Recommender System of Vijjana, a Pragmatic Model for Collaborative, Self-organizing, Domain Centric Knowledge Networks

Swetha Koduri

Problem Report submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements for the degree of
Master of Science in Electrical Engineering
Approved by
Dr. Yenumula V. Reddy, Ph.D., Chair
Dr. James D Mooney, Ph.D.
Dr. Sumitra Reddy, Ph.D.

Lane Department of Computer Science and Electrical Engineering Morgantown, West Virginia
2010

Keywords: Vijjana, semantic web, recommender systems, Item based collaborative filtering, collaborative filtering
ABSTRACT

Recommender System of Vijjana, a Pragmatic Model for Collaborative, Self-organizing, Domain Centric Knowledge Networks

Swetha Koduri

In today's world, the internet has become an unconcealed and an obvious application that helps human to acquire, study and learn the knowledge that is fast developing. This has become the huge and easy medium for the exchange of knowledge connecting various unconnected parts of the world. To aid this, the search engines play a prominent role. The existing search engines like Google, Bing, Yahoo etc have made 'search' very easy without which all the existing knowledge would have been inaccessible. Though the existing search engines help to sift through the existing knowledge base, it mostly results in irrelevant information as search results. The semantic web proves a better solution in this regards. Semantic web can be visualized as a well-organized, classified and efficient representation of the available data on the web. Vijjana is such a semantic web based application. It can be described as a Pragmatic Model for Collaborative, Self-organizing, Domain Centric Knowledge Networks. Unlike a key word based search, Vijjana searches the web semantically to achieve the most appropriate results.

In this report, recommender systems that aid Vijjana to be a more sophisticated search engine are discussed. The recommender systems are a personalized information filters that have been developed to identify a particular set of items basing on his history. These recommender systems have gained increased importance in many e-commerce systems. Many of the existing e-commerce systems have been implementing the user based collaborative filtering algorithm to discover the personalized items.
But due to the multiplying growth of users and computational complexity, the user based collaborative filtering systems failed to provide with appropriate recommendations. The item based collaborative filtering systems proved providing more befitting results. This report is more based on development, implementation and working of the item based collaborative filtering techniques in Vijjana. This item based collaborative filtering adds personalized recommendation to each user based on a collaborative methodology which produces recommendations based on the history with comparison to ratings given by other users.
Acknowledgments

I gratefully acknowledge my Committee Chair, Dr. Yenumula V. Reddy for his supervision, constant support, and proper guidance throughout the project. This project is originated from his ideas and evolved from the insights and suggestions of Vijjana team.

I would like to extend my thanks to Dr. James Mooney and Dr. Sumitra Reddy for serving on the committee and supporting the project with their suggestions and encouragement.

I would like to thank all my friends Luyi Wang, Sravanthi Myneni, Venkat Marepally for their valuable guidance and advices they have given me. I would also like to thank Anjaiah Katepalli, Vijaya Katepalli, Sundeep Katepalli, Sandhya Katepalli, Shiva Katepalli, Nanda Kishore Katepalli, Gopinath Reddy Desam, Hyma Yalamanchili, Anusha Gangadi, Praveen Kumar Reddy Godala, Vimala Goli, Rajesh Makineni for always being supportive during various phases of my life.

I am deeply thankful to my beloved parents Mr. Venkatappaiah Koduri and Mrs. Annapurna Devi Koduri for their encouragement and support in all the things I do.


**CONTENTS**

Abstract ................................................................................................................................. ii

Acknowledgements ................................................................................................................ iv

List of Figures .......................................................................................................................... viii

**Chapter 1. Introduction** .......................................................................................................... 1

1.1 Background ....................................................................................................................... 1

1.2 What is Vijjana? ............................................................................................................. 3

**Chapter 2. Problem Statement** .......................................................................................... 5

**Chapter 3. Vijjana Architecture** ........................................................................................ 6

3.1 Vijjana Agents .................................................................................................................. 6

3.2 Vijjana agents and functionalities ...................................................................................... 7

3.2.1 Taxonomy and the Semantic Net of Knowledge (T and R)........................................... 7

3.2.2 Discovery Agent (dA) .................................................................................................. 8

3.2.3 Organizing agent (oA) .............................................................................................. 9

3.2.4 Consistency/Completeness agent (cA) ....................................................................... 10

3.2.5 Search Agent (sA) ..................................................................................................... 11

3.2.6 Rating Agent (rA) ..................................................................................................... 11

3.2.7 Visualization Agent (vA) .......................................................................................... 12

3.2.8 Collaborative Filtering Agent (cfA) .......................................................................... 13

3.3 Vijjana Framework Architecture ....................................................................................... 14

3.4 Vijjana Client Architecture ............................................................................................. 15

**Chapter 4. History** .............................................................................................................. 16

4.1 History of Recommender Systems ..................................................................................... 16
4.2 History of Recommender System Technologies ........................................... 19
  4.2.1 Bayesian networks ................................................................................ 19
  4.2.2 Clustering ................................................................. 20
  4.2.3 Horting ................................................................. 20

Chapter 5. Recommender System Technology .................................................. 21
  5.1 What is a Recommender system? ................................................................. 21
  5.2 Definition of Collaborative filtering .............................................................. 22
  5.3 Collaborative filtering techniques ................................................................. 23
  5.4 Challenges to existing collaborative filtering technique .. ......................... 24

Chapter 6. Item-based collaborative filtering Algorithm .................................. 25
  6.1 Similarity Calculation ................................................................. 25
    6.1.1 Cosine based similarity ......................................................................... 27
    6.1.2 Correlation based similarity ................................................................. 27
    6.1.3 Adjusted cosine based similarity ............................................................ 28
  6.2 Prediction Generation ................................................................. 29

