Content Based Recommender Systems

[1] P. Deivendran M.Tech, [2], Dr. T. Mala Ph.D, [3] B.Shanmugasundaram, M.C.A

[1] Assistant Professor, Department of Computer Applications.
Velammal Engineering College, Chennai.600 066, Tamilnadu, India
E-mail: deivendran77p@yahoo.com

[2] Assistant Professor, Department of Information and communication Engineering
Anna University, Chennai – 600 025, Tamilnadu, India
E-mail: malal@cs.annauniv.edu

[3] Assistant Professor, Department of Computer Applications.
Veltech Multitech Dr.Rangarajan Dr.Sakunthula Engineering College,
Avadi, Chennai.600 062, Tamilnadu, India
E-mail: ten1107@gmail.com

Abstract : *Recommender systems help users to identify particular items that best match their interests or preferences. In this paper, we introduce our approach to recommendation based on Case-Based Reasoning (CBR). CBR is a paradigm for learning and reasoning through experience, based on human reasoning. We present a user model based on cases in which we try to capture both explicit interests (the user is asked for information) and implicit interests (captured from user interaction) of a user on a given item. When we apply CBR to recommender systems, some problems arise such as the adaptation of user profiles according to their interests and preferences over time or the utility problem. In order to cope with these problems, our approach includes a "forgetting mechanism" based on the drift attribute. Other systems have implemented CBR approaches to commendation, but unfortunately, only a few evaluate and discuss their results scientifically. This paper also proposes an evaluation technique based on a combination of real user profiles and a user simulator. The results of the simulations show that the forgetting mechanism produces an increase in precision, a decrease in recall and an important reduction of the number of cases in case bases.*

Keywords *– Metrics, Measure, Case Based Reasoning, Cycle, Profile, and Precision.*

1. Introduction

In the real world, making a selection from the incredible number of possibilities the market offers us indeed a laborious work. The main function of the assistants is to advise you. In order to do this, first of all, they have to learn your tastes, interests and preferences. Then, their task consists of looking for information and analyzing[5] the market in order to find out things that may interest you. Since personal assistants are always in contact with you, they also notice your changing interests over time. If you cease to be interested in a certain thing, your personal assistant takes note and finds out what you are presently interested in. Recommender systems draw on previous results from machine learning and other AI technology advances. Among the various machine-learning technologies, we concentrate on Case-Based Reasoning (CBR) as a paradigm for learning and reasoning through experience, as personal assistants do. The main idea of CBR is to solve new problems by adapting the solutions given for old ones. However, when we apply CBR to recommender systems, there are two things missing.

Humans have a vast store of experience on which to base their decisions. When a new problem comes up, humans look for similar problems and try to solve it based on the most similar experiences. However, the time dimension is also present in the human reasoning process. It means that humans have in mind the most recent cases and give them the greater importance when making a decision. When we are dealing with human interests and preferences, the relevance of the most recent cases becomes even more important.

2. Case-Based Recommendation Framework

The core of CBR is a case base which includes all the previous experiences that can give us information we can use to deal with new problems. Then, through the similarity concept, the most similar [12] experiences are retrieved. However, similarity is not a simple or uniform concept. Similarity is a subjective term that depends on what one's goals are. For instance, two products with the same price would get maximum similarity if the user was interested in products with that same price, but would get very different similarity for other concepts, such as quality [11] or trademark. In our approach, the case base represents the user profile and consists of a set of previous experiences (cases); that is, items explicitly and/or implicitly assessed by the user. Each case contains the item description (attributes describing a restaurant in the example) and the interest attributes describing the interests of the user concerning the item. These latter attributes can be explicitly given by the user or implicitly captured by the system.

This kind of recommendation based on similar items is our approach to content-based filtering. With regard to the CBR cycle, we reassess the different phases as follows:

a. In the retrieval phase, i.e. a new item, the system searches for similar items in the case base in order to find out whether the user might be interested in them. Local similarity measures are based on item attributes [14].

b. In the reuse phase, i.e. the retrieved set of similar items, the system calculates a confidence value of interest to recommend the new item to the user based on explicit and implicit interests and the validity of the case according to the user's current interests [8].

c. In the revision phase, i.e. the relevance feedback of the user, the system evaluates the user's interest in the new item. The idea is to track user interaction with the system to get to know relevant information about the user's interest in the recommended item, as well as explicit and implicit information, in order to retain the new case.

d. In the retain phase, the new item is inserted in the case base with the interest attributes that were added in the revision phase. In order to control the case base size, it is also important to know if the user ever gives new feedback [6] about items in the case base. In such a case, it is necessary to forget these interests with time. We propose the use of a new attribute that we call the drift attribute, which will be aware of such changes in user preferences and contribute to case maintenance. In the following sections the structure of the case base and the different CBR phases of the new approach are detailed.

3. Evaluation Metrics

A set of metrics [6] are proposed in order to evaluate recommender systems: precision, recall, measure, fallout, cases, diversity and accuracy.

