EXTENDING SINGULAR VALUE DECOMPOSITION BASED
RECOMMENDATION SYSTEMS WITH TAGS AND ONTOLOGY

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

YAKUP TURGUT

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

JUNE 2014

Approval of the thesis:

EXTENDING SINGULAR VALUE DECOMPOSITION BASED RECOMMENDATION SYSTEMS WITH TAGS AND ONTOLOGY

submitted by **YAKUP TURGUT** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** ______________

Prof. Dr. Adnan Yazıcı
Head of Department, **Computer Engineering** ______________

Prof. Dr. İsmail Hakkı Toroslu
Supervisor, **Computer Engineering Department, METU** ______________

Examining Committee Members:

Assoc. Prof. Dr. Pınar Karagöz
Computer Engineering Department, METU ______________

Prof. Dr. İsmail Hakkı Toroslu
Computer Engineering Department, METU ______________

Prof. Dr. Ahmet Coşar
Computer Engineering Department, METU ______________

Dr. Güven Fidan
AGMLab ______________

Dr. Onur Tolga Sehitoglu
Computer Engineering Department, METU ______________

Date: ______________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: YAKUP TURGUT

Signature :

ABSTRACT

EXTENDING SINGULAR VALUE DECOMPOSITION BASED RECOMMENDATION SYSTEMS WITH TAGS AND ONTOLOGY

Turgut, Yakup
M.S., Department of Computer Engineering
Supervisor : Prof. Dr. İsmail Hakkı Toroslu

June 2014, 43 pages

Due to increase of the volume of data related to user ratings on items, in recent years, recommendation systems became very popular, which uses this data in order to recommend items to users in many different domains. Singular Value Decomposition is one of the most widely studied collaborative filtering recommendation techniques. In some applications users are also allowed to enter (sometimes free) tags in addition to their ratings on items. Adding tags in addition to regular users' ratings on items have also been studied from different perspectives. In this work, we embedded tags entered by users into SVD technique in a simple but novel way. We also present methods that incorporate ontology to determine relationships between tags into consideration while dealing with movie recommender systems. We have applied our approach on movie recommendation system.

Keywords: Recommender Systems, Tagging, Ontology

ÖZ

TEKİL DEĞER AYRIŞIMI TABANLI ÖNERİ SİSTEMLERİNİN ETİKET VE ONTOLOJİ KULLANARAK GENİŞLETİMİ

Turgut, Yakup
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. İsmail Hakkı Toroslu

Haziran 2014, 43 sayfa

Son yıllarda kullanıcıların eşyaları oylarını barındıran verilerin artışı ile bu veriyi kullanıcılara yeni eşyalar önermekte kullanan öneri sistemleri oldukça popülerleşti. Tekil Değer Ayrışımı (TDA) en çok üzerinde çalışılan işbirlikçi filtreleme öneri tekniğidir. Bazı uygulamalarda kullanıcıların oyların yanına eşyalara (bazen serbest şekilde) etiket girmelerine de izin verilmektedir. Olağan kullanıcı oylarına ek olarak etiketlerin eklenmesi şimdiye dek farklı perspektiflerden çalışılmıştır. Bu çalışmada kullanıcılar tarafından girilen etiketleri TDA tekniğine basit ama yenilikçi bir şekilde katıyoruz. Ayrıca film öneri sistemleri üzerinde etiketler arası ilişkilerin ontoloji kullanılarak dahil edildiği yöntemler de sunuyoruz.

Anahtar Kelimeler: Öneri Sistemleri, Etiketleme, Ontoloji

To my family

ACKNOWLEDGMENTS

I would like to thank my supervisor Professor İsmail Hakkı Toroslu for his constant support, guidance, advice and criticism. It was a great honor to work with him for this work.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction

Currently, recommender systems are one of the most popular techniques studied by both academia and industry. Most of the companies are using these methods to increase their sales by providing items to their customers which may interest them. Recommendation is the process of recommending new personalized items to a user, based on the previously gathered data from the users, and/or the characteristics of users and items. The former approach is known as collaborative filtering technique, which develops a model to predict items for users by using user's past decisions and the similar decisions made by other users. The later technique is called as content-based filtering, which utilizes the characteristics of items/users in order to recommend new items with similar characteristics. Recommendation systems have been widely applied in many domains. Typical ones are movies, music, books, etc. In a typical movie recommender system, users give ratings to the movies and new recommendations are only made based on these ratings which are used for user and item based similarity. Since in movie domain the amount of data is huge, collaborative filtering techniques do not face with the so called cold-start problem. Therefore, it is preferred over content-based filtering techniques since it does not require complex analysis and modeling of items like movies. Since the amount of data (user-item ratings) is huge in movie domain, scalability also appears as an important problem. Moreover, although the data is huge, it is usually also very sparse. Therefore, achieving high accuracy for the recommendation is not easy either.

False negatives, the items user would like to buy, however not recommended by the recommender system and false positives, the items user does not like, and however recommended by the recommender system are two typical errors in recommender systems. Usually false positives are more crucial due to their negative impact on users because they may result in customer loss [30] [11].

Currently, most widely used technique to deal with performance problem and noisy data is Singular Value Decomposition (SVD) which reduces the dimensions of the original data matrix. SVD provides the best low-rank linear approximation of the original matrix and the low-rank approximation of the original matrix is sometimes even better than the original matrix itself [32] [6]. Filtering out of the small singular values can be introduced as removing "noise" data in the matrix. Researchers [32] [6] suggest that SVD-based approaches produce results better than traditional collaborative filtering algorithms most of the time.

In a typical movie recommender system, users give ratings to the movies and new recommendations are only made based on these ratings which are used for user and item based similarity. For example, most recommender systems based on Collaborative Filtering becomes less beneficial by only operating on user and item data [7] [16] [17] [21] [34]. With this kind of approach an important element, tags entered by users, is ignored therefore the quality of recommendations is reduced. Since tags have invaluable information about both users and movies, using tags in recommendation process may improve the results.