Chapter 7. Experiments to find best fit schema ................................................ 30
  7.1 Data set ................................................................. 30
  7.2 Platform ................................................................. 31
  7.3 Evaluation metric ................................................................. 31
  7.4 Experimental evaluations ................................................................. 31
    7.4.1 For Similarity algorithm ................................................................. 31
    7.4.2 For User based/Item based collaborative filtering techniques ............. 32
Chapter 8. Implementation ..................................................................................................................33
  8.1 User Interface .......................................................................................................................... 33
  8.2 Design ....................................................................................................................................... 34
  8.3 Real time constraints & performance oriented implementation ........................................... 37
    8.3.1 Real time constraints ........................................................................................................... 37
      8.3.1.1 Execution time .................................................................................................................. 37
      8.3.1.2 Large databases ............................................................................................................. 38
      8.3.1.3 Availability of Resources ............................................................................................... 38
      8.3.1.4 Utilization of resources ................................................................................................ 38
    8.3.2 Real time Implementation .................................................................................................. 39

Chapter 9. Future Work & Conclusion ................................................................................................. 41
  9.1 Future Work ............................................................................................................................... 41
  9.2 Conclusion ................................................................................................................................. 42

References ........................................................................................................................................... 43

Appendix: Presentation Slides ........................................................................................................... 46
LIST OF FIGURES

Figure 1. Semantic net of Vijjana Visualization Views (sample) ........................................... 7
Figure 2. Proposed view of Vijjana's Knowledge Base Web Interface to add a Jan ..................... 9
Figure 3. A proposed View of the User’s Vijjana Knowledge Network

after a new JAN was added ....................................................................................................... 10
Figure 4. Proposed Search Agent of Vijjana ........................................................................ 11
Figure 5. Proposed rating agent of Vijjana ........................................................................... 12
Figure 6. Radial view of visualization ................................................................................... 13
Figure 7. Vijjana Framework Architecture ............................................................................ 14
Figure 8. Vijjana client architecture ...................................................................................... 15
Figure 9. Collaborative filtering process ............................................................................... 24
Figure 10. Item based similarity computation process .......................................................... 26
Figure 11. Experimental results of various similarity computation measurements ............... 32
Figure 12. Experimental results for evaluation of better collaborative filtering algorithm ....... 33
Figure 13. Block Diagram ..................................................................................................... 35
Figure 14. Screenshot for Recommendations for user id =1 ................................................. 36
Figure 15. Screenshot for Recommendations for user id =3 .................................................. 37
Figure 16(i). Illustration of real time working of recommender system: Item association ............ 40
Figure 16(ii). Illustration of real time working of recommender system: Prediction generation... 41
1. Introduction

1.1 Background:

In the world of drastic developments, it’s a herculean task to keep oneself updated of required knowledge, with the never ending accelerated additions to knowledge base. But, the internet with its search engines makes it possible to a certain extent. The internet makes the science accessible to unconnected people in different parts of the world. The search engine accomplishes the task of obtaining the information a little easy by sifting through the huge knowledge base, of exabytes measure, and providing the user with desired data. The current day search engines like Google, Yahoo, Bing etc use various technological search strategies to screen through the loads of data. The existing search engines make use computer robot programs called spiders. These spiders crawl through the large databases by utilizing the links in the pages that are already in the database. After the spiders find the pages, the indexer indexes those pages. These indexes are useful in identifying text, links etc and saves in a search engine database. Whenever a query is generated, the search engine actually searches a copy of the data that has been created a while ago. Also, the search is merely based on text. This kind of just text based search leads to a irrelevant information that may misdirect the user and voids the valuable time of the user.

Accounting to all these discrepancies, the semantic web has proven a better solution. The term web has been coined by Tim Berners- Lee in 1999, who was director of the World Wide Web consortium (W3C). Semantic means 'connected with meaning'. He modeled web as the universal medium for data, information and knowledge exchange. He envisioned semantic web as the technology that becomes capable of analyzing all the data on the web. This can also be described as
intelligent web that can accomplish the assigned tasks without human direction. As such, the web alone, would be proficient in achieving the desired results. The semantic web, in short means elaborating things in a way that the computer applications can understand. This model of a web system is difficult to take shape as it involves a huge labor which changes the entire structure of the web. And this idea lacks real time implementation.
1.2. What is Vijjana?

Considering all the above mentioned facts, Vijjana might prove to be an attainable and implementable achievement that might change the face of the knowledge base that is available on the internet. Vijjana in Sanskrit stands as a synonym for the act of distinguishing or discerning, understanding, comprehending, recognizing and also intelligence, knowledge, skill, proficiency. Summarizing the above mentioned, the Vijjana can be said to be knowledge obtained by analysis and classification.

Vijjana can be actually modeled as an adaptation of semantic web with some additional procedures to obtain a wholesome functionality. Vijjana is semantic in that it makes an attempt to help the web understand the connections among the various data that is available. As it is said that 'Drop by drop each drop of water makes a big ocean', Vijjana utilizes the information input by each user in addition to its database, to make up a huge classified data. Vijjana takes into account the information each user associates with every article and uses that particular information in categorizing the knowledge. The key word generation algorithm designed for Vijjana reads through all the content of an article and generates appropriate key words that convey the real essence of the particular article. As such, Vijjana sets a series of sieves to the information available to let the end user obtain the required data. Thus Vijjana aims at transforming the web to semantically connected system. The Vijjana system combines the concepts of social bookmarking and semantic web. The system takes into account all the bookmarked pages and each one is termed as 'Jan'. So, the Vijjana system becomes collaborative in nature that it considers the information marked by the user in determining the relation among the various articles in database. Also it uses the collaborative nature
in making personalized recommendations to the users basing on their rating profile in relation with the ratings given by other users to various other items. These Jans are together stored in a database labeled as knowledge network. Thus, Vijjana proves to a pragmatic model for collaborative, self-organizing, domain centric knowledge networks.
2. Problem Statement

Even with the modified semantic search engine, it is a huge problem for an average user to study every piece of information available. Each day, there are a lot of inventions, improvements taking place in every knowledge domain which adds enormous data to the existing knowledge base. This might be termed as ‘information overload’. It is quite impossible for a user to read through all the material that has been updated. The solution is a recommender system that aids the user to resolve these issues to a certain extent. The ‘recommender system’ provides the user with good articles in their area of interest. The term ‘good articles’ refers to those which are rated collaboratively in relation to the history of this particular user and ratings of other users. The recommender system thus makes it easy for the users to sift through the available knowledge base and find the appropriate information by making personalized recommendations.
3. Vijjana Architecture

3.1 Vijjana Agents:

The Vijjana was modeled using the following elements.