3.1 Precision

The Precision measure is the fraction of the selected items which are relevant to the user's information need. It is also a measure of selection effectiveness and represents the probability [4] that a selected item is relevant. Precision is calculated with the following formula:

$$P = \frac{s}{n}$$

Where *s* is the number of successful recommendations and *n* is the number of recommendations. The result is a real value ranging from 0 to 1. Precision [9] can also be seen as the probability that a recommendation be successful.

3.2 Recall

The Recall measure is the fraction of the actual set of relevant items which have been correctly classified as relevant. It is a measure of selection effectiveness and represents the probability [13] that a relevant document will be selected. It is interesting to evaluate the number of recommendations that the system makes, since; of course, a recommendation algorithm that recommends all the items will obtain all the possible successes. Recall is computed as follows:

$$R = \frac{n}{t}$$

Where *n* is the number of recommendations and *t* is the total number of possible recommendations. The result of this formula is a real number ranging from 0 to 1. Recall can also be seen as the probability that an item be recommended.

3.3 F-Measure

It is, on occasion, important to evaluate precision and recall in conjunction, because it is easy to optimize either one separately [15]. The F-Measure consists of a weighted combination of precision and recall which produces scores ranging from 0 to 1. When recall increases, precision decreases. Weighting measure between precision and recall called the f-measure. However, we have used a variation of this measure, where the weights[9] are controlled by a parameter *b* [4]. This new approach is calculated as follows:

$$FM = \frac{(b^2 + 1) * P * R}{b^2 * P + R}$$

Where *P* is precision, *R* is recall and *b* is the weighting factor [1]. For example, b = 0.0 means that *FM = precision*; *b = unlimit* means that *FM = recall*; *b* = 1.0 means that recall and precision are equally weighted; *b* = 0.5 means that recall is half as important as precision; and *b* = 2 means that recall is twice as important as precision. We can also see this measure as a modification of precision by recall.

3.4 Fallout

The Fallout measure is the fraction of the non-relevant items selected. It is a measure of rejection effectiveness. We use Fallout to evaluate the percentage of failed recommendations. It is computed like precision, but instead of measuring the recommendations successfully evaluated by the user, we take into account the number of recommendations that the user has valuated as bad. Fallout is calculated with the following formula:

$$F = \frac{u}{n}$$

Where *u* is the number of failed recommendations and *n* is the number of recommendations. Fallout can also be seen as the probability[5] that a recommendation be a failure. The result is a real value confined to the [0-1] interval, although fallout charts represent F normalized between 0 and 100. A fallout value close to 0 means that the system never recommends bad choices; a fallout value of 1 means that the system is always recommending uninteresting items to the user.

3.5. N Cases

The study of the average number of items (cases) contained in the user profile (case base) over time is very important, since it is desirable to reduce the size of the user profiles (solving the utility problem) while preserving or even increasing precision (while adapting the profile to the user). Certainly, the forgetting mechanism will reduce the time and the capacity needed by the algorithms to perform a recommendation.
Thus, Cases is calculated as follows:

$$NC = \frac{\sum_{i=0}^{k} |NC_i|}{k}$$

Where *NCi* is the number of items at the moment *i*, and *k* is the number of moments. That is, the simulation time has been split into *k* units and, in each unit, the number of cases in the case base *NCi* has been measured. At the end of the simulation, the average is computed. Cases is not normalized, therefore, this number is relative to the total number of possible recommendations. What we want to study is the difference between the different Cases from the point of view of different parameters that the forgetting mechanism depends on.

3.6 Diversity

How the reduction of the number of items contained in the user profile affects the diversity within the resulting profiles is an interesting phenomenon for study. To evaluate the diversity, we propose using a well-known clustering method that calculates the number of groups of similar items contained in the profile. The clustering algorithm that we have implemented belongs to a particular subset of clustering methods knows as SAHN[11]: Sequential, Agglomerative, Hierarchical and Non-overlapping methods. The proposed algorithm can be summarized as follows:

3.7 Evaluation Methods

STEP 0: Construction of an initial similarity matrix that contains the pair wise Measures of proximity between the different items of the user profile.

STEP 1: Selection of the two items that are most similar. These alternatives will form a new cluster.

STEP 2: Modification of the similarity matrix creating a cluster with the selected items and recalculating the similarity between the new cluster and the remaining objects. Similarity is calculated with an Arithmetic Average criterion where the similarity between a given item and the cluster is the average similarity between the items composing the cluster and the given item.

STEP 3: Repeat steps 1-2 until the two most similar items have a similarity value over a threshold α. This threshold has to be defined previously, taking into account that it determines the abstraction level achieved. Increasing the threshold we obtain a smaller number of wider (more general) clusters [13]. The number of clusters obtained after the execution of the proposed algorithm is the diversity measure that allows system simulations performed with different parameters to be compared. Thus, a key task is to select a suitable α. Depending on this parameter, the number of clusters constituting the user profile will change. A low α means that only the most similar cases join up and, therefore, the algorithm gives a high number of clusters.

