Using Social Network Services as an Input for a Trust Clustered - Collaborative Filtering Recommendation System

TEO ETEROVIĆ, BENJAMIN KAPETANOVIĆ, DZENANA DONKO
Faculty of Electrical Engineering
University of Sarajevo
Zmaja od Bosne bb
Bosnia and Herzegovina
{ teo.eterovic, bkapetanovic, ddonko }@etf.unsa.ba

Abstract: - In this paper we present a Recommendation system that uses the Social Network Services like Facebook Connect, OpenSocial etc. to get useful interest and friend interest data as an input for a Trust Clustering – Friends interest clustering applied on the results of a Collaborative filtering algorithm. We also discuss the advantages and disadvantages of Collaborative filtering and Trust based Clustering and how our solution uses the advantages and solves the disadvantages by combining them.

Key-Words: - social networks, recommendation system, social clustering, trust clustering

1 Introduction

For a long time the only way to search through big unstructured repositories was keyword based search. With the introduction of recommendation algorithms based on collaborative filtering such as XEROX Tapestry in 1992 [1] the users had a totally new way of browsing through digital repositories based on browsing interest clusters.

At this time, many recommendation algorithms use ether the item based or the user based collaborative filtering [2].

Protocols such as the OpenSocial, Facebook Connect etc. are services/interfaces given by the Social Networks such as Facebook, Orkut and MySpace that allow the user to easily register on websites enabling the developers to access more reliable information about the user. After the user agrees with the conditions of sharing, the system gains access to the social data such as the user's and user friend's interests. The benefit for the user is that he doesn't have to enter all the information when registering on a website, it's just copied from the social network.

These services – the social network data available through the API, gives the designers of the intelligent web [3] completely new abilities. The results of Recommendation systems are highly dependent on the amount and quality of social/interaction data collected / produced by the users that are used to make interest clusters formed between users and items.

One of the major problems for new websites is the lack of social/interaction data to “feed” the algorithm. This can result in bad recommendations and lead to the wrong way rather than being useful.

2 Problem Formulation

The recommendation algorithm needs a large amount of quality historical interaction/social data in order to give overall quality results and form clusters. The collection of relevant data usually requires a long period of time for new web applications. Another solution would be to use an existing dataset like the BX-Dataset for digital libraries and bind the ratings with our system. It is hard to find such datasets because they represent one of the key competitive advantages for eCommerce applications.

The recommendation algorithms based on collaborative filtering (SlopeOne etc.) usually use only the data based on the user's interaction on the website (item-to-item, ratings etc.). For example the SlopeOne algorithm is based on the principle that people tend to relay on the advice of others who have made the decision before [4] – in the case of SlopeOne algorithm it would be the item ratings. In real world the users usually ask their friends for advice. Research has been made [5,6,7] on “All Consuming” dataset that showed that the users were significantly more similar to their trusted peers than to the population as a whole.

For example let's assume that we have two students, one in Berlin and the other in Hamburg that are interested in a book about Operating Systems. If we use the CF algorithm, there is a high possibility that the same book would be suggested to both students. The problem could be if the different professors use different teaching books for the course. This problem could be easily solved by using the friend suggestions – it is logical that we have many friends from the faculty who bought/rented the same book. Also DuBois et al. gives very good arguments about the advantages of the trust based systems and compares the Trust based recommendation systems with Collaborative filtering shown in [8].

But if we only use the friend based suggestion algorithm the problem would be if we don't share the same interests with our friend. It would result in unusable suggestions as discussed in [2].

3 Problem Solution

In this paper we suggest a solution that uses the correctness of the Collaboration filtering and combine it with the analysis shown in [5,6,7]. This research show that users were significantly more similar to their trusted peers than to the population as a whole, whereby we use the CF algorithm to avoid the problems of Trust based recommendation systems discussed in [2].

3.1 Data gathering

An earlier discussed problem is data gathering. It is especially hard to get the social data like interests or friends that user usually skips when registering on a new website, but that are crucial information for Trust based recommendation.

We suggest using the dataset of popular social networks like Facebook, Orkut, MySpace available through standard APIs like Facebook Connect and OpenSocial for registration in order to get quality social data like interests and the interests of users friends.

3.2 The Algorithm

The first step is the user's registration on the web application and data gathering using standard APIs like Facebook Connect, OpenSocial etc. The user gives the Recommendation system the access to his interests, information about his friends and their interests.

The actual algorithm begins when the user selects an item, for example a book. First we find all users friends that are registered on the website and have rated the book (we could also use the information about who bought the book – Amazon patent). After that we form clusters based on friends interests for Employers, College, Music, Books, Television, Interests, Religion, etc. based on the values we construct clusters and the topic of the website for example "Faculty of Electrical Engineering", "ACME Inc." etc. and assign the friends to categories.

Next we use some of Collaborative filtering algorithms – for example SlopeOne and get all recommendations for the selected book. After that we put the resulting books of the CF into interest groups and resort them according to the clusters user ratings of the resulting books. This weight between the book and the cluster can be calculated as:

$$\omega = \frac{\sum_{i=1}^{|FR|} rating(FR_i, selected\_book)}{|FR|} \qquad (3)$$

Where FR is the set of users that have rated the book of the selected cluster, and |FR| the cardinal number.

If we visualize the result of the algorithm we get something like shown in Fig. 1. that enables us to see which friends have reviewed the book(we only rate then) and so we could ask them for advice in the real world. Also the navigation through the graph is intuitive for the user, as discussed in [2]. The practical thing is that we can easily express the relevance of a friend or book by scaling it by the factor of relevance. If we look at Fig. 2 – in Amazon suggestions for the same book there is a mix of professional and scientific-academic books making the overview harder if we are looking only for scientific books.