Vijjana-“X” = { J, T, R, dA, oA, cA, vA, sA, rA, cfA}

where

‘X’ = the domain name

‘J’ = the collection of Jans in the Vijjana-X

‘T’ = the Taxonomy used for classification of Jans

‘R’ = the domain specific relations

‘dA’ = the discovery agent which discovers relevant Jans

‘oA’ = the organizing agent which interlinks the Jans based on R

‘cA’ = the consistency/completeness agent

‘vA’ = the visualization agent

‘sA’ = the search agent

‘rA’ = the rating agent

'cfA' = the collaborative filtering agent

These agents are explained below.
3.2 Vijjana agents and functionalities:

3.2.1 Taxonomy and the Semantic Net of Knowledge (T and R)

Vijjana means classified knowledge. So a classified knowledge base is the primary constituent in building Vijjana. Any knowledge domain is termed as 'Jan'. As such the knowledge base can be described as a collection of Jans. These Jans are to be classified and interlinked with one another in order to create a semantic net. So this requires that a taxonomy appropriate for the domain has to be established.

Figure 1. Semantic net of Vijjana Visualization Views (sample)[6]
The same knowledge can be associated with a variety of other taxonomies, Vijjana model has to be organized such that it can be related to any other taxonomy and appropriate semantic data by supposing them as parameters that could be altered as the knowledge base increases. This permits Vijjana builder to import and export the knowledge when exporter and importer use the same taxonomy.

The Figure 1. depicts the representation of Vijjana taxonomy. In this knowledge base, as new Jans are added it expands and contracts when dead links are removed.

3.2.2 Discovery Agent (dA):

The discovery agent is responsible for collection of different Jans form the when and adding to Vijjana knowledge base. The collection of the Jans could be manual or automated. In these early stages of Vijjana development, manual version of the agent is implemented. The manual system allows the users to mark up a domain as Jan to the appropriate category in the taxonomy. The 'mark up' implies associating the new knowledge to the actual node ( in the above figure) representing the domain in Vijjana's knowledge base. In future, the automated implementation might be utilized which is implemented by subscribing to the RSS feeds of interested domains which updates the respective Jans in knowledge base.

Other collaborators like email can also be used to for parsing and addition of Jans by implementing a predefined syntax to these emails which results in automated updating of knowledge base.
Figure 2. Proposed view of Vijjana's Knowledge Base Web Interface to add a Jan [7]

3.2.3 Organizing agent (oA):

The Jans can be added to Vijjana's knowledge base by several ways: by collaborative agent or by RSS feed or by markup button installed on the browser or by Vijjana's client interface. After the addition of these Jans to the knowledge base, these have to be organized in a correct fashion else this results in dysfunctional additions. To organize such data, the oA examines the meta data of each Jan and adds that particular knowledge to the related domain in Vijjana's knowledge base. The oA would first determine the working status of the link, whether its broken link or a working link. Then it also investigates about the source of the link if it is reliable or unreliable. After this inspection, the oA would add the particular Jan evaluating its content and meta data. As such, it organizes the new data being added to the knowledge base. In addition to organizing, the oA also maintains the attributes in regards to adding information such as date, time, editor etc. This helps in keeping track of
modifications made to the specific Jan. The oA also sends out updates to the subscribed users regarding changes made to the Jan.

Figure 3. A proposed View of the User’s Vijjana Knowledge Network after a new JAN was added.

3.2.4 Consistency/Completeness agent (cA):

The cA conducts a periodic check about the status of the links made available in all the Jans. The cA reads all the relational links connected to a particular Jan and examines the working status, if they are broken or working links. The cA also checks for the incomplete relational links associated to a particular Jan, such incomplete relations are reported in the agenda file which allows the users to fix manually. Such incomplete nodes would also be color coded for easy identification and visualization purposes.
3.2.5 Search Agent (sA):

The search agent helps in sifting through huge databases and to fetch the relevant information. In Vijjana, the search agent implements varied search mechanisms that include keyword based search or an advanced search. The search agent would get the appropriate information only if all the semantic links of a Jan are completely filled.

![Proposed Search Agent of Vijjana](image)

3.2.6 Rating Agent (rA):

The rating agent (rA) aids in marking good articles in the whole knowledge base of Vijjana. The rating agent allows the users to rate the articles as per their own constraints. A simple model of rating user interface is shown below. This rating agent is helpful in making Vijjana collaborative. This
rating scale is utilized in constructing a recommender system for the Vijjana system. The recommender system aids in users discovering good articles which saves their time in browsing.

![Proposed rating agent of Vijjana](image)

Figure 5. Proposed rating agent of Vijjana

This rating agent could also be added with more specifics to enable the rating speak more regarding the particular article. Generally, users might rate the article depending on various factors like content, presentation, understandability etc of the article. So if more rating scales are available, the final user could reach his apt data more easily by using some relevant specifics as per his requirement.

3.2.7 Visualization Agent (vA):

The visualization agent is another important module which at times sets the social bookmarking websites apart. The vA represents the knowledge with its semantic links in the form of a tree/graph. This helps in easy understanding of the varied dimensions of the search topic available. By clicking on each node of graphical representation, it unfolds many sub divisions present in it
helping the user to get to final destination more promptly. This visualization is implemented with Adobe Flex as its front end.

