How to Improve Multi-Agent Recommendations Using Data from Social Networks?

Fabiana Lorenzi
Universidade Luterana do Brasil
Canoas, RS, Brasil
fabilorenzi@gmail.com

Bruno Fontanella
Universidade Luterana do Brasil
Canoas, RS, Brasil
brantznakbca@gmail.com

Edson Prestes
Instituto de Informatica, UFRGS
Porto Alegre, RS, Brasil
prestes@inf.ufrgs.br

Andre Peres
Instituto Federal do Rio Grande do Sul
Porto Alegre, RS, Brasil
andre.peres@poa.ifrs.edu.br

Abstract

User profiles have an important role in multi-agent recommender systems. The information stored in them improves the system's generated recommendations. Multi-agent recommender systems learn from previous recommendations to update users' profiles and improve next recommendations according to the user feedback. However, when the user does not evaluate the recommendations the system may deliver poor recommendations in the future. This paper presents a mechanism that explores user information from social networks to update the user profile and to generate implicit evaluations on behalf of the user. The mechanism was validated with travel packages recommendations and some preliminary results illustrate how user information gathered from social networks may help to improve recommendations in multi-agent recommender systems.

Introduction

Multi-agent Recommender Systems help users search products according to their preferences. In these types of systems, the existing agents collaborate and argue their different proposals to reach a consensus about the recommendation to be presented. The decentralized architecture results in a reduction of the time that the recommendation is presented to the user.

Recommendations may be improved through the trust mechanism that may be applied to the system. Multi-agent recommender systems need the user feedback to evaluate how was the performance of each agent involved in the generation of the recommendation. According to the user feedback the system may update the trust of each agent in order to ensure the quality of next recommendations.

However, obtaining user feedback is not an easy task, mainly because users rarely return to the system and evaluate the received recommendation. This user behavior inhibits the ability of the system to update its trust.

Considering the challenge of obtaining feedback information directly from the user, this paper presents a mechanism that evaluates the recommendations on behalf of the user. The proposed tool is capable of gathering information about the user from the social networks, and using these information in two ways: 1) to update the user profile, and 2) to create the user evaluation (feedback) automatically, without the user interaction. These evaluations are used by the multi-agent recommender system to update the agents trust levels.

This paper is organized as follow: in the next section we present the related work. Then we present the proposed mechanism and how it works; we show the experiments that were done in order to validate the mechanism and finally we present the conclusions of this paper and some future work.

Related Work

This section presents the main topics involved in this work.

Recommender Systems

Recommender systems provide easily accessible, high-quality recommendations for a large user community (Jannach et al. 2010). These systems try to learn the user preferences using information gathering techniques, recommend relevant information or alternatives for the search being performed and analyze the user feedback in order to improve next recommendations.

Different recommendation techniques may be applied in recommender systems. The two main approaches used are the content-based filtering and the collaborative filtering (Melville and Sindhwani 2010), being this last one the most popular approach (Ricci et al. 2011).

Collaborative filtering approach consists in comparing users through their profiles attributes and ratings searching for similarities. In (Massa and Avesani 2007), authors show that the collaborative filtering by similarity has weak points, for example, the need that users rate at least one item in common for the comparison to be done. The creation of a trust-based recommender systems can reduce the effect of the cold start problem, as shown in (Pitsilis and Knapskog 2012), where they propose a trust-based model where a new user receives recommendations of similar users based on the propagation of trust.

Trust-based models may solve the sparsity of information that can be used to create a relationship between users (Massa and Bhattacharjee 2004). Collaborative filtering takes into consideration user data and searches for recommendation given to user with similar interests. Content-based filtering, on the other hand, tries to recommend items similar to those the user searched or already bought, bringing similar results to these.

In (Quijano-Sánchez et al. 2012), for instance, authors present a case-base solution applied in a group recommender to deal with the cold-start problem. The system is able to copy ratings into a new user's profile from the profile of the most similar user in the most similar group from the case base.

Recommender systems can be feed by agents. An agent is autonomous and it may execute big part of its tasks without human intervention, as well as answer other agents or users requests when necessary. Also, an agent has social abilities and it is able to communicate and exchange information with each other agent. These interactions make the execution of agent tasks more reliable and optimized (Wooldridge 2002).