There are two different dimensions related to using tags in recommender systems. One approach tries to recommend tags to users [34]. The idea in this approach is that, the items will be both rated and tagged, but the tags are not going to be chosen arbitrarily, and the user has to choose a tag from a list recommended to her. This way tags associated with items will be controlled. The other approach tries to incorporate tags in the recommendation process. This simply adds another dimension to the data structure, in addition to users and items, which is called as tensor. There are several works trying to resolve issues related to tensor structure [35].

Since the standard approach requires tensor structures in order to expand user-item relationship with tags, many researchers focused on converting tensors to matrix structures [35] [28]. In [24], tag similarity has also been added into the process instead of forcing users to pick tags from predefined lists.

This thesis is also related to this second approach, namely using tags to improve recommendation accuracy. The idea is quite simple: a user's behavior and her use of tags are related to each other, that is, it may reflect one of her mood states. For example, when she uses a tag "funny" she is in one mood state, and when she uses a tag "bad" she is in a different mood state. So, she may have different rating behaviors for each one of these states. Similar comments may also be made for the items. The interpretation of an item by a user and the tag associated with it are related with each other. So, we can have several copies of an item for each tag, corresponding to different interpretations of the item. When the number of tags are very large, this will extend the user-item ratings matrix a lot, and also it will make it much sparser as in the case implemented in [25]. Therefore, only frequent tags must be considered for this extension. Also using free-text tag data as it is may not be enough, therefore we also use ontology to define relationships between tags and incorporate them into recommendation process.

1.2 Organization

This thesis is started with introduction section. Then, detailed information of recommender systems and ontology are given in the second chapter. In Chapter two, types of recommendation techniques and SVD are explained. For the rest of the chapter, information about ontology, semantic similarity and similarity methods are given. Proposed algorithms that utilizes tags and ontology in recommendation process are introduced in chapter three. Next in Chapter four, experimental results which are based on MovieLens[3] data are presented. Finally, the thesis is concluded with some information about future work.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this chapter, background knowledge related to this work and related works in the literature are explained.

2.1 RECOMMENDER SYSTEMS

Recommendation is the process of recommending new items to a user based on the previously gathered data from the users. In a typical e-commerce environment, there are a lot of items that a user can buy and most of the time, it is impossible for a user to know every aspect of the items that he/she wants to buy. Therefore, recommender systems aim to help people choose the products they need. At the same time, these systems aim to increase sales rate by recommending the suitable items to their customers. Several online firms, including Yahoo!, Amazon.com and Movie Critic, recommend documents and products to consumers. Recommendation systems mainly examine user data and make predictions based on extracted information from this data [5].

Recommender systems can be classified into five different categories based on the information and how they use this information: Collaborative, Content-based, Demographic, Utility-based and Knowledge-based. These techniques utilizes the background information about users and items, the data collected as user interacts with the system and make recommendations by combining them [10].

2.1.1 Collaborative Recommender Systems

Most widely used recommendation technique is collaborative recommendation. These systems are based on the idea that users are represented with vectors containing ratings that are given to the items by the user. Rating data that constitute the user profile are collected over the time by with the user interaction. Then collaborative recommender systems try to find similarities between users by taking these rating vector into account and make suggestions [10]. These systems can be categorized into two classes [7]:

- Memory-based algorithms which compare active user with every other user in the user database.
- Model-based algorithm which extract a model based on rating data collected on user database over time.

2.1.2 Content-based Recommender Systems

Content-based recommendation systems use item features. These systems recommend items based on these features and user profile that is formed with user history of preferred items. Definition of item features may differ between various systems, however for all of them, use of features for items and creation of user profile over ratings given by user to these features are the common characteristics of these systems. As user interacts with the systems, similar to collaborative systems, user profile is updated with the feedback given [10] [26]

2.1.3 Demographic Recommender Systems

Demographic recommender systems will either collect information about user's personal attributes by explicitly asking users via surveys or implicitly via machine learning technique to discover classifiers on demographic data [26] and they will use this information to classify users and make recommendations to them. However, different system may represent in various ways. With this approach, recommender system does not need to build a history of user ratings for recommendation [10].

2.1.4 Utility-based Recommender Systems

In utility-based recommender systems, for each item for the user, the utility of it is computed and new recommendations are made based on this computed value. As background data, features of items are used, description of user preferences are brought out with utility functions over items from users to derive user profile and this utility function is applied in order to find the rank of items for a user. With this technique, non-product attributes can be taken into the recommendation process by incorporating these attributes into computation of utility function, such attributes may be vendor reliability and product availability. How the utility function is computed is the central problem for this technique because each user needs to build preference function and weight each attribute according to its importance. However, these systems have the advantage of not having sparsity or inclusion of new users or items [10][18].

2.1.5 Knowledge-based Recommender Systems

Knowledge-based recommender systems make use of the knowledge about their users and items so that they can make recommendations based on this knowledge and reasoning about user's requirements of items. They try to infer user's needs and preferences by explicitly asking from user, implicitly gathering user-entered data such as search queries in Google case and combine this information with the data about items such as catalog knowledge. Knowledge-based recommender systems do not need to have large amount of data to make recommendations since they do not depend on collection of user ratings and they do not depend on individual tastes [10] [9].

2.2 SINGULAR VALUE DECOMPOSITION

Singular Value Decomposition (SVD) is the most widely used technique to deal with noisy data and is used as dimensionality reduction technique to overcome performance problems caused by dealing large amount of data. SVD provides the best low-rank linear approximation of the original matrix and this approximation of original matrix may give better results than the original matrix itself [32] [6].

2.2.1 Mathematical Definition of SVD

Given an mxn matrix A, where without loss of generality $m \geq n$ and $rank(A) = r$, the SVD of A is defined as

$$A = U\Sigma V^T \tag{2.1}$$

where $U^{\mathrm{T}}U = V^{\mathrm{T}}V = I_n$ and $\Sigma = diag(\alpha_1, ..., \alpha_n)$, $\alpha_{\mathrm{i}} > 0$ for $1 \leq i \leq r, \alpha_j = 0$ for $j \geq r + 1$. The first r columns of the orthogonal matrices U and V define the orthonormal eigenvectors associated with the r nonzero eigenvalues of AA^{T} and $A^{\mathrm{T}}A$, respectively. The columns of U and V are respectively referred as the left and right singular vectors and the singular values of A are defined as the diagonal elements of Σ, which are the nonnegative square roots of the n eigenvalues of AA^{T} [14] [6].

