MAUT Based Recommendation

A Project Report

submitted by

BHARATH REDDY A

in partial fulfilment of the requirements
for the award of the degrees of

MASTER OF TECHNOLOGY
&
BACHELOR OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

April 2014

THESIS CERTIFICATE

This is to certify that the thesis titled **MAUT Based Recommendation** submitted by **Bharath Reddy A**, to the Indian Institute of Technology, Madras, for the award of the degrees of **Bachelor of Technology and Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Sutanu Chakraborti
Assistant Professor
Dept. of Computer Science and Engineering
IIT-Madras, 600 036

Place: Chennai

Date:

ACKNOWLEDGEMENTS

I am very thankful to Dr. Sutanu Chakraborti ¡fill-TODO¿

I am ever grateful to my friends Skanda Raj, Shubranshu Shekar, Saurabh Gupta and Dileep for helping me a lot during the project.

I thank the Department of Computer Science as whole. I am immensely happy to say that I have got such a good experience during the last 5 years. For this, the credit must go to remarkable teachers like Dr. Madhu Mutyam, Dr. Shankar Balachandran, Dr. Sutanu Chakraborti, Dr. Ravindran and Dr. C Pandu Rangan

ABSTRACT

Most commercial recommender systems in practice use collaborative filtering (CF) techniques that rely heavily on user-ratings to make recommendations. However, CF may not perform well in high-risk product domains like cars, cameras, houses etc. where there a low number of ratings.Knowledge based Recommenders are used to provide recommendations in these scenarios. Users often want to define their requirements explicitly - ”The maximum price of PC should be *x* and HDD capacity should be atleast *500 GB*.” and engage in an interaction with the system. Thus, the recommendation process of a knowledge based recommender is highly interactive, and thus they are also characterized as *conversational recommender systems*. Conversational recommender systems mimic the kind of dialog that takes place between a customer and shopkeeper involving multiple interactions and where the user can give feedback at every interaction. *Critiquing* is a popular form of feedback in conversational recommendation systems.

Dynamic generation of appropriate compound critiques in each cycle is a critical issue for critique-based conversational recommender systems. In earlier research, Apriori algorithm and MAUT (Multi Attribute Utility Theory) based generation of compound critiques have been proposed. MAUT based recommendation has been shown to be slightly superior to Apriori Algorithm based recommendation in offline experiments and live user studies. ”Average number of interaction cycles per recommendation session” is a measure that is often used to measure the goodness of a recommendation algorithm. Lower the number of cycles, better is the performance of the algorithm. In this project, we propose several modifications to the MAUT based generation of compound critiques and report the improvements in performance caused by each of these modifications.

CHAPTER 1

Introduction

"Which digital camera should I buy? Which movie should I rent? Which book should I buy for my next vacation?" These are some situations where people have to make decisions about how they are going to spend money, or in a broader level, about their future. Traditionally, people have used a variety of strategies to solve such decision making problems: conversations with friends, obtaining information from a trusted third party, hiring an expert team or simply follow the crowd. In the present age where e-commerce is flourishing, most e-commerce sites have very large number(often in millions) of products in their databases. For a user wanting to purchase a product, examining all the products (eg: books) present in the catalog one after another in the hope of a finding an product that is of interest to him is impractical. We would like to have systems that assist the user to find products of his interest and enable him to efficiently navigate through the complex product space. *Recommender systems* are constructed for this purpose - assisting a user in his/her (online) decision-making. Recommender systems play an extremely important role in matching users to products or items that they might find interesting. They filter out huge amounts of information to give personalized suggestions that its users might be interested in. This reduces the cognitive effort on the users who are spared of the need to examine a large number of irrelevant items before reaching their desired product.

Recommender Systems are broadly classified into three categories: Collaborative, Content Based and Knowledge Based.

1.1 Collaborative Recommender Systems

The main idea in these systems is that if users share the same interests in the past - if they viewed or bought the same books - they will also have similar tastes in the future. This technique is also called as *Collaborative Filtering*(CF). Pure CF based approaches require only rating data and do not require the additional knowledge about underlying users/items. Hence, the algorithms are usually domain independent. Most commercial recommender sytems use collaborative filtering for recommending items. There are two approaches to do CF: Memory Based Approaches and Model Based Approaches

1.1.1 Memory Based Approaches

In this approach, the original rating matrix is held in memory and directly used to generate predicted ratings and recommendations. There are two popular memory based approaches:
User based Nearest Neighbor(NN) Recommendation: Given a user u, the system computes top K similar users to u according to a pre-defined similarity measure. It recommends those items to user that haven't been rated/purchased by u but liked by the top K similar users.
Item based NN Recommendation: Given a user u, the system recommends items that have received similar ratings to the ones that u had previously liked.