In multi-agent recommender system agents can be found performing specific roles. They may have different approaches from each other, considered a heterogeneous recommender system (Murugesan 1998), or they may work for the same goal, considered a homogeneous recommender system (Balabanovic and Shoham 1997).

As agents can return different information for the same recommender system request, it is important to have the evaluation of the information recommended. Considering this, we propose a mechanism that works in a multi-agent recommender system, evaluating the received recommendation on behalf of the user, sending this evaluation back to the system to allow the update of the trust levels of each agent involved in the recommendation.

Information Gathering There are two main approaches for gathering information in Recommender Systems: explicit and implicit (Aggarwal and Yu 2000). The explicit approach gathers information provided explicitly by the user, as interests, feedback, ratings, and others. In other way, the implicit approach does not need the user interaction directly to make a profile. This approach gathers information through actions as click number, accessing time on the page, similar user profile information, and so on.

The main advantage of the explicit approach is the fact that it can draw a better user profile, making the Recommender Systems more efficient in its recommendation. However, this kind of data is generally scarce, what makes the user profile not so rich in relevant information. As discussed in (Parra et al. 2011), even that the explicit feedback seems to be the best way of gathering information it can suffer with the user inconsistencies called *natural noise*. The implicit approach can get a bigger quantity of information by analyzing and qualifying aspects related to the user.

We choose to work with the implicit approach, having in mind that the focus of the paper will be in the social networks scope, so a great amount of data can already be gathered without the need of the user to report it again. It is believed that in this way, the system can obtain a rich profile in interests without the direct participation of the user.

The mechanism

This work presents the mechanism that exploits user information from social networks to update the user profile and to improve recommendations generated by a multi-agent recommender system.

Multi-agent recommender systems use the information from the user profile to generate the recommendations. In order to ensure good recommendations, the user profile should be updated according to changes in tastes and preferences of the user.

However, updating the user profile is not a simple task and the user information may be obtained in two different ways. Asking the user about some preferences (explicit way) or observing the user behavior (implicit way). Usually users do not like to waste time answering questions. Thus, to overcome this inconvenient to the user, we propose updating the user profile through the analysis of his/her social network account (Facebook).

Figure 1 presents the mechanism flow. When the user asks for a recommendation, the agents start to work to present the result. We can see that meanwhile the user interacts with the social network (Facebook) through his/her account, in the background, the mechanism runs to get user information to update the user profile and to evaluate the received recommendation when the process is finished.

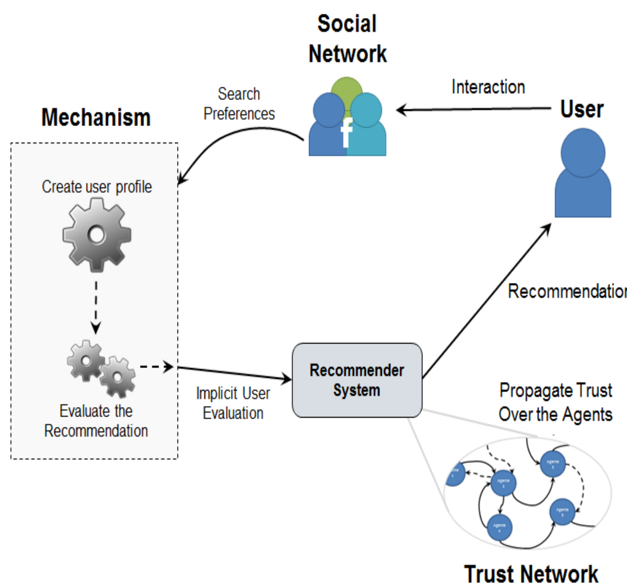

Figure 1: The Mechanism Flow

The user profile has an important role in multi-agent recommender systems because agents take into account the user preferences to compose a recommendation. Usually, the user profile keeps information regarding user preferences and/or behavior and it is considered a description of the user preferences and likes.

To give the mechanism the ability to understand and to create a user profile, we had created a terminology (list of words) to analyze the picture's caption and the area of interest of the places that the pictures have been taken. Having a database populated with words, places and words of value (words that demonstrate liking or disliking of something) and having put them in relation with the areas of interests being analyzed, it is possible to classify the user's pictures and create or update the user profile.