In regular recommender systems SVD is applied to rating matrix. For the matrix A which SVD produces 3 matrices namely U, Σ, V; U and V^{T} matrices are reduced and used for recommendation, U for user-based recommendation and V^{T} for item-based recommendation.

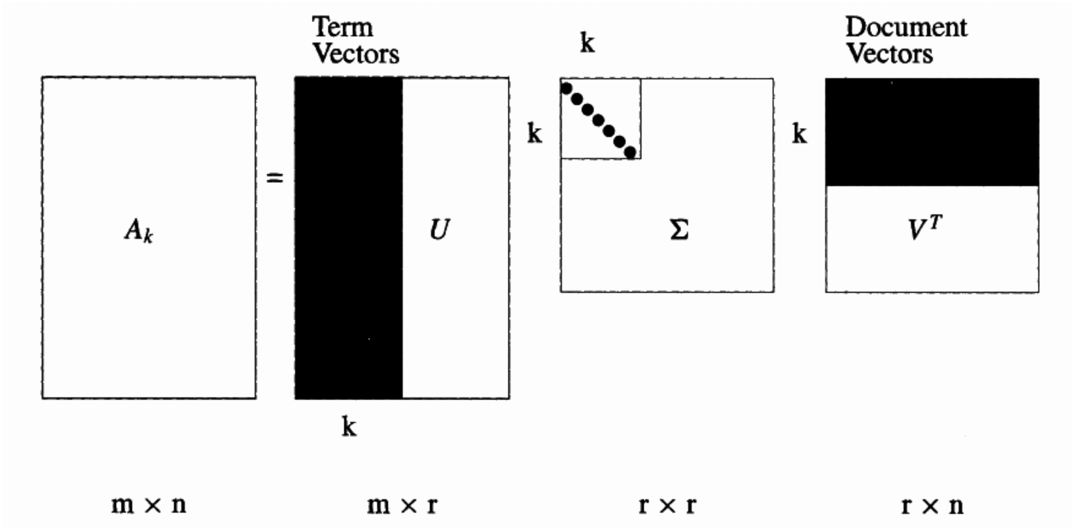

Figure 2.1: Visualization of Singular Value Decomposition. Adapted from [6]

2.3 ONTOLOGY

In Philosophy, Ontology is defined as studying and modeling real world objects with the way they are in the nature [8][12]. In computational terms, Ontology is defined as explicit formalization of knowledge, concepts of objects and the relationship between them in a particular area to give a simplified, abstract view [15]. Ontologies help computers to work with real world data in a better way and without having a structured way of handling these data, computers may fail give good results even though lots of data generated [13]. In this thesis, ontology is used to handle the unstructured tag data entered by user for movies they watched and rated.

2.3.1 WordNet

WordNet [4] is an online vocabulary and a lexical ontology which tries to model lexical knowledge of a native speaker of English into a taxonomic hierarchy. This model composed of connected semantic links such as generalization similarity, exclusion which makes WordNet an ontology for natural language. Organization of WordNet starts with synsets that is synonymy list of terms or concepts and these synsets forms senses, different meanings of the same term of concept. Every term or concept is linked with each other by different kinds of relationships. Examples of these relationships are [19]:

- **Synonymy:** Similarity in meaning of words, which is used to build concepts represented by a set of words.
- **Antonymy:** Dichotomy in meaning of words - mainly used for organizing adjectives and adverbs.
- **Hyponymy:** Is-a relationship between concepts. This is-a hierarchy ensures the inheritance of properties from super concepts to sub concepts.
- **Meronymy:** Part-of relationship between concepts.
- **Morphological Relations:** These relations are used to reduce word forms.

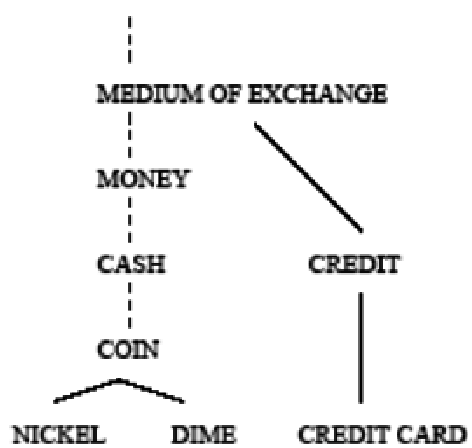

Figure 2.2: A fragment of WordNet Taxonomy. Adapted from [19]

2.3.2 Semantic Similarity

Semantic similarity is the term used for computation of the similarity between lexicographically dissimilar concepts. Several methods have been developed to perform to compute this similarity. These methods make use of the idea that there are additional properties, besides their names, of the entities, different levels of generality and relationships with other concepts defined in ontologies when comparing them. Traditional keyword-based comparison methods cannot take advantage of this information. The methods used for semantic similarity comparison can be summarized as [19]:

- **Edge Counting Methods:** These methods determine the path linking the terms and the position of them in the taxonomy assuming that similarity between two concepts are measured by the idea of having more links between the concepts shows that they are more similar and closely related [27].
- **Information Content Methods:** These methods are based on the idea that similarity of two concepts can be defined using Information content. These methods assume that similarity between concepts can be calculated by determining the information they share in common.
- **Feature based Methods:** In order to compute similarity, these measures also consider the features of terms which valuable information exist.
- **Hybrid methods:** Combination ideas from the above three approaches in order to compute semantic similarity between two concepts.

CHAPTER 3

EXTENDED SVD RECOMMENDATION

3.1 FINDING N-MOST COMMON TAGS

In order to extend rating matrix with tags, N most common tags are found for both users and movies. For each user and for each movie, the frequency of the tags entered is found and N most common tags is taken into consideration while calculating Extended SVD Matrix. Before applying any method, tags are cleansed to eliminate user errors. Following methods have been used to calculate N most common tags.