Figure 6. Radial view of visualization[6]

3.2.8 Collaborative Filtering Agent (cfA):

The collaborative filtering agent aids in making sifting process through the knowledge base easy for the user. These collaborative filtering agents model an algorithm which predicts interesting articles depending on the user's rating profile. The algorithm is based on the rating profile of a
particular user and identifying specific 'good' articles in relation to the ratings given by other users to various other articles. As such, a recommender system is built collaboratively with all the users.

3.3 Vijjana Framework Architecture:

![Figure 7. Vijjana Framework Architecture](image-url)
This figure depicts the architecture of the Vijjana system. On the left side is shown the admin part of the Vijjana model which is responsible for checking the working status of Jans and organizing them depending on the meta data of the link. The collaborative filtering agent attempts at filtering out appropriate recommendations for a particular user. On the right side is shown the user interface, which allows the user to visualize the Vijjana network, import/export the entire knowledge network, add particular Jans to knowledge base, communicate with similar users which is implemented by rating phenomenon. And to the bottom of the figure is shown the discovery agent that adds Jans to Vijjana's knowledge base form the world wide web with aid of discovery agent(dA). On the top are the taxonomies and their relational semantics which help is constructing the semantic net.

3.4 Vijjana Client Architecture:

![Vijjana Client Architecture](image)

Figure 8. Vijjana client architecture figure[10]
The picture shown above depicts the client side architecture model of Vijjana. The user/client can contribute to the existing knowledge base by adding some good articles. This addition could be by email or markup. These emails are then screened by the parser and related Jan is identified. The consistency agent checks for the reliability of the link and passes on to the organizing agent, which inspects if the semantic connection is appropriate or not. Finally after all evaluations and screenings, the Jan is added to appropriate node in the knowledge base.

4. HISTORY

4.1 History of Recommender Systems:

It’s a known fact that people mostly tend to believe in the word-of-mouth. This conception landed in the development of recommender systems which propagate the idea of word-of-mouth on a large scale. The recommender systems spread the suggestions/advices given by an individual to be shared by a group of people who might be interested in similar products. So, recommender systems gained popularity over time. As this implementation is based on a network of users sharing their opinions on several things, such a system is termed as collaborative filtering based recommender systems. This term was first coined by the developers of Tapestry. Tapestry was one of the earliest applications of collaborative filtering based recommender systems. This was developed at the Xerox Palo Alto Research Center. This was first designed to filter the mailing lists which had a very huge volume inflow. In this model, the filtering is designed to be done in conjunction with the other users' reactions which are termed as annotations. These annotations can be utilized by other filters to separate the required mails from the unimportant ones. Tapestry achieved a good extent of success in small groups of people, generally office friends, relatives etc, who generally know each other. But this system failed to produce good recommendations when the size of the group was increasing.
The Lotus Notes mailing system also had an inbuilt collaborative filtering system though it was used more in corporations as groupware.

In 1992, Group lens recommender was developed by Group lens research team at the University of Minnesota. John Riedl and Paul Resnick first began on working on collaborative filtering for Usenet news. This was the first automated collaborative filtering system in which predictions were generated by algorithms based on their previous rating pattern. This recommender would help people find useful articles in a huge collection of available articles. News reader clients display predicted scores and thus make it east for other users to rate articles after they read through them. Rating servers that are termed as Better Bit Bureaus collect the ratings, compute the prediction, display. These predictions were based on heuristic that people who showed agreement in the past would tend to agree in the future. This recommender system gained a lot of prominence as it proved better results than other existing systems, even when implemented on large scale. This recommender system allowed the addition of new users and new data and still produced good recommendations.

In the line of developing collaborative filtering systems, Mosaic was the first graphical web browser developed at the University of Illinois-Champaign that aided in collaboration among users by allowing them to publish notes and comments which is additional information added to each web page.

Helpful Online Music Recommendations (HOMR) is a music recommender system that helped users navigate the music domain to find other albums, bands, artists that a user might be interested
in. In 1995, MIT developed the AI technology to allow music lovers to share information with each other about the bands they liked. In 1995, the company incorporated as Agents Inc.

Ringo was also another such music recommender system which evaluated a list of good music that the user might like by computer analysis. This also gained popularity as a good social information filtering system.

Later on, in 1996 at MIT, Firefly technology was developed. It gathered huge amounts of user preferences and utilized this information to produce recommendations on music and movies. This Firefly technology gained importance as it had advanced features in comparison with the earlier developed methods. It had built in privacy protection and gave users the choice over the amount of personal information that was shared on several websites. This technology was later bought by Microsoft in 1998 and was named Microsoft Passport.

Yahoo! which was first developed by Princeton University students David Filo, Jerry Yang was a web link directory. The classification system utilized both human and machine technology to produce better results. The inputs given by any user are are filtered through a team of librarians to make better information more accessible. Point's Top 5% was another New York City based firm which was the starter to qualitatively rate the websites. Reviewers surfed and critiqued the information input by the consumers to judge for the best websites. Another system called PHOAKS(People Helping One Another Know Stuff) checked through the Usenet groups and would sort out the most posted URLs to a website as a collection based on the number of times they were posted. Fab, developed by Marko Balabanovic was another website recommendation system which allowed users to create content based filters that allowed to compare and select information that was
more useful. Webdoogie was another such technology that helped users to search websites basing on their interests. In 1996, Brewster Kahle and Bruce Gilliat developed Alexa Internet which was a model of web navigation system that utilizes people's preferences of websites to make suggestions to others. Whenever a user browses through a website, the system gives a list of websites that have been browsed by others who have gone through the same page.