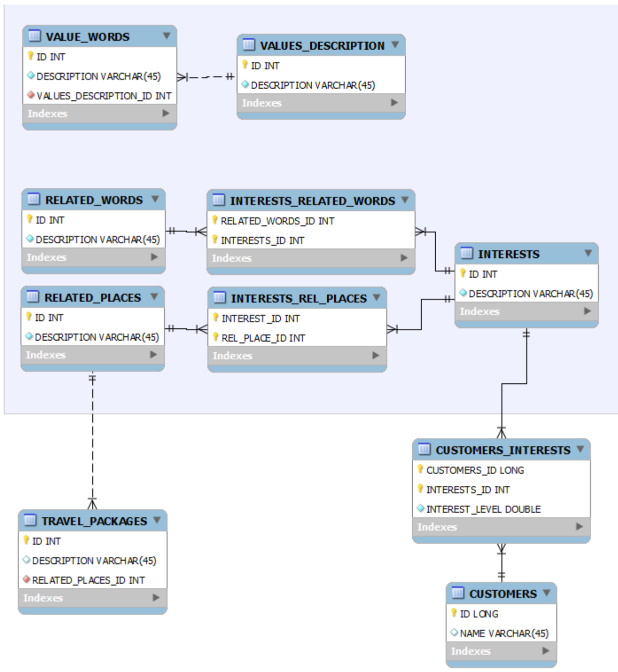

Figure 2: The created terminology

Figure 2 shows how the terminology was modelled in relation with other objects from the mechanism. First, we have the value words that will classify the description as positive or negative. For example, words like 'not', 'hated', 'horrible' will be considered as a negative description and, consequently, the user level of interest for this specific interest will be decreased. Second, we have the related words and the related places. Each of these words or places can be related to one or more interests, dividing them into different groups.

Creating the user profile The user profile is defined through user preferences that are extracted from Facebook. As figure 2 shows, there is the users' interests that are populated by the mechanism after extracting the data from the user Facebook account and analyzing it through the terminology.

The mechanism analyzes the Facebook user account and searches the user pictures albums where each picture has a location and a description posted by the user. Thus, with the terminology it is possible to read and understand the tags of the text posted in each picture, learning things the user likes and does not like.

When the user profile is created for the first time, all his/her interests start with a neutral value (represented by the value 0.5, assuming a classification range from 0 to 1,

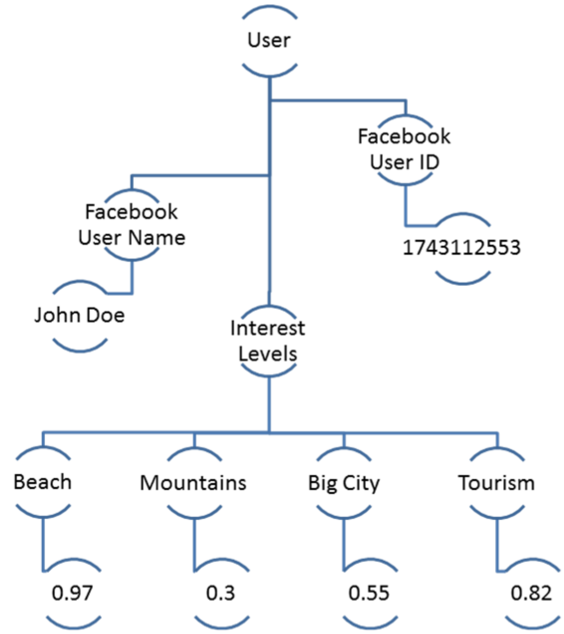

Figure 3: Example of user profile

being 0 the lowest level of interest and 1 the highest). Figure 3 shows an example of the user profile and its main attributes. After obtaining and analysing his/her preferences and pictures, this neutral value is added, subtracted or even kept (in case of not having information of that interest) with the values raised by the mechanism.

In Figure 4 it is possible to see an example of a picture posted by the user in Facebook. Using the terminology previously created, the mechanism separates the picture description and analyzes each expression to classify them into an interest, understanding if it is a good or bad description about the interest.

Obtaining user information from the pictures To obtain the information necessary to create the user profile, we decided to use Facebook's API to retrieve user data from his/her account. Using Facebook Query Language, it is possible to query Facebook database and retrieve information that the user can access on the usual Facebook interface. The mechanism then analyzes the descriptions picture by picture, following the interpretation process to classify the picture caption.

Using the defined terminology the mechanism starts to search first for value words, which will recognize if the description is a positive or a negative description.