String Equality: Java string equality method is used to calculate N most common tags.
Edit Distance: Edit distances are calculated and tags whose distances are below threshold assumed to be equal to calculate N most common tags.
Substring: Java substring method is used and tags either of which is substring of another is assumed to be equal to calculate N most common tags.

3.2 EXTENDING MATRIX WITH N-MOST COMMON TAGS

In this method, rating matrix is extended with the tags found with the methods described in previous section. In order not to lose information about rest of the tags, a special tag is used i.e. <other> in the rating matrix if there exists some tags not included in N-most common ones. For both regular and extended matrices following steps are applied:

1. Apply SVD to the matrix and find U, S, V^t

2. Reduce the dimensionalities of the U and V^t matrices

3. For user-based recommendation, calculate distances between users by taking U matrix into consideration.

4. Determine the item's rating of the user that is closest to the relevant user.

5. For item-based recommendation, calculate distances between items by taking V^t matrix into consideration.

6. Determine the item's rating of the item that is closest to the relevant user's items.

3.2.1 EXAMPLE OF REGULAR AND EXTENDED MATRIX RECOMMENDATION

3.1 shows a regular rating matrix where users' ratings to movies are stored and 3.6 shows an extended rating matrix. In the extended matrix, when tags match for both user and item, then rating is stored in that cell, if they do not match then zero is stored.

Table 3.1: Regular Rating Matrix

User / Movie	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$
$\mathbf{U_1}$	4.0	4.0	4.0
$\mathbf{U_2}$	3.0	4.0	3.0
$\mathbf{U_3}$	4.0	1.0	0.0

First step of the recommendation process is to apply SVD and find the U and V^t matrices and reduce their dimensions. After performing these step to regular rating matrix, 3.2 and 3.3 are obtained.

Table 3.2: Reduced User Matrix for Regular Matrix

$\mathbf{U_1}$	-0.7245	-0.2030
$\mathbf{U_2}$	-0.6058	-0.2685
$\mathbf{U_3}$	-0.3289	0.9417

Table 3.3: Reduced Movie Matrix for Regular Matrix

M_1	-0.6339	-0.5938
M_2	0.7539	-0.3312
M_3	-0.1728	0.7333

Table 3.4: User Distance Matrix for Regular Matrix

U_1, U_2	0.1356
U_1, U_3	1.2111
U_2, U_3	1.2415

Table 3.5: Movie Distance Matrix for Regular Matrix

M_1, M_2	1.4124
M_1, M_3	1.4049
M_2, M_3	1.4113

3.4 and 3.5 show the distances within the users and within the movies respectively. According to the user distance matrix, most similar user to user U_3 is U_1, therefore the user-based recommendation system will calculate U_1's rating as predicted value for U_3's rating of the movie M_3 , which is 3.

According to the movie distance matrix, most similar item to movie M_3 is M_1, therefore the item-based recommendation system will calculate M_1's rating as predicted value for U_3's to the movie M_3 , which is 4.

Table 3.6: Extended Rating Matrix

	M_1, t_1	**M_1, other**	**M_2, t_2**	**M_2, t_3**	**M_2, other**	**M_3, t_2**	**M_3, other**
U_1, t_1	4.0	0.0	0.0	0.0	0.0	0.0	0.0
U_1, other	0.0	4.0	0.0	0.0	4.0	0.0	4.0
U_2, t_3	0.0	0.0	0.0	4.0	0.0	0.0	0.0
U_2, other	0.0	3.0	0.0	0.0	4.0	0.0	3.0
U_3, t_2	0.0	0.0	1.0	0.0	0.0	0.0	0.0
U_3, other	0.0	4.0	0.0	0.0	1.0	0.0	0.0

The suggested algorithm will work almost in the same way reference algorithm work in this phase.

When we evaluate 3.9, we can conclude that the most similar user to U_3 is U_1, there-

Table 3.7: Reduced User Matrix for Extended Matrix

U_1, t_1	0.0	0.0
U_1, <other>	0.7245	0.0
U_2, t_3	0.0	1.0
U_2, <other>	0.6058	0.0
U_3, t_2	0.0	0.0
U_3, <other>	0.3289	0.0

Table 3.8: Reduced Movie Matrix for Extended Matrix

M_1, t_1	0.0	0.0
M_1, <other>	0.6339	0.0
M_2, t_2	0.0	0.0
M_2, t_3	0.0	1.0
M_2, <other>	0.5938	0.0
M_3, t_2	0.0	0.0
M_3, <other>	0.4956	0.0

Table 3.9: User Distance Matrix for Extended Matrix

(U_1, t_1), (U_1, <other>)	0.7245
(U_1, t_1), (U_2, t_3)	1.0000
(U_1, t_1), (U_2, <other>)	0.6058
(U_1, t_1), (U_3, t_2)	0.0000
(U_1, t_1), (U_3, <other>)	0.3289
(U_1, <other>), (U_2, t_3)	1.2349
(U_1, <other>), (U_2, <other>)	0.1187
(U_1, <other>), (U_3, t_2)	0.7245
(U_1, <other>), (U_3, <other>)	0.3956
(U_2, t_3), (U_2, <other>)	1.1692
(U_2, t_3), (U_3, t_2)	1.0000
(U_2, t_3), (U_3, <other>)	1.0527
(U_2, <other>), (U_3, t_2)	0.6058
(U_2, <other>), (U_3, <other>)	0.2769
(U_3, t_2), (U_3, <other>)	0.3289

fore predicted rating will be 4.0 for the item M_3 for this user while using user-based recommendation.

