friends should be obtained, i.e.  $P(R_{u,i} = k | \{R_{v,i} = r_{v,i} : \forall v \in U_i \cap N_u\})$ . Some methods use the correlation between the user and its neighbors based on user attributes, but this correlation is hard to capture with a simple similarity or correlation function. Then the authors in [27] suggest that we can use the histogram of the differences between the immediate friends rating and the user rating. Therefore, for

each user u and her neighbor v, we have equation 4.8:

$$P(R_{u,i} = k | R_{v,i} = r_{v,i}) \propto H(k - r_{v,i})$$
 (4.8)

In order to calculate it for all the neighbors of u, these differences are multiplied and divided by a normalization factor of the histogram of each immediate friend pair.

#### 4.5 Link Prediction for Social Networks

Online social media is growing with a significantly important pace. An applicative domain of social-based recommender system is link prediction on social media [4]. Want et al. [57] provide a thorough review over this topic. They divide the link recommendation on social networks to two major categories: similarity-based approach and learning-based approach. They also explore the social theory-based metrics, the node-based metrics and the topology-based metrics. The latter ones mainly considers the neighbors and the path for qualification metrics.

A number of papers has focused on particular social networks. For example, Yao et al. [59] explore the friend suggestion in online photo-sharing communities such as Facebook and flicker. In another article, Liben-Nowell et al. [15] explores the problem in the context of freind suggestion over Twitter. As a future work, emojis can be considered as a tuner for link prediction techniques, because the emoji usage analysis shows regularities [16, 49] and semantics [48] on Twitter users. Thus, the user with similar feelings and common friend may be subjected for link suggestions.

If we consider the networks of researchers as a social network (e.g. Mendeley, ResearchGate, etc.), then research paper recommendation may be treated as a social-based recommendation system. Because one of the major applications of the recommender systems is to recommend a set of relevant and useful papers to a scholar in the right time. In addition to the time limita-

tions, the issue of copyright prevent a recommender system to access to the full content of a paper. Two popular approaches are context-based collaborative filtering [32] and co-citation [36]. In the former, the authors use the network of citations to create the rating matrix. The latter takes into account the assumption that if two papers cite the same papers, they are similar.

Haruna et al. [26] propose a collaborative method that uses the public data about the paper for the recommendation purposes. In their method, if author A writes a paper P, they consider recommending papers that have two conditions: They are co-cited with the paper P of the author A, and have common references with paper P. They show their method outperforms context-based collaborative filtering and co-citation techniques. Another method for research paper recommendation is to analyze the topics of the papers [40].

# ACKNOWLEDGMENTS

The authors would like to thank the NSF for funding the AMALTHEA REU and Florida Institute of Technology for hosting the program. The authors would also like to acknowledge support from the NSF grant No. 1560345. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

# **Bibliography**

- [1] G. Adomavicius and A. Tuzhilin. Incorporating context into recommender systems using multidimensional rating estimation methods. In Proceedings of the 1st International Workshop on Web Personalisation, Recommender Systems and Intelligent User Interfaces Volume 1: WPRSIUI, (ICETE 2005), pages 3–13, 2005.
- [2] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE transactions on knowledge and data engineering, 17(6):734–749, 2005.
- [3] G. Adomavicius and J. Zhang. Stability of recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 30(4):23, 2012.
- [4] M. Al Hasan and M. J. Zaki. A survey of link prediction in social networks. In *Social network data analytics*, pages 243–275. Springer, 2011.
- [5] M. Balabanović and Y. Shoham. Fab: content-based, collaborative recommendation. Communications of the ACM, 40(3):66-72, 1997.
- [6] L. Baltrunas, B. Ludwig, S. Peer, and F. Ricci. Context relevance assessment and exploitation in mobile recommender systems. *Personal* and *Ubiquitous Computing*, 16(5):507–526, 2012.