1.1.2 Model Based Approaches

As opposed to memory based approaches that use the ratings matrix to directly generate predictions, model based approaches learn models corresponding to each item and each user from ratings matrix and the learned models are used to make predictions at run time. Model based approaches perform well in practice for large datasets. *Matrix factorization* is a popular model based approach. The superiority of matrix factorization techniques over traditional CF in improving prediciton accuracy was clearly seen during *The Netflix prize* competition. Broadly speaking, matrix factorization methods derive a set of latent(hidden) factors from the rating patterns and characterize each item and user as vectors of these factors. In the movie domain, such latent factors can correspond to some aspects of a movie like genre, but most of them are completely uninterpretable (Koren *et al.* (2009))

1.1.3 Limitations of CF

Cold Start Problem: To provide recommendations for a user u, pure CF techniques rely on u's ratings. This means that for a new user who has not yet rated a single item, there is no way of generating personalized recommendations (*new-user problem*). Similarly, a new item that has been recently added to the catalog and has not been rated by a single user, has no possibility of being recommended to a user (*new-item problem*)
Sparsity: The relavance and accuracy of CF recommender's predictions is high when the user-item ratings matrix is dense. But in real-world systems, the rating matrices are typically very sparse and thus, the quality of recommendations of pure CF approaches may not be good. For example, a user u whose rating pattern is very different from most of the other users would find it difficult to receive useful recommendations because the number of similar users to u is very less(Balabanović and Shoham (1997)).

1.2 Content based recommender systems

Collaborative Filtering Systems do not require any knowledge about underlying users/items to make recommendations. As opposed to this, content based recommender systems rely on item descriptions and explicit/learned user profiles to recommend items. For example, if the recommender system knows that "Harry Potter" is a fantasy novel and the user *Alice* has always like fantasy novels, the system can recommend the new "Harry Potter" book right away. Content-based recommender systems need not rely on the existence of a large user base to generate recommendations. It overcomes the cold-start problem described in Section 1.1.3. However, item characteristics are hard to acquire normally and hence, they have to be entered manually into the system, which can be potentially expensive for some domains.

Having its roots in *Information Retrieval*(IR), content based recommendation most often focuses on textual products - items which can be described in terms of textual features (Eg: documents, news articles and websites). Most news recommendation systems use content based recommendation to recommend relevant news articles to the users. A news recommendation system typically recommends news articles by comparing the main keywords of a news article with the keywords that appeared in other articles that the user has rated highly in the past. There are two ways in which content based systems can create user-profiles- by explicitly asking the user to rate a set of items/topics/categories when the user is new to the system or by ”learning” the user profile automatically by examining the user’s past behavior/ratings. Learning user profiles from user’s past behavior can be expensive and sometimes not be accurate because of time-effects(user’s interests changing over time) and sparsity of user ratings. But it has the advantage that it requires no effort from the user.

1.2.1 Limitations of Content-based Recommendation

Limited Content Analysis: Content-based recommender systems perform a shallow content analysis which might not be sufficient in many scenarios (Jannach *et al.* (2010)). Particularly for recommending resources such as web pages, aspects other than the keywords like aesthetics, usability and correctness of hyperlinks play a part in establishing the quality of recommendations (Jannach *et al.* (2010)). Also, content based recommender systems using limited content analysis based on just keywords, have no way to distinguish between well written and poorly written articles, both of which use the same set of keywords. Feature extraction techniques for text documents is relatively mature, but the same cannot be said about many multimedia objects like images and videos Hence, the usability of content-based recommender systems is limited in multimedia domain. (Adomavicius and Tuzhilin (2005))

Overspecialization: Another drawback of content based recommender systems is that they often tend to recommend items that the user might have already seen/rated. A general goal therefore is to increase the serendipity of the recommendation lists - that is, to include ”unexpected” items in which the user might be interested in. The system described by Billsus and Pazzani (1999) therefore defines a threshold to filter out not only items that are too different from the profile but also those that are too similar.

New user problem: The cold-start problem discussed in Section 1.1.3 also exists for content based recommender systems. Although content-based techniques do not require a large user community, they require at least an initial set of ratings from the user, typically a set of explicit *like* and *dislike* statements. The prediction accuracy of these systems improves with increase in the number of ratings.