On the other hand, for item-based recommendation, 3.10 shows that for user U_3 both M_1 and M_2 are equally similar item to M_3. This means that, depending on our strategy, we may randomly choose one and use its rating as predicted value or we may calculate

Table 3.10: Movie Distance Matrix for Extended Matrix

(M_1, t_1), (M_1, <other>)	0.6339
(M_1, t_1), (M_2, t_2)	0.0000
(M_1, t_1), (M_2, t_3)	1.0000
(M_1, t_1), (M_2, <other>)	0.5938
(M_1, t_1), (M_3, t_2)	0.0000
(M_1, t_1), (M_3, <other>)	0.4956
(M_1, <other>), (M_2, t_2)	0.6339
(M_1, <other>), (M_2, t_3)	1.1840
(M_1, <other>), (M_2, <other>)	0.0401
(M_1, <other>), (M_3, t_2)	0.6339
(M_1, <other>), (M_3, <other>)	0.1383
(M_2, t_2), (M_2, t_3)	1.0000
(M_2, t_2), (M_2, <other>)	0.5938
(M_2, t_2), (M_3, t_2)	0.0000
(M_2, t_2), (M_3, <other>)	0.4956
(M_2, t_3), (M_2, <other>)	1.1630
(M_2, t_3), (M_3, t_2)	1.0000
(M_2, t_3), (M_3, <other>)	1.1161
(M_2, <other>), (M_3, t_2)	0.5938
(M_2, <other>), (M_3, <other>)	0.0982
(M_3, t_2), (M_3, <other>)	0.4956

the average rating of them and use the average.

3.3 EXTENDING MATRIX WITH TAGS COUNTS

In this method, regular rating matrix is extended with tags and the cells that corresponds to (user, tag) are filled with the number of times the user entered that tag. Tag counting is performed with the function defined in 3.3. To ensure that same interval (i.e. 0 – 5) exists on all cells, the function defined in 3.3 is used for tag counts for the relevant user.

Function TagCounter()

$variables$: $UserTagCountMap := \{\}$ // counts initialized with zero.
$variables$: $MovieTagCountMap := \{\}$ // counts initialized with zero.
//for each tag data
for each $(user, movie, tag) \in TagData$ **do**
  UserTagCountMap[user, tag] := UserTagCountMap[user, tag] + 1
  MovieTagCountMap[movie, tag] := MovieTagCountMap[movie, tag] + 1
end for
return UserTagCountMap, MovieTagCountMap
EndFunction

Function ScaleCounts(TagCountMap)

$arguments$: $TagCountMap := \{\}$ // tag counts for users or movies computed in 3.3
$variables$: $maxCount := -1$ //
//for each tag data
for each $((element, tag), count) \in TagCountMap$ **do**
  if $maxCount < count$ **then**
    $maxCount := count$
  end if
end for
$logbase := maxCount^{\frac{1}{5}}$
for each $((element, tag), count) \in TagCountMap$ **do**
  $TagCountMap[element, tag] := log_{logbase}(count)$
end for
EndFunction

Same rule is applied for movie, tag cells. After filling whole matrix with these data, SVD is applied to it and regular recommendation process is carried out. The only exception is that rather than searching for similar user in the decomposed U matrix, only submatrix of it where user data, not tag data exist are considered, because otherwise some tag might show up as similar user which does not have any value on recommendation process.

3.4 EXTENDING MATRIX WITH ONTOLOGY

In this section, extension of regular recommender systems with ontology is described. In this method, rating matrix is extended with tags as in the case of tag count extension described in section 3.3. However, cells that correspond to (user, tag) or (movie, tag) are not filled with tag counts, they are filled with the semantic similarity distances between tags using WordNet ontology. Table 3.11 shows the extended matrix when the semantic distance between t_1 and t_2 is 0.56. Rest of the steps performed in algorithm is same with Tag Count Extension method.

Table 3.11: Extended Rating Matrix with Ontology

	$\mathbf{M_1}$	$\mathbf{M_2}$	$\mathbf{M_3}$	$\mathbf{t_1}$	$\mathbf{t_2}$
$\mathbf{U_1}$	4.0	4.0	4.0	0.0	0.0
$\mathbf{U_2}$	3.0	4.0	3.0	0.0	0.0
$\mathbf{U_3}$	4.0	1.0	0.0	0.0	0.0
$\mathbf{t_1}$	0.0	0.0	0.0	1.0	0.56
$\mathbf{t_2}$	0.0	0.0	0.0	0.56	1.0

3.5 EXTENSION WITH MEANINGFUL TAGS

In this method, extension of the rating matrix is same, however the tags used for extension are filtered to ensure that only meaningful and relevant tags are involved.

Two techniques are used. First one is to remove tags whose variance is higher than certain threshold when tags are associated with the ratings. Second one is to remove tags if the tags is used both 1.0 rated movies and 5.0 rated ones. Rest of the recommendation process is the same with the extension methods described above.

3.6 HYBRID RECOMMENDATION USING TAGS SET DISTANCE

For hybrid recommendation, the way the distance between users or between movies is changed. In order to calculate the distance between users, first SVD applied to regular rating matrix. Then two distances are calculated and merged in a weighted manner. First distance is the distance calculated in regular collaborative filtering recommender

(d_{cf}). Second one is the distance calculated with Jaccard Distance (d_j) [20] where tag sets are used.

$$d_j(A,B) = 1 - J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

$$d_{hybrid}(U_1, U_2) = (weight \times d_c f(U_1, U_2)) + (1 - weight) \times d_j(U_{1tags}, U_{2tags})$$

Finally, similar users or items are found with this hybrid distance method in the recommendation process.

3.7 PRE-CLUSTERED RECOMMENDATION

Pre-clustered recommendation is the process of clustering users first based on their tags and then applying regular recommendation technique on subsets of rating data which are separated based on these clusters. The aim of this technique is to include tags as similarity measure before ratings considered and work on smaller data set in order to make recommendation process faster. Users are clustered using k-medoids [22] clustering algorithm and Jaccard Distance is used as a distance metric while clustering. After the clustering, only difference between regular and pre-clustered recommender is to locate the subset of the rating dataset where the interested user is located which can be found by having a hash map. Rest of the process is same.