4.2 History of Recommender System Technologies:

4.2.1 Bayesian networks:

This one of the primary technologies that was designed to be used in recommender systems. The term 'Bayesian Networks' was first coined by Judea Pearl in 1985. These Bayesian networks construct a probabilistic graphical model which represents conditional dependencies and variables. In recommender system model is based on the results obtained from a training data set. So experiments are conducted on a training data set and the results are built into a tree like structure named decision tree. This decision tree comprises nodes and edges which stand for user information. This model is built off-line from the results obtained and then utilized in making predictions real time. The construction of this decision tree offline might take time in the order of hours or days. The predictions obtained from a recommender system based on Bayesian networks would be very accurate, very fast. The back log of this technology lies in updating the information. The information cannot be updated fast as it involves in the reconstruction of the whole decision tree which may consume lot of time. So this technology is suitable only at such platforms where the user preferences stand unchanged with time or user preferences change slowly with respect to the time needed to build the model. This technology may not be quite apt for the current day interactive e-commerce applications.
4.2.2 Clustering:

The clustering techniques function by constructing clusters of users who hold the same list of preferences. After the clusters are formed, the predictions are generated by averaging opinions of other users in that particular cluster. In these techniques, each user may be part of several clusters. Considering this, while prediction is calculated, final prediction is the average among the clusters the user is a part of and is weighted by the degree of participation in each cluster. The technique of dividing the users into clusters may be one of the deciding factors for efficiency of the whole system as it is the atomic structure. The clustering techniques initially may produce many less personal recommendations and at times produce very less accurate predictions. But with time, formation of clusters would be more well-defined resulting in better recommendations. Also the recommendation would be more accurate as the analysis is to be performed on much smaller groups of users. These clustering techniques might be very useful as a primary step for reducing the size of group of users in nearest neighbor algorithm or in distributing nearest neighbor algorithm computations.

4.2.3 Horting:

This is a graph based technique constructed with nodes and edges. The nodes represent the users in the network and the edges connecting them stand a measure for similarity between those two users. The predictions are generated by traversing through the graph to the nearest users (in terms of similarity) and combining their opinions. Horting at times produces better recommendations than nearest neighbor algorithm. But its quite different in the technique that is implemented in nearest neighbor algorithm as horting also traverses through the users who have not rated the item to which
the prediction is to be generated, but it just traverses because of greater degree of similarity between those users.

These techniques discussed above gained prominence in the early stages. Some of them produced good results than others but extensive usage of these techniques revealed their limitations. The limitations included sparsity in data set, high dimensionality, huge sizes of databases etc. The item based recommendation algorithms answered these barriers to a great extent.

5. Recommender System Technology

5.1 What is a Recommender system?

A recommender system is a technique that gives suggestions to the user based on the article he is interested in. These recommender systems effectively reduce the time spent and increase accuracy in finding the relevant information. These suggestions may be top N rated articles in a particular genre/most visited etc. Many technologies have been developed to propose a best working recommender system, which makes most reliable recommendations.

The current day recommender systems are based on the theory of 'social information filtering'. This is based on the principle that similar items interest similar people.

The earlier recommender systems used made recommendations on a very generalized basis for all the users. A screening system with a mere text based filtering, which is not adequately sufficient. It requires a more enhanced filtration system which provides more appropriate content to the user. This is termed as content based filtering technique. These recommendations may be based on the top 10 well rated or most visited articles etc. Such recommendations may not be apt for all users. These
recommendations are based on the ‘search term’ input by the user and the average rating provided by all the users as a whole.

**Social Information filtering exploits similarities between the tastes of different users to recommend (or advise against) items. It relies on the fact that people's tastes are not randomly distributed: there are general trends and patterns within the taste of a person and as well as between groups of people. Social Information filtering automates a process of "word-of-mouth" recommendations. A significant difference is that instead of having to ask a couple friends about a few items, a social information filtering system can consider thousands of other people, and consider thousands of different items, all happening autonomously and automatically (Shardanand and Maes, 1995).**

Information Filtering has gained importance as it was the prime development towards personalized recommendations. The information filtering is mainly dependent on the syntactic and semantic information of each article to assess the interest of each user. The information filtering estimates interest of a particular user on articles based on his history thus making personalized recommendations.

### 5.2 Definition of Collaborative filtering:

The collaborative filtering technique, gained importance in e-commerce systems as it was generating more accurate personalized recommendations over the time. The collaborative filtering technique is based on the fact that word-of-mouth is much relied upon. The collaborative filtering technique utilizes the choices of other users in the system to predict the likeliness of a user to that particular article. The system constructs a ‘**virtual like user**’ for every individual consumer. The virtually developed ‘like-user’ develops a tendency similar to the user based on the history of the end
user with respect to the ratings given by other users too. Thus, basing on the rating pattern of the user, for various articles, in relation with other user ratings the collaborative filtering system evaluates similarity among users/items. Depending on these similarities, a tendency is built for the particular user. Appropriate recommendations are thus made in a collaborative fashion. Collaborative filtering and information filtering both make personalized recommendations but CF searches for such articles that are rated well and liked by the user rather than recommending him the articles which the system estimates will be of interest to the user.

5.3 Collaborative filtering techniques:

In today’s fast developing world, where each individual is posed with the puzzle of ‘information overload’, collaborative filtering technique proves a promising solution. This system sieves all the information available with the collaborative aid of other users to provide apt and useful information to the user.

The conventional collaborative system that’s being implemented is based on developing a group of users. This might also be termed as user-based collaborative filtering system. In this kind of filtering mechanism, the users are grouped into several classes basing on their rating history. Each user in the system is assigned to that class of users to which his rating/ item history can be almost matched. As such, each class gets added with more users every day. The users in that class will be suggested with those items that the other individual users have rated as ‘good’. Thus, the recommendations made are the opinions of other users on various other products. The recommendations could be generated like the top-N list for every class in the system. This
collaborative filtering system proved beneficial in the early stages of development when such systems were only utilized on a small group basis. This system of classification becomes unmanageable when the users and information in the knowledge base multiplies day by day.

![Collaborative filtering process](image)

Figure 9. Collaborative filtering process[^3]

**5.4 Challenges to existing collaborative filtering technique:**

Technically, the user based collaborative system can be spotted with some major problems. These conventional filtering systems’ algorithms lack to hold the level of performance when the size of users is increasing in a tremendous fashion. This problem of less scalability can be marked as a hindrance. So, the system requires new algorithms that are capable of searching through tons of information and also which can make good recommendations to ever increasing number of users. Another significant challenge is the quality of recommendations that are made by the collaborative filtering system. The appropriateness of the suggestions to a user also has to be high. But, these two problems stand contrary to one another. If the system spends more time on each user, it enhances the quality of recommendations made but diminishes the scalability of the system as a whole and vice
versa. So the user based collaborative filtering system failed to suit the fast developing commercial environments.