4. Profile Discovering

In order to solve all the shortcomings of the current techniques while benefiting from their advantages, we propose a method of results acquisition called "the profile discovering procedure". This technique can be seen as a hybrid approach between real or laboratory evaluation, log analysis and user simulation. First of all, it is necessary to obtain as many real user profiles as possible. These profiles must contain subjective assessments of the items (preferably explicit evaluations of the user, although the implicit information obtained from the user interaction with the system is also useful). It is desirable to obtain these user profiles through a real or laboratory evaluation although it implies a relatively long period of time. However, it is also possible and faster to get the user profiles through a questionnaire containing all the items which the users have to evaluate. Once the real user profiles are available, the simulation process; that is the profile discovering procedure starts. It consists on the following steps:

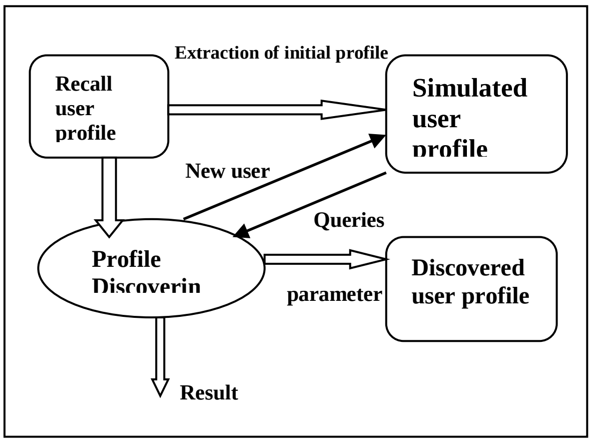
a. Generation of an initial user profile (*UP*) from the real user profile (*RUP*, $UP \subset RUP$).

b. Emulation of the real recommendation process, where a new item (*r*) is recommended From the *UP[9]*.

c. Validation of the recommendation: Otherwise, *r* is rejected.

d. Repeat 2 and 3 until the end of the simulation.

As in the real evaluation, the simulation process starts with the generation of an initial user profile[2]. It is desirable to initially know as much as possible from the user in order to provide satisfactory recommendations from the very beginning. Analyzing the different initial profile generation techniques, namely: manual generation, empty approach, stereotyping and training set, we found different advantages and drawbacks. In manual generation, the user tailors his or her own profile.



(Fig. 1 User Profile Structure)

5. Conclusion

This paper has focused on the study of recommender systems. In particular, we have proposed a recommender system consisting of collaborative recommender agents based on case-based reasoning (CBR) and trust. The CBR cycle has been redefined in order to perform the recommendation task. Assuming that the user has similar interests in similar items, the recommender system predicts the 195 user preferences in new items from the implicit/explicit interest given by the user in similar items.

6. Future Work

The design of a recommender system involves the consideration of a wide range of questions. In addition to the different solutions which have been adopted and described in this paper, many ideas have been proposed, discussed and finally rejected. On the other hand, other questions have remained as undeveloped ideas, which need to be analyzed further and worked on in depth in future work.

References

[1] I. F. A. Iamnitchi, M. Ripeanu. Small-world file-sharing communities. In The 23rd Conference of the IEEE Communications Society (InfoCom 2004), Hong Kong, 2004.

[2] G. Ruffo, R. Schifanella, E.Ghiringhello A Decentralized Recommendation System based on Self-Organizing Partnerships In IFIP-Networking'06, May 2006, LNCS 3976:618-629, Coimbra (Portugal), 2006.

[3] R. L. Jun Wang, Johan Pouwelse and M. R. J. Reinders. Distributed collaborative filtering for peer-to-peer file sharing systems. In Proc. of the 21st Annual ACM SAC, New York, NY, USA, 2006. ACM Press.

[4] B. Krulwich. Lifestyle finder: Intelligent user profiling using large-scale demographic data. AI Magazine, 18(2):37–45, 1997.

[5] K. Lang. NewsWeeder: learning to filter netnews. In Proc. of the 12th ICML, pages 331–339. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1995.

[6] M. E. J. Newman. The structure and function of complex networks. SIAM Review, 45:167, 2003.

[7] M. J. Pazzani, J. Muramatsu, and D. Billsus. Syskill webert: Identifying interesting web sites. In AAAI/IAAI, Vol. 1, pages 54–61, 1996.

[8] A. Popescul, L. H. Ungar, D. M. Pennock, and S. Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In Proc. of UAI '01, pages 437–444, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[9] P. Resnick and H. R. Varian. Recommender systems - introduction to the special section. Communication ACM, 40(3):56–58, 1997.

[10] A. Tveit. Peer-to-peer based recommendations for mobile commerce. In WMC '01, pages 26–29, New York, NY, USA, 2001. ACM Press.

[11] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. Nature, 393(6684):440–442, June 1998.

[12] Y. Z. Wei, L. Moreau, and N. R. Jennings. A market-based approach to recommender systems.ACM Trans. Inf. Syst., 23(3):227–266, 2005

[13] Susan Gauch, Jeason Chaffee, and Alaxander Pretschner. 2003. Ontology-based personalized search and browsing. Web Intelligence and Agent Systems 1, no. 3–4, pages 219–234.

[14] Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. Computational Linguistics 28, no. 3, pages 245–288.

[15] Internet Movie Database. http://www.imdb.com.