After that, the mechanism checks for related words and related places in the picture description. During this search, some expressions may be found, for example, "I love beach" or "I think mountain is boring". These expressions also demonstrate how the user feels about some interests.

Each word or place that is found equal between the description and the terminology counts as 1 for the positive or negative evaluation of the description analisys, based on the value words found. After analyzing all the descriptions,

Figure 4: Example of the picture posted by the user

the values are put together for each interest. The Equation 1 shows the evaluation for each interest I, as follows:

$$I = \frac{\sum_{i=0}^{k} p_i - \sum_{j=0}^{m} n_j}{\sum_{i=0}^{k} p_i + \sum_{j=0}^{m} n_j} \times 0.5 \quad (1)$$

Where:

- I represents the interest level of an interest from the user;
- p represents the positive evaluation values, where $p = \{p_0, ..., p_k\}$;
- n represents the negative evaluation values, where $n = \{n_0, ..., n_m\}$.

The result from I is added with the initial 0.5 from the interest level in order to have the final evaluation of the interest for the user. Table 1 shows some examples of evaluation values using equation 1.

As we can see in algorithm 1, all the process of creating the user profile can be represented in a high level algorithm.

Creating evaluations on behalf of the user After creating the user profile, the mechanism uses the profile to evaluate the received recommendation on behalf of the user. Instead of asking the user to evaluate the recommendations through a form with questions or rates, the mechanism evaluates the recommendations in a implicit way, using the updated user profile.

To evaluate a recommendation made by the multi-agent recommender system, the mechanism gets the user profile previously created to use its interests levels. It analyses if the recommendation fits into one or more interests.

For example, let us consider that the recommendation has presented a travel package to *Pensacola Beach*. The mechanism searches in the terminology for the interests related to this city, returning all interests that fits with it (in this case beach and tourism city).

Then, the mechanism uses the interests levels of each returned interest, summing them and dividing by the number of returned interests. The resulting average is used as the recommendation evaluation. Still considering the *Pensacola Beach* example, if the user fully likes tourism cities (interest level equals 1) but fully dislikes beaches (interest level equals 0), the mechanism will evaluate it as neutral (evaluation of 0.5). The equation that represents this evaluation is represented in equation 2:

Table 1: Example of the applied equation on the evaluation of the user interest level

$\sum p$	$\sum n$	I	**Final Interest Level (0.5 + I)**	**Grade**
0	32	-0.50	0.00	Poor
2	20	-0.41	0.09	Poor
6	6	0.00	0.50	Neutral
42	31	0.08	0.58	Neutral
27	0	0.50	1.00	Good
15	1	0.44	0.94	Good

Algorithm 1 Creating user profile

```
getUserFacebookInfo() {If user profile does not exist in the database then it must be created}
if (!getProfileFromDB()) then
    getInterestList()
    createUserInterests() {Return the terminology}
    getValueWordsTerminology()
    getRelatedPlacesTerminology()
    getRelatedWordsTerminology()
    getPicturesCaptions()
    for (eachInterest) do
        compare(picture.caption, valueWords) {If a negative word is found, then set this description as negative}
        compare(picture.caption, relatedPlaces)
        compare(picture.caption, relatedWords)
        interestSum(positive,negative)
        setInterestLevel(interestSum)
    end for
    createUserProfile(user.FacebookID, user.FacebookName, listOfInterests)
end if
```

$$E = \frac{\sum_{i=0}^{k} I_i}{\sum_{i=0}^{k} q_i} \quad (2)$$

Where:

- E represents the total value of the evaluation of the recommendation. This value will be returned to the recommender system as being the user evaluation for the recommendation;
- I represents the interest level values of each interest related to the recommendation, where $I = \{I_0, ..., I_k\}$;
- q is the number of interests related to the recommendation, being $q = \{q_0, ..., q_k\}$.

Algorithm 2 shows the main steps of the evaluation algorithm used to implicitly evaluate the recommendation made to the user and explained earlier.