To encounter these problems of the day, a more enhanced filtering system was developed. This is named item based collaborative filtering system. This collaborative filtering technique is based on the calculation of similarity of items and users unlike the former system. The recommendations are made on the basis of ratings a particular user has given.

6. Item-based collaborative filtering Algorithm

In item based CF, the appropriate item suggestions are computed basing on that particular user's rating profile in relation with the item ratings given by other users. A fundamental difference between item based CF and the user based CF is that the computations in user based system are based on similar user profiles (along the rows), on the contrary, in item based CF the computations are based on the ratings given to those particular items (along the columns). The calculation takes place in two separate steps namely

(i) similarity calculation

(ii) prediction generation

6.1 Similarity Calculation:

In the similarity calculation, likelihood among the different items in the database is computed. This can be developed by various statistical methods which result in a most appropriate conclusion. This proves the vital step in deducing conclusions for the recommendations that are to be produced. As
this is the atomic unit involved producing a recommendation, it stands a determining factor for the accuracy of the whole system.

Figure 10. Item based similarity computation process\textsuperscript{[3]}

The statistical methods based similarity calculations are elaborated below.
6.1.1 Cosine based similarity:

Consider the user item data as a nxm matrix. Each item can be visualized as a vector in the space of customers. The angle between these vectors determines the likeliness/similarity between those particular items. Mathematically, the angle between two vectors can be obtained by dot product. Thus, the similarity between two items i, j is defined by the cosine of the angle between the n dimensional vectors corresponding to $i^{th}$ and $j^{th}$ columns of user item data matrix obtained by the dot product of those two vectors.

Supposing $a, b$ to be two items (represented as in vector form). So mathematically, the similarity can be expressed as

$$sim(i, j) = \cos(\vec{i}, \vec{j}) = \frac{\vec{i} \cdot \vec{j}}{||\vec{i}|| \times ||\vec{j}||}$$

$sim(i,j)$ is the similarity between $i^{th}$ and $j^{th}$ items

6.1.2. Correlation based similarity:

This method is based on statistical variable named the Pearson-r correlation. The Pearson's correlation coefficient depicts the degree of relation between the two different objects. It helps to reveal the measure of linear relationship between the two items. The Pearson-r coefficient stands as a scale for the similarity. This Pearson-r coefficient varies between -1 and +1. A '-1' affirms a 'perfect negative relation' 'inverse relation' or between the two items. A '0' affirms a 'no relation' between the two items where as a measurement of '1' reaffirms a 'positive linear relationship' between the items. A variation between '0 and 1' scales from a 0% to 100% similarity among the articles.
Mathematically, it can be written as

\[
sim(i, j) = \frac{\sum_{u \in U} (R_{u,i} - \bar{R}_i)(R_{u,j} - \bar{R}_j)}{\sqrt{\sum_{u \in U} (R_{u,i} - \bar{R}_i)^2} \sqrt{\sum_{u \in U} (R_{u,j} - \bar{R}_j)^2}}.
\]

\( \sim(i, j) \) is the similarity between \( i^{th} \) and \( j^{th} \) items

\( R_{u,i} \) is the rating given by user \( u \) for the \( i^{th} \) item

\( \bar{R}_i \) is the mean rating for the \( i^{th} \) item.

### 6.1.3 Adjusted cosine based similarity:

The adjusted cosine based similarity enhances the result obtained in cosine based similarity by the inserting the mean of the user term for mean of the item. This replacement produces a better result as it offsets for the differences in rating measure of each user. As such, this method can be more accurate in measuring the similarities than the above mentioned methods.
\[ sim(i, j) = \frac{\sum_{u \in U} (R_{u,i} - \bar{R}_u)(R_{u,j} - \bar{R}_u)}{\sqrt{\sum_{u \in U} (R_{u,i} - \bar{R}_u)^2} \sqrt{\sum_{u \in U} (R_{u,j} - \bar{R}_u)^2}}. \]

Formula detail: In this method, \(sim(i,j)\) is the similarity between \(i^{th}\) and \(j^{th}\) items

- \(R_{u,i}\) is the rating given by user \(u\) for the \(i^{th}\) item
- \(\bar{R}_u\) is the mean rating for the user \(u\).

This is the similarity calculation that is employed to develop the current module of recommender system. This particular similarity calculation proves more appropriate in calculating relation between the items.

\section*{6.2 Prediction Generation:}

This is the module for forecasting the interest of a particular user basing on the ratings of all the users in the system. Taking the similarities generated and ratings given by the user, the prediction is concluded.

The similarities thus calculated are used in evaluating the prediction of an item respective to a particular user. The prediction represents the likeliness of the particular user towards a specific item. This can be measured by taking a weighted average of the ratings of the particular user on the basis of the similarities calculated. The weighted average stands a good measure as it reflects the opinion of a particular user's ratings rather than the user based method, which disregards the specific user's ratings. This prediction calculation stands a better scale of user's preferences as all the ratings given by that
user are taken into consideration. Finally, the weighted sum is scaled by the sum of similarities to assure the prediction is present in the predefined range.

\[ P_{u,i} = \frac{\sum_{\text{all similar items, } N} (s_{i,N} \times R_{u,N})}{\sum_{\text{all similar items, } N} |s_{i,N}|} \]

Formula detail: \( P_{u,i} \) is the prediction generated for user \( u \) for \( i^{th} \) item

- \( s_{i,N} \) is the similarity between \( i^{th} \) item and \( N^{th} \) item
- \( R_{u,N} \) is the rating given by user \( u \) for \( N^{th} \) item

7. Experiments to find best fit schema

This implementation of recommender system uses item based collaborative filtering technique which applies the adjusted cosine based similarity calculation. These methods have been chosen above others for their accuracy and working. The experimental evaluation supports these facts. These experiments were published in the publication 'Item-based Collaborative Filtering Recommendation Algorithms' by Group lens research group/Army HPC Research Center.\[3\]

7.1 Data set:

The data used is from Movie lens recommender system. The users rate the movies and get recommendations for movies depending on their rating history. The Movie lens was launched in 1997. When, this study was conducted, it had about 43000 users and 3500+ movies in the database. From this huge database, 100000 ratings were considered for this study which constituted to 943 users, 1682 movies. The users who have rated 20 or movies have only been selected for the experiment. This study
data was classified into training set, which had 80% of the information and test set which had 20% of information.