3.8 HOSVD-BASED RECOMMENDATION

In previous methods, two-dimensional matrices are used for recommendation process, however same data can be represented with multi-dimensional matrix, namely with tensors and recommender system can work on that. More specifically, since the data is about users, movies and tags, tensor will be constructed in a three-dimensional way. Front side of the 3D-cube contains the ratings given by users to the movies. Rest of the third dimension is indexed with tags and contains the information whether user is used that tag for that movie. For example if U_1 used t_1 for M_1 then the corresponding cell will be filled i.e. Matrix[U_1][M_1][t_1] = 5.0. In order to ensure that data is within the same range everywhere, 5.0 is used while filling the cells. Ontology distance between tags is also used while filling the rest of the cells. For instance, if user U_1

did not use t_2 for M_1, then ontology distances (d_{ont}) between t_2 and U_1's other tags are calculated and highest one is used. After the construction of tensor, Higher-Order SVD is applied to it and recommendation is made based on that.

$$Matrix[U_{\mathrm{i}}][M_{\mathrm{j}}][t_{\mathrm{k}}] = \begin{cases} 5.0, & \text{if } t_k \in tags_{U_i,M_j} \\ \max(d_{ont}(t_k,t_l))\forall t_l \in tags_{U_i} & \text{otherwise} \end{cases}$$

3.8.1 APPLICATION OF HOSVD

Our HOSVD-based recommendation is based on the work of Symeonidis et.al [34] applied to movie recommendation domain with ontology included. There are also other works based on authors' method such as geo-activity recommendation domain [33].

In this approach after the construction, tensor is unfolded to three to two dimensional matrices which are called 1-mode, 2-mode and 3-mode unfoldings respectively [23]:

$$\begin{aligned} A_1 &\in R^{I_1 \times I_2 I3} \\ A_2 &\in R^{I_2 \times I_1 I3} \\ A_3 &\in R^{I_1 I_2 \times I3} \end{aligned}$$

Figure 3.1 shows the visualization of unfolding of the tensor. After this process, SVD is applied to these 2-D matrices which:

$$\begin{aligned} A_1 &= U^{(1)} S_1 V_1^T \\ A_2 &= U^{(2)} S_2 V_2^T \\ A_3 &= U^{(3)} S_3 V_3^T \end{aligned} \tag{3.1}$$

HOSVD-based recommendation works very similar to SVD-based one. We operate on generated $U^{(1)}$ and $U^{(2)}$ matrices which are analogous to U and V^T matrices generated by SVD. We reduce the dimensionality of these matrices and rest of the process works exactly the same. $U^{(1)}$ is used for user-based recommendation and $U^{(2)}$ is used for item-based recommendation.

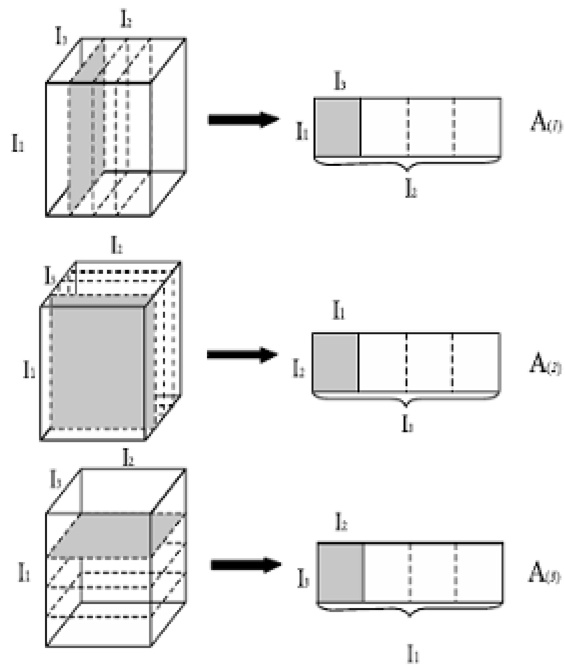

Figure 3.1: Visualization of Unfoldings [34]

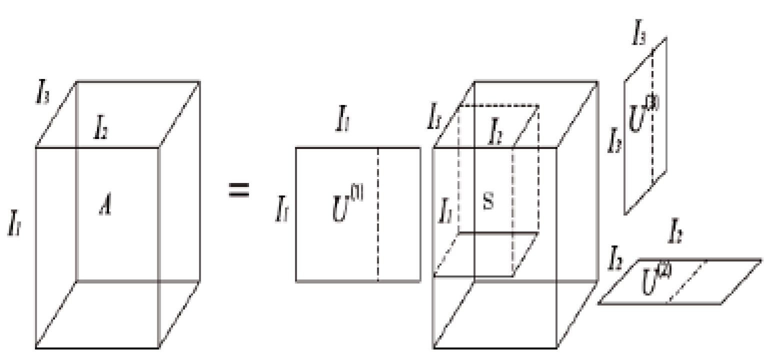

Figure 3.2: Visual representation of HOSVD [34]

CHAPTER 4

EXPERIMENTAL RESULTS

In this chapter, we are going to compare algorithms by using MovieLens data with the reference SVD-based recommendation algorithm.

4.1 Data Set

MovieLens 10M data which contains 10 million ratings and 100,000 tags to 10,000 movies by 72,000 users is used in experiments [3]. Data have the following structure:

Rating Data: UserID::MovieID::Rating::Timestamp
Tag Data: UserID::MovieID::Tag::Timestamp

MovieLens is a popular dataset used by many researchers. A very similar work which our work based on was to categorizing items by their genres or users by their gender and use of tags by just appending them to the rating matrix [25]. There is also another research that compares item-based recommendation to user-based recommendation and different item-based recommendations that concludes item-based recommendation methods are superior to the user-based ones [31]. Likewise, Data set is used in testing some hybrid recommendation systems that try to improve collaborative filtering technique with content based filtering methods where properties such as actors, directors, and film genres are used [29].