1.3 Knowledge based recommender systems

Typically, we do not buy a house, a car or a computer very frequently. In such a scenario, both collaborative-filtering or content-based recommender systems may not be able to generate relavant recommendations because of the low number of available ratings. User-profiles learnt by content-based systems may not be useful for making recommendations due to a heavy influence of time effects(change in user's interests and product catalogs with time). For example, if a user has given a high rating to a 'Pentium III' computer four years ago, we cannot rely on that rating for generating relevant recommendations. In complex and high-risk product domains such as computers, customers often want to define their requirements explicitly - for example, "the maximum price of the computer should be x and the hard-disk capacity should be atleast *500GB*". Knowledge based systems are used to provide recommendations in such scenarios. Recommendation process of knowledge-based recommender applications is highly interactive, a foundational property that is a reason for their characterization as ***conversational systems***. The recommender system that we consider in this project is a conversational recommender system.

Conversational systems assume that a user's initial query is merely a starting point for search, perhaps even an unreliable starting point. The job of a conversational system is to help the user refine his initial preference query as the interactions proceed. Knowledge-based recommender systems can be divided into two classes: constraint-based and case-based recommender systems.

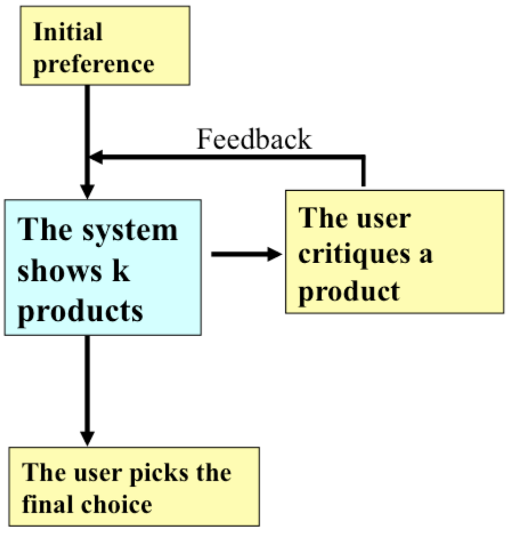
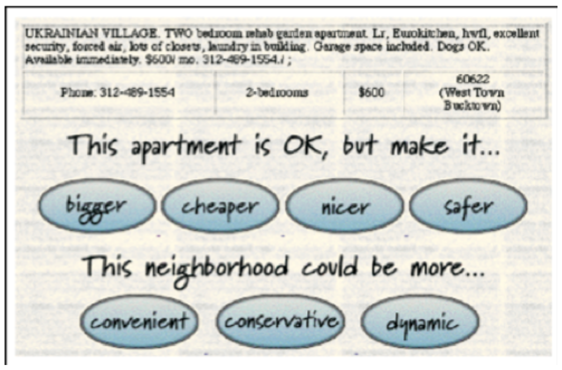

Initial preference
Feedback
The system shows k products
The user critiques a product
The user picks the final choice

Figure 1.1: Critiquing



UKRAINIAN VILLAGE. TWO bedroom rehab garden apartment. Lr, Eurokitchen, hwfl, excellent security, forced air, lots of closets, laundry in building. Garage space included. Dogs OK. Available immediately. $600/ mo. 312-489-1554./ ;

Phone: 312-489-1554	2-bedrooms	$600	60622 (West Town Bucktown)

This apartment is OK, but make it...
bigger
cheaper
nicer
safer
This neighborhood could be more...
convenient
conservative
dynamic

Figure 1.2: RentMe Recommender System: Burke (2000)

Constraint-based recommender systems rely on explicit recommendation rules and case-based recommender systems use similarity/utility measures to generate recommendations. Specifically, constraint-based recommendation is generated by looking at a collection of items satisfying the recommendation rules; case-based recommendation is typically generated based on similarity of the items in the database with the user defined query. MAUT based recommendation falls in the category of case-based recommendation. We discuss case-based recommendation in detail in Chapter 2

1.3.1 Critiquing

Critiquing is one of the most popular forms of feedback in conversational recommender systems. In each interaction cycle, the user is presented with a list of products. User selects a product and expresses directional preference(s) over one or more item feature values. For example, one might indicate that he/she is looking for a less expensive restaurant or a more formal setting(Figure 1.2). These are two individual critiques, first critique being on the *price* attribute and the second critique on the *setting* attribute. The recommender updates it's user model according to this feedback provides another set of products and proceeds to the next recommendation cycle. This continues till the user finally chooses a product. (Figure 1.1)