**7.2 Platform:**

The experiments implemented by Group lens research group were conducted on a linux based PC with Intel Pentium III processor having a speed of 600 MHz and 2 GB RAM. The methods were in C, compiled using optimization flag -O6.

**7.3 Evaluation metric :**

**MAE :**

This is a statistical accuracy metric to assess the accuracy of the recommendations generated by the system. MAE means Mean Absolute Error which is more if the predicted value is more than the actual value. It is computed by taking mean of the absolute values of deviations from original values. It is a measure of the deviations of recommendation from the true values assigned by the user.

\[
MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N}
\]

\[|p_i - q_i|\]

is the difference between the experimentally predicted value and original rating given by the user. So, the methodology that results in a less MAE is the recommended scheme.

**7.4 Experimental evaluations :**

**7.4.1 For Similarity algorithm**

In conducting the evaluation of the accurate similarity computation algorithm, two types of results are to be taken into account. They are quality results and performance results. Some parameters
like neighborhood size, training/test data set ratio, etc. are to be described before the conduct of the experiment.

The experiment is based on evaluating the most accurate similarity algorithm of the cosine similarity, Pearson correlation based similarity and adjusted cosine similarity. For each similarity algorithm, an algorithm is implemented to compute the neighborhood and generated the final prediction by implementing a weighted sum algorithm. These calculations were performed on the training data sets and were evaluated with respect to the test data set. The following figure depicts the MAE produced by three different algorithms. This concludes that Adjusted Cosine similarity provides better results than other similarity algorithm.

![Figure 11.Experimental results of various similarity computation measurements](image)

7.4.2 For User based/Item based collaborative filtering techniques:

The optimal values of the parameters like neighborhood size, training/test ratio are evaluated. The experiment is performed using both the techniques user based and item based techniques. Both the procedures are collaborative in nature. The figure shows quality of prediction produced by all methods.
The figures conclude that item based collaborative filtering technique gives good results at all sparsity levels in comparison to user based technique.

Figure 12. Experimental results for evaluation of better collaborative filtering algorithm\(^3\)

8. Implementation

8.1 User Interface

The user interface in the collaborative filtering based recommender system is primarily the rating scale which helps taking the input from the users for the articles. The rating scale is measured in the range of 1-10, 1 marking towards the bad side and 10 marking towards the good side of rating. This rating stands as the scale of measurement for a user re-recommending that particular article.
8.2 Design

In theory, there could be a logical solution to the problem of recommender systems that provides with, say 90%, precise recommendations. The user based recommender system would entirely be a solution for such systems, but with a little database and small group of people. This methodology lacks a real time performance for a large database and a huge network of users. The item based recommender system proves a better solution in real time large e-commerce sites. The implementation of item based/model based recommender system starts with the correlation-based similarity computation which is performed on the whole of item space. The similarity computation starts off with evaluation of similarities among the items in the database and construction of item similarity matrix. In general, the number of items is mostly static, in comparison with the number of users which keeps increasing day by day. As such, the pre-computation of item-item similarities proves an easy real time implementation. For the calculation of the similarities, each item is associated with every other item in the database. And the "Identifier" runs through the rating data of both the particular articles, marking all the instances with both the items rated. Now the identifier holds the "bundle of users" who have rated both the respective items. The "Mean Fetcher" collects the mean for those respective users $R^\text{u}_\cdot$ (mean of the user) to account for any discrepancies in the rating scale. These "collection of users" and the "mean data" serve as inputs to the "Similarity calculator". The similarity calculator determines the "Item item similarity matrix", which depicts the scale of likeliness among the various articles in the database. As such the similarities are pre-measured.
Figure 13. BLOCK DIAGRAM

Identifier ➔

$\begin{array}{cccccc}
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\end{array}$ ➔

Mean Calculator

Mean Fetcher

Collection of users ➔

Mean data ➔

Similarity calculator

$\begin{array}{cccccc}
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\end{array}$

Items similarity matrix ➔

$\begin{array}{cccccc}
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\end{array}$

$\begin{array}{cccccc}
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\ast & \ast & \ast & \ast & \ast \\
\end{array}$

Prediction generator ➔

Prediction ➔
The prediction is estimated basing on the ratings, of say the \( n^{\text{th}} \) user, and its corresponding similarities. The similarities already calculated are input to the “Prediction Generator” that generates the final prediction. The formula is weighted sum of the items the \( n^{\text{th}} \) user has rated and the corresponding similarities with the item the prediction has to be made. Say the prediction has to be generated for the \( 100^{\text{th}} \) item, the prediction would be the weighted sum of the ratings of the already rated items and their similarity with the \( 100^{\text{th}} \) item. Then weighted sum is then scaled by the sum of item similarities to generate a value within the range, which is the final measure of recommendation generated.

The block diagram shown above depicts the control flow of data from one stage to other.

Shown below are the screen shots of the recommender system:

![Screenshot](image)

Figure 14. Screenshot for Recommendations for user id = 1
8.3 Real time constraints & performance oriented implementation

Any proposal that is sound by theory may not always be possible for a real time implementation. It surely has to be slightly modified or must be completely remodeled for a working design to take shape. In software development there are many such constraints that hinder the exact theoretical replica of the product.