4.2 Accuracy Metric

Mean Absolute Error (MAE) is used to determine and compare accuracy of the recommendation algorithms while evaluating the experiment results. The formula is as follows:

$$\text{MAE} = \frac{1}{n}\sum_{1}^{n}|f_i - y_i| \qquad (4.1)$$

where f_i is the predicted rating and y_i is the actual rating of the user[2]. In order to compare results between reference algorithm and suggested algorithms, 5-fold cross-validation is used. The data are split into two parts where %80 of that is used for training and %20 of that is used for testing.

4.3 Pre-processing of Tag Data

Since users are allowed to enter any tag to any movie freely without having a computer-aided way in MovieLens, these tags need to be cleaned before they can be used. For example, users may have misspelled the words, therefore to match same tag for different users, we need to correct them. JaSpell library [1] is used to correct misspellings of the words. Some punctuation marks are removed such as '?' and some of them replaced i.e. ‘ -> ’ to have a unified view of tags.

4.4 Results for Extended SVD Recommendation

In this section, experimental results of the algorithm described in Chapter 3 are given. Figure 4.1 shows the results when using string equality to match tags in Extended Recommendation. Results show that there is not a significant improvement in the accuracy of recommendation while using string equality as tag matching algorithm. In fact for user-based recommendation, suggested algorithm performs a little worse.

In figure 4.2, the effect of using different methods for tag matching is shown with using the subset of MovieLens data.

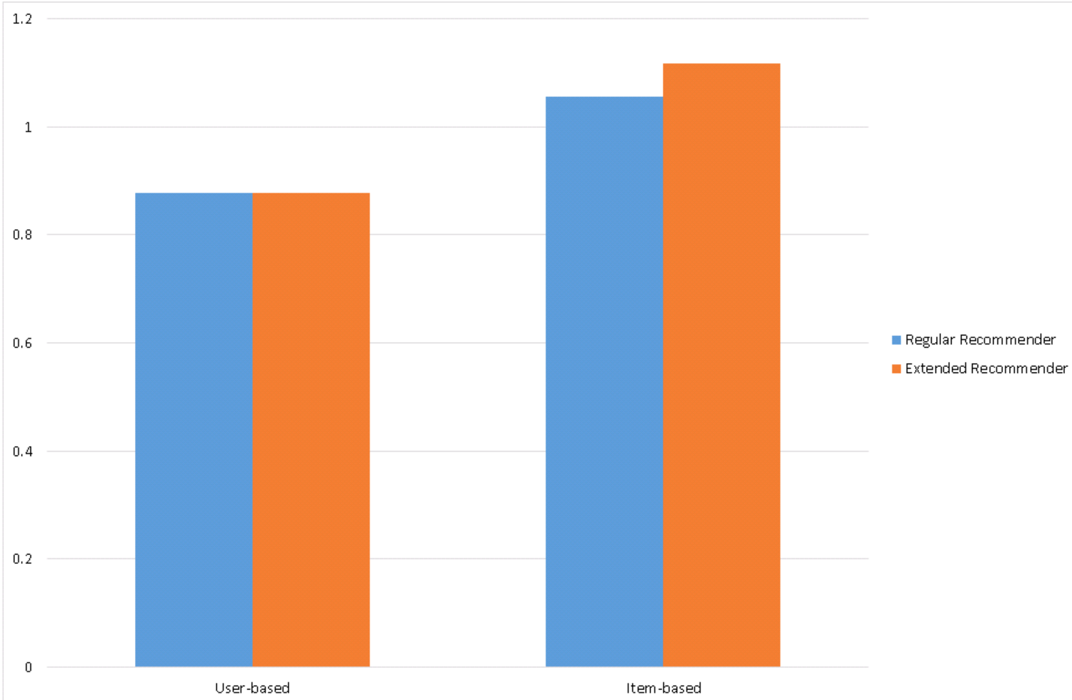

Figure 4.1: Experiment Results - MAE

In figure 4.3, the effect of changing N which is the number of most common tags used for extending matrix is depicted. These results shows that working with unstructured tag data to extend recommendation matrix does not improve recommendation accuracy.