Unit critiques allow users to express their preference over one attribute in each interaction cycle. ***Compound critiques*** enable users to input their preferences on several attributes at a time. This can potentially shorten the number of interaction cycles in finding a target product. The early FindMe Systems Burke *et al.* (1996) had ***static critiques***. The critiques wouldn't change when users selected a particular critique. This can lead to some serious limitations. For example, the critique 'cheaper' would continue to be visible, even if there are no cheaper apartments available and when user clicks on 'cheaper', there would be no results displayed at all. Static critiques also do not represent the best set of tweaks that a user will want to make given his preference model. The notion of ***dynamic critiquing*** was first proposed by McCarthy *et al.* (2004) to overcome the limitations of static critiques. Compound critiques are generated on-the-fly for each recommendation cycle. Dynamic critiquing has been shown to improve user-experience and lower the average number of interaction cycles it takes for a customer to find his desired product.

There are two popular approaches to dynamic critiquing: Apriori algorithm based generation of compound critiques (McCarthy *et al.* (2004)) and MAUT based generation of compound critiques (Zhang and Pu (2006*a*)). The algorithm for MAUT based recommendation is discussed in Chapter 2.

1.4 Our contribution: Improvements to MAUT based recommendation

1.5 Organization of the Thesis

CHAPTER 2

Background & Related Work

2.1 Case-based Recommendation

Case-based recommendation traces its roots to case-based reasoning (Aamodt and Plaza (1994)). Case-based reasoning is a problem solving methodology which makes use of a case-base (database) of past problem solving experiences as its source of knowledge. A typical case in a case-base consists of a *problem specification* outlining the problem and a *solution part* which describes the solution used to solve the corresponding problem. Given a new problem at hand with a problem specification P_s, a case S is retrieved from the case-base whose problem specification is similar to P_s and then the solution of S is adapted to come up with a solution for P_s. (Smyth (2007)) Case-based recommender systems are particularly suitable for generating recommendations when we are dealing with structured representations of items and there are similarity measures that can be defined across features in the particular domain. Many e-commerce websites deal with products such as cameras, computers etc. which are usually represented in terms of their features in a structured way. Once suitable similarity measures are devised, case-based recommender systems are ready to be taken to the field.

Consider a user who specificies the following query to the system: ″I need a camera having a resolution of 8 Megapixels, manufactured by Canon, with a price less than \$500″. A case-based recommender might retrieve all cameras that have ′Canon′ as their manufacturer and are similar in terms of ′Price′ and ′Resolution′ as mentioned in user′s query and display them as recommendations. The similarity between a query q and a camera C, similarity is estimated according to weighted similarity model as:

$$Similarity(q, C) = \frac{\sum_{i=1}^{n} w_i \times sim_i(q_i, C_i)}{\sum_{i=1}^{n} w_i} \tag{2.1}$$

According to equation 2.1, the similarity between a user query q and camera C is estimated as a weighted sum of individual similarities between the corresponding features of q and C. n is the total number of features. The weights associated with each feature reflect the importance of that feature in the overall similarity calculation process and individual feature level similarities are calculated according to the similarity function pertaining to the particular feature i which is denoted by $sim_i(q_i, C_i)$. These feature level similarities are referred to as local similarities, which are defined by domain experts. The similarity between two price values p_i and p_j can be defined as follows:

$$sim_{price}(p_i, p_j) = 1 - \frac{|p_i - p_j|}{max(P) - min(P)} \tag{2.2}$$

$max(P)$ and $min(P)$ refer to the maximum and minimum values of $price$ feature in the database respectively. As we can see from Equation 2.2, greater the difference between p_i and p_j, lesser is the similarity between them. To estimate the similarities between the nominal feature (Eg:"manufacturer") values of two cameras, specialized domain knowledge is required. Case-based recommenders can be classified into two categories - 'Single-shot systems' and 'conversational systems'.

2.1.1 Single-shot Systems

Single shot systems are reactive systems which respond to user's query by showing him a single list of k items in a single interaction. An analog device recommender recommends those op-amps to the user which are most similar to his query (Wilke *et al.* (1998)). Recommender systems perform well in general if their results are diverse. There have been several attempts done to achieve this goal.