8.3.1 Real time constraints:

8.3.1.1 Execution time is one of those constraints that restricts such exact theoretical development, particularly in dynamically changing e-commerce web sites. Dynamically changing implies that those
websites keep altering with respect to time which might be a result of continuous addition of information to the existing database. Such additions result in changing conditions/circumstances that might change the entire working of the model.

8.3.1.2 Large databases, which is main component of existing web based systems. The databases of the current day are mostly huge due to the ever growing 'accessible' knowledge base. The boundless growth in the users of the internet is also aiding for the huge databases that act as a hindrance for theoretical model development.

8.3.1.3 Availability of Resources, which imply the ability of a system to run the whole existing designed model. This sometimes is the key reason for any modification applied to the logical model that has been constructed. The resources have to be utilized very efficiently for a better performance in a long run. The resources stand the reason for limitation if the available resources do not quite accommodate the required means to execute that particular assigned job. In short, if there are no available resources to fit the schema, the structure of the whole project has to be remade.

8.3.1.4 Utilization of resources: The resources that are accessible have to be handled in a competent way for a longer life and a better performance. If the resources are over loaded heedlessly, the system would sound perfect for a very short period. But in a long standing, a strict theoretical implementation without any consideration to the daily constraints would be in vain.

In our implementation of user based collaborative filtering system, there are thousands of computations that should be carried per second. As such, there are several constraints that are to be
reasoned. Accounting for all such discrepancies, some modifications have been made to the theoretical illustration of the procedure.

### 8.3.2 Realtime Implementation:

As illustrated, the Identifier marks all the users in the database who have rated both the items and gathers the collection of users. The Mean fetcher accumulates the mean values of those specific users. These are provided as inputs for the similarity calculator which outputs the similarity between the two marked items. In the real time implementation, the item-item similarity matrix is not constructed; instead a similarity tree is built.

The tree can be imagined to have loads of branches that are comparable to the tons of items in the database. Each branch is further connected to sub branches. These sub branches resemble the most similar items to that particular item. When the similarities are being calculated, there is a temporary array of similarities that is stored relating to a specific item. This array is sorted out by the order of similarities and each item is associated with a bunch of most similar items in the database. The similarity tree is thus arranged with each item attached with a group of most similar items, say 50. This calculation is performed once a day during the maintenance period. This module of the design is modified to avoid the thousands of iterations. The similarities of all items changes even when a single item is rated by one user. Though not a remarkable change in the similarity, that may be in the order of one tenths or one hundredths, the iterations are required to be redone. To avoid overloading of resources, the similarity calculation is performed during the maintenance period. And the pre calculation of similarities and association of an item with its similar items also prevents the numerous
arithmetic that is to be initiated every time the user logs in or whenever a new entry is made into the database.

Now, each item is chained with a group of most similar items. When a prediction is to be generated, for an item say for user 'u', the prediction generator looks up for the group of similar items associated with that particular item. It then computes a weighted sum of the similarities of those items with the respective ratings made by user u for items in the group. The final weighted sum is scaled by the sum of similarities to obtain a prediction that stands in the scale.

After Item-item similarity calculation:

We have each item associated with a group of items, like

![Diagram of Item Association](image)

Figure 16(i). Illustration of real time working of recommender system: Item association
When a prediction for say, 6th user for 6th item is to be determined,

![Diagram of prediction generation process]

User's rating profile

Weighted Sum

Prediction

Figure 16(ii). Illustration of real time working of recommender system: Prediction generation

9. Future Work & Conclusion

9.1 Future Work

Vijjana can be developed as a tool with various functionalities with respect to the users. The users could be built a personal space that aids them to design their personal space. The Jans should be personal to the user or could be shared depending on the user. The taxonomy has to be built perfectly well for a better model of Vijjana. The automated taxonomy algorithm has to accurate enough to place each Jan is appropriate genres.

In the context of future work for recommender systems, some improvisations can be made to the suggested model. In this model, the ratings act as the scale for the interest shown by the users. But, algorithms can also be modeled considering the time spent by each user on an article as a measure for their interest, which can be termed as 'implicit rating'. This reduces the necessity for the user to explicitly rate the items.
9.2 Conclusion

Vijjana has been equipped with a recommender system base on collaborative techniques which provides appropriate recommendations for its users. This recommender system saves time and work for the user in finding the most useful data on the web. This Vijjana's recommender system acts as a series of sieves that provides good results. Statistical techniques have been implemented which produce more accurate results with time, as it would be able to predict more efficiently.
References


[24] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, John Riedl
Recommender System in Vijjana

By

Swetha Koduri

Date: 04/08/2010

What is a Recommender system?

- Chooses 'good' articles
- Makes recommendations
- Eases the process of sifting through the knowledge base
In the context of Vijjana...

Is it necessary ??
User based collaborative filtering
User based collaborative filtering

Top N items

Top N items

Top N items

Top N items

User based collaborative filtering

Average item rating

Average item rating

Average item rating

Average item rating

Average item rating

50
**Drawbacks**

- Scalability
- Sparsity

---

**Recommender System in Vijjana**

- Personalized recommendations
- Sifting through the knowledge base
- Item based analysis
- Overcomes sparsity and scalability
### Item based collaborative filtering

<table>
<thead>
<tr>
<th>USERS</th>
<th>ITEMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>??</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

---

52
**Similarity Calculation**

- By following the adjusted cosine similarity calculation, similarity is determined.

$$
\text{sim}(i, j) = \frac{\sum_{u \in U} (R_{u,i} - \bar{R}_u)(R_{u,j} - \bar{R}_u)}{\sqrt{\sum_{u \in U} (R_{u,i} - \bar{R}_u)^2} \sqrt{\sum_{u \in U} (R_{u,j} - \bar{R}_u)^2}}
$$
Implementation in Vijjana

- Input – rating from the users
- Output – recommendations for the users
Conclusion

- Recommender system has been implemented
- Item based personalized recommendations

Thank you!!