Fig. 1: Results for the book „Algorithms of the Intelligent Web"

Fig. 2: Amazon results for the book „Algorithms of the Intelligent Web" – for larger view goto Amazon

ISSN: 1792-6157

ISBN: 978-960-474-245-5

Notation	Meaning
$F=\{u_1,\ldots,u_n\}$	The set of n friends of the user
$B=\{b_1,\ldots,b_x\}$	The set of n books from the catalog
$FI=\{u_1,\ldots,u_m\}$	The set of m friends given as the result of the mapping function (Figure 4)
$K=\{C_1,\ldots,C_j\}$	The set of clusters
$C_j=\{u_1,\ldots,u_n\}$	The cluster of users
$U_k=\{i_1,\ldots,i_c\}$	The set of user k interests
$c_{name\ \ j}$	The name of the cluster j (the name of the interest like „Faculty of Electrical Engineering“)
$slopeOne(item)$	Function that returns the SlopeOne reuslt set for the item as described in [9]
$selected\_item$	Current item selected by the user
$S=\{r_1,\ldots,r_n\}$	The set of SlopeOne results
$\lvert X\rvert$	Cardinal number of set X
$CB_i=\{b_1,\ldots,b_n\}$	Set of books for cluster i
$rating(u,b)$	The rating value that the user u gave to book b

Given a set of $B=\{b_1,\ldots,b_x\}$ books
Given a set of $F=\{u_1,\ldots,u_n\}$ friends
a function has_rated:

$$R(u,x) \to \begin{cases} true & rating(u,x) \neq \emptyset \\ false, & else \end{cases} \quad (4)$$

for i:=1 **to** |F| **do** // friends that rated selecte_book
  if has_rated(u_i, selected_item) **then**
    FI := FI ∪ { u_i }
  end if
end for
for j:=1 **to** |FI| **do**
  K := K ∪ (U_j ≔ $\{i_1,\ldots,i_c\}$ \ K)
  // Get unique cluster names from user interests
end for
for j:=1 **to** |K| **do**
  for k:= 1 **to** |FI| **do**
  if $c_{name\,j} \in U_k ≔ \{i_1,\ldots,i_c\}$ **then**
  // if the cluster name is in users interests
   $C_j := C_j \cup \{u_k\}$ // Add the user to the cluster j
  end if
end for
S := slopeOne(selected_item);
for i:= |S| **downto** 1 **do**
  CB := findBestCluster(s_i,C)
  CB = CB ∪ { s_i } // append the best matching cluster with book si
  S = S \ { s_i }; // remove element si from the set S
end for

According to formula :

function findBestCluster(book)
  for i:= 1 **to** |K| **do** // for every cluster
   num_users = 0;
   cscore = 0;
   for j:= 1 **to** $|C_i|$ **do** // for every user in cluster
    if has_rated(u_j, book) **then**
     cscore=cscore+rating(u_j,book)
     num_users = num_users +1
    end if
   end for
   clusterScore = cscore / num_users
   if clusterScore > bestScore **then**
    return_custer=C_i
    bestScore=clusterScore
  end for
  return return_cluster
end

Fig.3: The algorithm’s pseudo code

4 Conclusion

In this paper we combined advantages of collaborative filtering and trust based clustering/ friend interest clustering that is based on an assumption (proven in [5,6,7]) that we share common interest with our trusted peers. We combined the models in order to overcome the disadvantages of trust based clustering we can have friends-peers without common interests [2] using the correctness of the collaborative filtering by improving it for user navigation by adding a clustered view over user interests (compare Figure 1 and 2). Also we suggested how to use the Social Network Services like Facebook Connect, OpenSocial, etc. to improve input data quality for the Trust based clustering.

References:

[1] Goldberg, David, David Nichols, Brain M. Oki, Douglas Terry, Using collaborative filtering to weave an information tapestry, *Communications of the ACM 35* (12): pp 61–70. , 1992

[2] John O’Donovan, Brynjar Gretarsson, Svetlin Bostandjiev, Tobias Hollerer Barry Smyth, A Visual Interface for Social Information Filtering, International Conference on Computational Science and Engineering, pp. 74–8, 2009

[3] H.Marmanis, D.Babenko, *Algorithms of the Intelligent Web*, Manning Publications; 2009

[4] Schotter, A., et al, *Advice and Behavior in Intergenerational Ultimatum Games: An Experimental Approach, Department of Economics*, New York University New York, 2001

[5] Cai-Nicolas Ziegler, Jennifer Golbeck, Investigating Correlations of Trust and Interest Similarity - Do Birds of a Feather Really Flock Together?, Journal of Artificial Intelligence Research, 2006.

[6] Jennifer Golbeck, Trust and nuanced profile similarity in online social network, *ACM Transactions on the Web, in press communities, In Proceedings of the Second International Conference on Trust Management*, 2004.

[7] Cai-Nicolas Ziegler, Georg Lausen, Analyzing correlation between trust and user similarity in online Communities, *Proceedings of Second International Conference on Trust Management*, 2004

[8] Tom DuBois, Jennifer Golbeck, John Kleint, Aravind Srinivasan, Improving Recommendation Accuracy by Clustering Social Networks with Trust, *CEUR-WS*, Vol-532

[9] Daniel Lemire, Anna Maclachlan, Slope One Predictors for Online Rating-Based Collaborative Filtering, *SIAM Data Mining (SDM'05)*, Newport Beach, California, April 21-23, 2005