- [7] J. Bobadilla, A. Hernando, F. Ortega, and J. Bernal. A framework for collaborative filtering recommender systems. *Expert Systems with Applications*, 38(12):14609–14623, 2011.
- [8] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez. Recommender systems survey. *Knowledge-based systems*, 46:109–132, 2013.
- [9] R. Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.
- [10] L. Candillier, F. Meyer, and M. Boullé. Comparing state-of-the-art collaborative filtering systems. In *International Workshop on Ma*chine Learning and Data Mining in Pattern Recognition, pages 548–562. Springer, 2007.
- [11] W. Carrer-Neto, M. L. Hernández-Alcaraz, R. Valencia-García, and F. García-Sánchez. Social knowledge-based recommender system. application to the movies domain. *Expert Systems with applications*, 39(12):10990–11000, 2012.
- [12] F. Cena, L. Console, C. Gena, A. Goy, G. Levi, S. Modeo, and I. Torre. Integrating heterogeneous adaptation techniques to build a flexible and usable mobile tourist guide. *AI Communications*, 19(4):369–384, 2006.
- [13] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. Chi. Short and tweet: Experiments on recommending content from information streams. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '10, pages 1185–1194, New York, NY, USA, 2010. ACM.
- [14] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston, et al. The youtube video recommendation system. In *Proceedings of the fourth ACM conference* on *Recommender systems*, pages 293–296. ACM, 2010.

- [15] A. Epasto, S. Lattanzi, V. Mirrokni, I. O. Sebe, A. Taei, and S. Verma. Ego-net community mining applied to friend suggestion. *Proceedings of the VLDB Endowment*, 9(4):324–335, 2015.
- [16] H. Fede, I. Herrera, S. M. Seyednezhad, and R. Menezes. Representing emoji usage using directed networks: A twitter case study. In *Inter*national Workshop on Complex Networks and their Applications, pages 829–842. Springer, 2017.
- [17] M. Gartrell, X. Xing, Q. Lv, A. Beach, R. Han, S. Mishra, and K. Seada. Enhancing group recommendation by incorporating social relationship interactions. In *Proceedings of the 16th ACM international conference* on Supporting group work, pages 97–106. ACM, 2010.
- [18] M. Ge, C. Delgado-Battenfeld, and D. Jannach. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings* of the fourth ACM conference on Recommender systems, pages 257–260. ACM, 2010.
- [19] A. T. Gediminas Adomavicius. Context-aware recommender systems. In *Recommender systems handbook*, pages 191–226. Springer, 2015.
- [20] A. Geil, Y. Wang, and J. D. Owens. Wtf, gpu! computing twitter's who-to-follow on the gpu. In Proceedings of the second ACM conference on Online social networks, pages 63–68. ACM, 2014.
- [21] J. Golbeck. Generating Predictive Movie Recommendations from Trust in Social Networks, pages 93–104. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
- [22] S. Gong. A collaborative filtering recommendation algorithm based on user clustering and item clustering. *JSW*, 5(7):745–752, 2010.

- [23] C. Guan, S. Qin, W. Ling, and G. Ding. Apparel recommendation system evolution: an empirical review. *International Journal of Clothing Science and Technology*, 28(6):854–879, nov 2016.
- [24] I. Guy. Social recommender systems. In Recommender systems hand-book, pages 511–543. Springer, 2015.
- [25] U. Hanani, B. Shapira, and P. Shoval. Information filtering: Overview of issues, research and systems. *User modeling and user-adapted interaction*, 11(3):203–259, 2001.
- [26] K. Haruna, M. A. Ismail, D. Damiasih, J. Sutopo, and T. Herawan. A collaborative approach for research paper recommender system. *PloS* one, 12(10):e0184516, 2017.
- [27] J. He and W. W. Chu. A social network-based recommender system (snrs). In *Data Mining for Social Network Data*, pages 47–74. Springer US, 2010.
- [28] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.
- [29] Z. Huang, D. Zeng, and H. Chen. A comparison of collaborative-filtering recommendation algorithms for e-commerce. *IEEE Intelligent Systems*, 22(5), 2007.
- [30] T. Jiang and A. Tuzhilin. Improving personalization solutions through optimal segmentation of customer bases. *IEEE transactions on knowledge and data engineering*, 21(3):305–320, 2009.
- [31] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