The *Bounded Greedy Selection* procedure described in Smyth and McClave (2001) has been shown to be giving the best results in many recommendation scenarios. In this method, the retrieval set R is iteratively constructed till it contains k items. The set S containing top bk items are considered in the beginning of recommendation procedure. Item which has the highest $quality$ score (Equation 2.3) amongst all items in set S is added to the set R and removed from S. The $quality$ scores are re-computed in the next iteration and the item with highest $quality$ score is promoted from set S to set R. This continues till there are k items in the set R. $quality$ score of a product wrt the retrieval set R is computed as follows:

$$Quality(q, P, R) = similarity(q, P) * RelDiversity(p, R) \tag{2.3}$$

$$RelDiversity(p, R) = 1 if R = \{\} = \frac{sum_{i=1}^{m}(1.0 - similarity(p, r_i))}{m}, otherwise \tag{2.4}$$

2.2 Benefits of Critiquing Systems

Over the past decade, a variety of critique-based recommendation methodologies have been proposed. Researchers have demonstrated the benefits of employing critiquing as the form of feedback in conversational recommender systems (e.g., Show me more like item A, but cheaper).

The primary reason why critiquing has become so popular is that it strikes an acceptable balance between the effort that a user must expend when providing feedback and the information value it provides. In comparison to the standard value elicitation approach, critiquing is a very low-cost form of feedback (i.e., in terms of user effort) that provides a relatively unambiguous indication of the user's current requirement Critiquing is also well-suited to even the most basic interfaces and to users with only a rudimentary understanding of certain recommendation domains.

In many domains, we cannot assume that users will be able to express their preferences at the beginning of interaction. Most users will not have an idea of the trade-offs/compromises that exist. Instead, as users become more familiar with the domain and the product options available, their preferences often change, becoming more rigid. Critique-based conversational recommenders offer support while users navigate product catalogues and help them to better understand their preference requirements. Instead of requiring users to specify their preferences from the outset, user preferences are built up over a series of *recommendation cycles*. In each cycle of a recommendation session the system makes one or more recommendations to the user and invites them to critique one of the examples. Feature critiques typically take the form of *directional* or *replacement* critiques. Through directional critiques can express a request to increase or decrease over one or more numeric attribute values (e.g., cheaper implies [¡ price]). Through replacement critiques, user can request for the substitution of any value (i.e., aside from critiqued value) for a non-numeric feature (e.g., different manufacturer implies [! = manfacturer]).

REFERENCES

1. **Aamodt, A.** and **E. Plaza** (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, **7**(1), 39–59.

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

3. **Balabanović, M.** and **Y. Shoham** (1997). Fab: content-based, collaborative recommendation. *Communications of the ACM*, **40**(3), 66–72.

4. **Billsus, D.** and **M. J. Pazzani**, A personal news agent that talks, learns and explains. *In Proceedings of the third annual conference on Autonomous Agents*. ACM, 1999.

5. **Burke, R.** (2000). Knowledge-based recommender systems. *Encyclopedia of library and information systems*, **69**(Supplement 32), 175–186.

6. **Burke, R. D.**, **K. J. Hammond**, and **B. C. Young**, Knowledge-based navigation of complex information spaces. *In Proceedings of the national conference on artificial intelligence*, volume 462. 1996.

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

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

9. **McCarthy, K.**, **J. Reilly**, **L. McGinty**, and **B. Smyth**, On the dynamic generation of compound critiques in conversational recommender systems. *In Adaptive Hypermedia and Adaptive Web-Based Systems*. Springer, 2004.

10. **Smyth, B.**, Case-based recommendation. *In The adaptive web*. Springer, 2007, 342–376.

11. **Smyth, B.** and **P. McClave**, Similarity vs. diversity. *In Case-Based Reasoning Research and Development*. Springer, 2001, 347–361.

12. **Wilke, W.**, **M. Lenz**, and **S. Wess**, Intelligent sales support with cbr. *In Case-based reasoning technology*. Springer, 1998, 91–113.

13. **Zhang, J.** and **P. Pu**, A comparative study of compound critique generation in conversational recommender systems. *In Adaptive Hypermedia and Adaptive Web-Based Systems*. Springer, 2006*a*.

14. **Zhang, J.** and **P. Pu**, A comparative study of compound critique generation in conversational recommender systems. *In Adaptive Hypermedia and Adaptive Web-Based Systems*. Springer, 2006*b*.