Algorithm 2 Evaluation algorithm

getProfileFromDB()
getRecommendation()
checkRecommendationRelatedInterests()
for (eachInterest) **do**
  getInterestLevel()
  setPartialEvaluation()
end for{Evaluates the recommendation}
eval = calcFinalEvaluation() {returns the final evaluation to the multi-agent recommender system}
return setFinalEvaluation(eval)

Experimental Results

The presented mechanism was validated through a multi-agent recommender system applied to the tourism domain. This system was developed and presented in (Lorenzi et al. 2011). In this system, agents work in a cooperative way to compose a whole recommendation to the user. Each agent is responsible for a part of the final travel package that is presented to the user. The recommendation algorithm is case-based, where each agent has its own case-base and may become an expert in a part of the travel package. The agents trust levels are updated according to the user evaluation of the received recommendation. Thus, we applied the presented mechanism in order to ensure that the recommendation system receive the evaluation on behalf of the user.

Figure 5 shows the evaluation screen that we developed to run the experiments. The multi-agent recommender system runs and send the recommendations to the evaluation screen that presents them to the user.

Students from the Computer Science courses were invited to participate of our experiment. We run the multi-agent recommender system in three different classes (a total of 67 students). Each student received a travel package recommendation and had to evaluate (manually) the received recommendation. Later, the mechanism has evaluated the same recommendations, considering the students profiles created from their Facebook accounts.

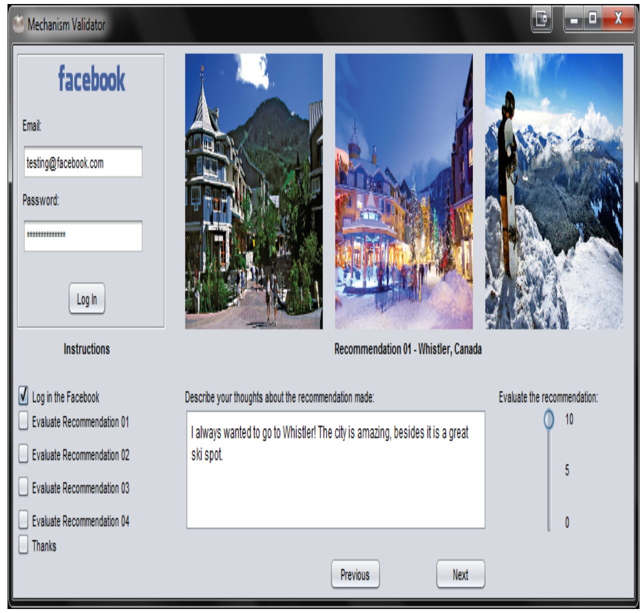

Figure 5: Evaluation screen

A comparison of the evaluations generated by the mechanism with the manual evaluations (by the students) establishes the validity of the presented mechanism.

Table 2 presents interesting results. We count all evaluations and we separated them in positive, negative and null (when the user did not evaluate the recommendation). We can see that using the mechanism we got a shy increase in the positive evaluations (79% comparing to 76% obtained from the manual evaluations).

Table 2: Comparison of the evaluations: manual x mechanism

Type of evaluation	Positive	Negative	Did not answer
Manual (students)	76%	19%	5%
Mechanism	79%	21%	0%

In the negative evaluations, the mechanism had an increase of 2% (21% comparing to 19% obtained from the explicit user evaluation). This could be considered as a negative result, but analyzing the values of the third column (did not answer) we can see that it was an improvement. Instead of having 5% of recommendations without evaluations, the mechanism has 0% and 21% of negative evaluations.

These negative evaluations can be use by the multi-agent recommender system to improve next recommendations. If the users do not evaluate, the system has no way to guess whether the recommendations presented were appreciated by the user. This is a drawback of getting the user evaluations in the explicit way because the user sometimes decides not to answer the evaluation questions. It means that the multi-agent recommender system may not have the evaluations of some users and maybe it can provide similar recommendations in the future.

With the implicit evaluation we ensure that the multi-agent recommender system always get the user feedback and this feedback may be used to update the user profile and to improve next recommendations.

In Figure 6 the chart shows the comparison of the average evaluation for each interest from the explicit way (manual) and implicit way (mechanism). As we can see, the implicit way kept a good proximity to the explicit evaluation average. For this difference, there are some possible factors that may have influenced: Manual evaluation tends to vary when a quantitative kind of evaluation is applied (different from simple qualitative 'like' or 'dislike' evaluations). For example, the same user may evaluate the same recommendation as a 8/10 and some time later evaluate it again with a 9/10. This is explained earlier in the paper as the *natural noise*. Another situation for this difference in results is the lack of information in the user profile, or the lack of matches between the terminology and the users pictures descriptions.