- [32] H. Liu, X. Kong, X. Bai, W. Wang, T. M. Bekele, and F. Xia. Context-based collaborative filtering for citation recommendation. *IEEE Access*, 3:1695–1703, 2015.
- [33] J. Liu, P. Dolan, and E. R. Pedersen. Personalized news recommendation based on click behavior. In *Proceedings of the 15th international conference on Intelligent user interfaces*, pages 31–40. ACM, 2010.
- [34] X. Luo, Y. Xia, and Q. Zhu. Incremental collaborative filtering recommender based on regularized matrix factorization. *Knowledge-Based Systems*, 27:271–280, 2012.
- [35] C. M. F. N. G. S. Marco de Gemmis, Pasquale Lops. Semantics-aware content-based recommender systems. In *Recommender systems hand-book*, pages 119–159. Springer, 2015.
- [36] S. M. McNee, I. Albert, D. Cosley, P. Gopalkrishnan, S. K. Lam, A. M. Rashid, J. A. Konstan, and J. Riedl. On the recommending of citations for research papers. In *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 116–125. ACM, 2002.
- [37] A. Odić, M. Tkalčič, J. F. Tasič, and A. Košir. Predicting and detecting the relevant contextual information in a movie-recommender system. *Interacting with Computers*, 25(1):74, 2013.
- [38] K. Oku, S. Nakajima, J. Miyazaki, and S. Uemura. Context-aware sym for context-dependent information recommendation. In *Mobile Data Management*, 2006. MDM 2006. 7th International Conference on, pages 109–109. IEEE, 2006.
- [39] C. Palmisano, A. Tuzhilin, and M. Gorgoglione. Using context to improve predictive modeling of customers in personalization applications. *IEEE transactions on knowledge and data engineering*, 20(11):1535–1549, 2008.

- [40] C. Pan and W. Li. Research paper recommendation with topic analysis. In Computer Design and Applications (ICCDA), 2010 International Conference on, volume 4, pages V4–264. IEEE, 2010.
- [41] U. Panniello, A. Tuzhilin, M. Gorgoglione, C. Palmisano, and A. Pedone. Experimental comparison of pre-vs. post-filtering approaches in context-aware recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, pages 265–268. ACM, 2009.
- [42] M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial intelligence review*, 13(5-6):393–408, 1999.
- [43] F. Ricci, L. Rokach, and B. Shapira. *Recommender Systems Handbook*, chapter 1. Springer Nature, 2015.
- [44] T. H. Roh, K. J. Oh, and I. Han. The collaborative filtering recommendation based on som cluster-indexing cbr. *Expert systems with applications*, 25(3):413–423, 2003.
- [45] J. Salter and N. Antonopoulos. Cinemascreen recommender agent: combining collaborative and content-based filtering. *IEEE Intelligent Systems*, 21(1):35–41, 2006.
- [46] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Readings in information retrieval*, 24(5):355–363, 1997.
- [47] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th* annual international ACM SIGIR conference on Research and development in information retrieval, pages 253–260. ACM, 2002.
- [48] S. M. Seyednezhad, H. Fede, I. Herrera, and R. Menezes. Emoji-word network analysis: Sentiments and semantics. In *The 31th International FLAIRS Conference*, page In press. AAAI, 2018.

- [49] S. M. Seyednezhad and R. Menezes. Understanding subject-based emoji usage using network science. In *Workshop on Complex Networks CompleNet*, pages 151–159. Springer, 2017.
- [50] R. Sharma and R. Singh. Evolution of recommender systems from ancient times to modern era: A survey. *Indian Journal of Science and Technology*, 9(20), 2016.
- [51] S. Siersdorfer and S. Sizov. Social recommender systems for web 2.0 folksonomies. In *Proceedings of the 20th ACM conference on Hypertext and hypermedia*. ACM Press, 2009.
- [52] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. Advances in Artificial Intelligence, 2009:1–19, 2009.
- [53] Z. Sun, L. Han, W. Huang, X. Wang, X. Zeng, M. Wang, and H. Yan. Recommender systems based on social networks. *Journal of Systems and Software*, 99:109–119, 2015.
- [54] S. Teltscher. Ict data and statistics. Technical report, International Telecommunication Union, 2015.
- [55] R. Van Meteren and M. Van Someren. Using content-based filtering for recommendation. In *Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop*, pages 47–56, 2000.
- [56] K. Verbert, N. Manouselis, X. Ochoa, M. Wolpers, H. Drachsler, I. Bosnic, and E. Duval. Context-aware recommender systems for learning: a survey and future challenges. *IEEE Transactions on Learning Technologies*, 5(4):318–335, 2012.
- [57] P. Wang, B. Xu, Y. Wu, and X. Zhou. Link prediction in social networks: the state-of-the-art. *CoRR*, abs/1411.5118, 2014.

- [58] W. Woerndl and G. Groh. Utilizing physical and social context to improve recommender systems. In 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology Workshops. IEEE, nov 2007.
- [59] T. Yao, C.-W. Ngo, and T. Mei. Context-based friend suggestion in online photo-sharing community. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 945–948. ACM, 2011.
- [60] R. B. Yehuda Koren. Advances in collaborative filtering. In *Recommender systems handbook*, pages 77–118. Springer, 2015.
- [61] C. Zhang and S. Zhang. Association rule mining: models and algorithms. Springer-Verlag, 2002.