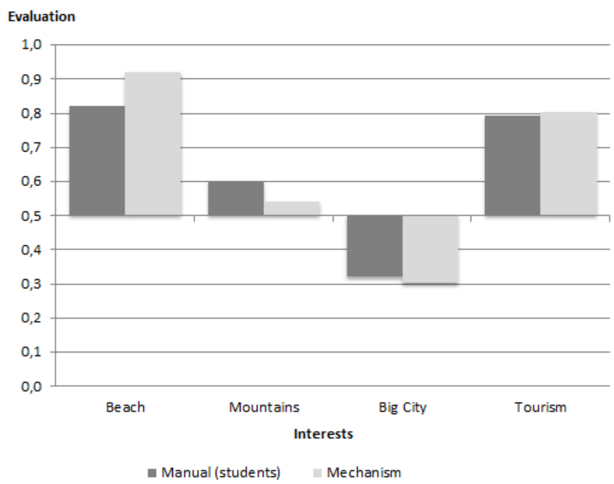

Figure 6: Evaluation screen

Conclusions and Future Work

This paper presented the mechanism capable of creating and updating a user profile and generating implicit evaluations on behalf of the user according to information gathered from her/his Facebook account. The user profile has an important role in multi-agent recommender systems because they help to improve next recommendations generated by agents.

Experiments were conducted to test the efficiency of the proposed mechanism. A real multi-agent recommender system (with an evaluation screen) was used in the experiments to recommend travel packages in the tourism domain. A comparison of the evaluations done by the volunteers and the evaluations generated by the mechanism establishes the validity of the presented mechanism. The results corroborate the idea that the proposed mechanism helps to update the user profile and it uses this profile to avoid null evaluations, i.e., the problem of the user not evaluating the recommendations. Moreover, the mechanism increases the positive evaluations.

As future work we intend to apply the proposed mechanism to other social networks. We would like to try to gather user information from *Linkedin* that is considered a business related social network.

Acknowledgments. This work was supported by the brazilian agency FAPERGS (ARD 003/2012).

References

Aggarwal, C. C., and Yu, P. S. 2000. Data mining techniques for personalization. *IEEE Data Engineering Bulletin* 23(1):4–9.

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

Jannach, D.; Zanker, M.; Felfernig, A.; and Friedrich, G. 2010. *Recommender systems: an introduction*. Cambridge University Press.

Lorenzi, F.; Bazzan, A. L.; Abel, M.; and Ricci, F. 2011. Improving recommendations through an assumption-based multiagent approach: An application in the tourism domain. *Expert Systems with Applications* 38(12):14703–14714.

Massa, P., and Avesani, P. 2007. Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems*, 17–24. ACM.

Massa, P., and Bhattacharjee, B. 2004. Using trust in recommender systems: an experimental analysis. In *In Proceedings of iTrust2004 International Conference*, 221–235.

Melville, P., and Sindhwani, V. 2010. Recommender systems. *Encyclopedia of machine learning* 1:829–838.

Murugesan, S. 1998. Intelligent agents on the internet and web. In *TENCON'98. 1998 IEEE Region 10 International Conference on Global Connectivity in Energy, Computer, Communication and Control*, volume 1, 97–102. IEEE.

Parra, D.; Karatzoglou, A.; Amatriain, X.; and Yavuz, I. 2011. Implicit feedback recommendation via implicit-to-explicit ordinal logistic regression mapping. In *CARS*, volume 231, 160. Chicago, IL, USA.

Pitsilis, G., and Knapskog, S. J. 2012. Social trust as a solution to address sparsity-inherent problems of recommender systems. *arXiv preprint arXiv:1208.1004*.

Quijano-Sánchez, L.; Bridge, D.; Díaz-Agudo, B.; and Recio-García, J. A. 2012. A case-based solution to the cold-start problem in group recommenders. In *Case-Based Reasoning Research and Development*. Springer. 342–356.

Ricci, F.; Rokach, L.; Shapira, B.; and Kantor, P. B. 2011. *Recommender Systems Handbook*. New York, NY, USA: Springer-Verlag New York, Inc., 1st edition.

Wooldridge, M. J. 2002. *An Introduction to MultiAgent Systems*. Chichester: John Wiley & Sons.