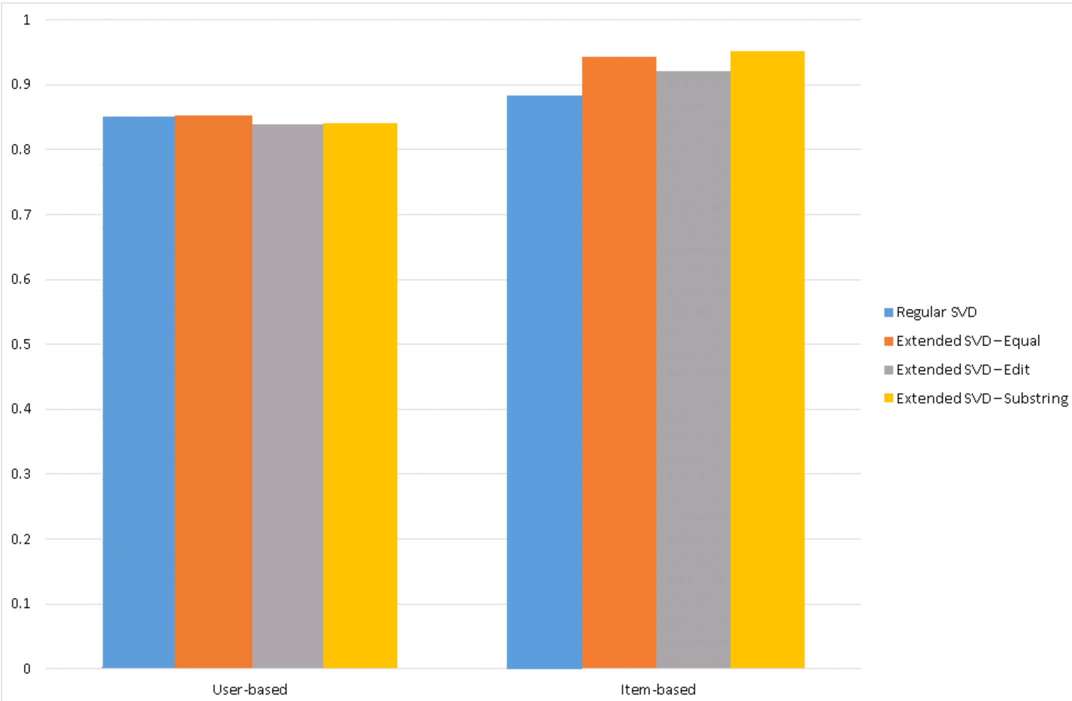

Figure 4.2: Recommendation Results for Different Tag Matching Methods

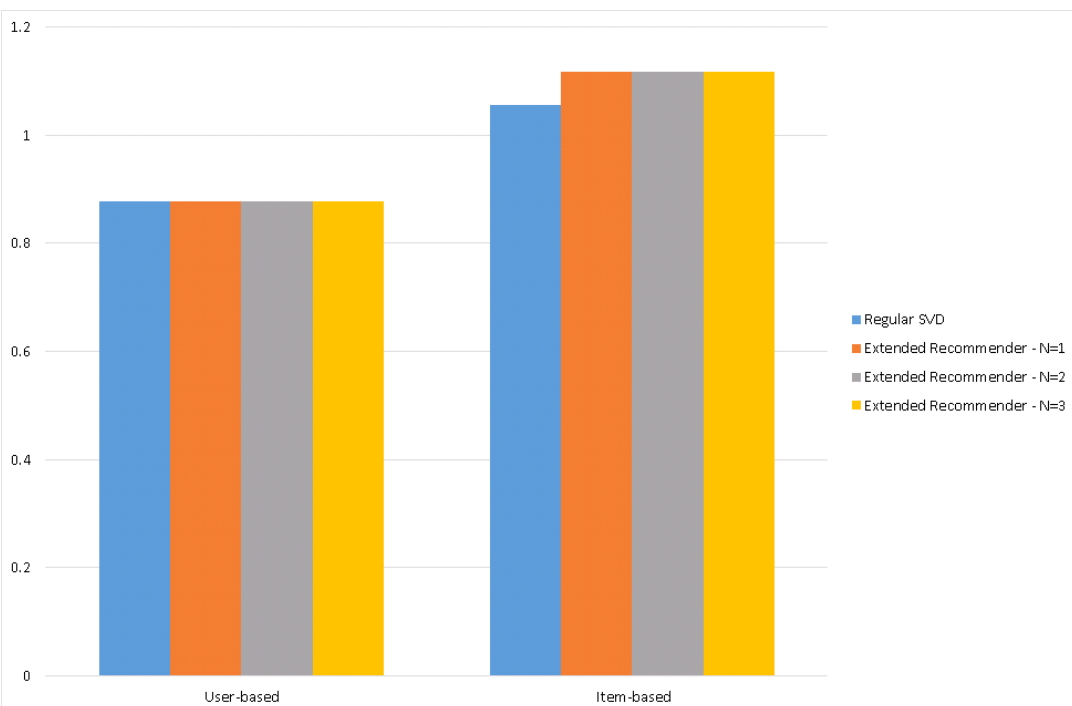

Figure 4.3: Experiment Results for Different Values of N-most Common tags - MAE

4.5 Results for Extended SVD Recommendation with Tag Count

In this section, experiment results for tag count extension are provided. Figure 4.4 shows the effects of extension regular matrix with tag count data. Accuracy results for both regular and extended regular matrix are almost same. It seems that added data does not affect results since after SVD is applied and matrices' dimensions are reduced, recommendation process works on subset of the resulted matrices. Our conclusion is that inclusion of tag count data does not influence the working set of recommender system.

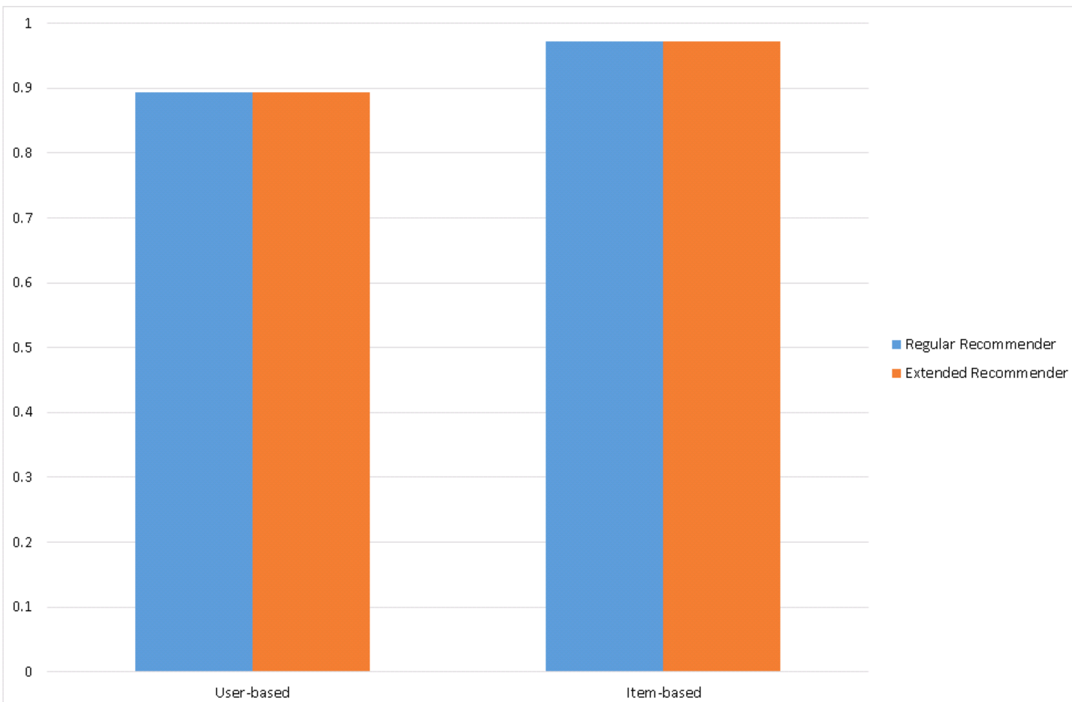

Figure 4.4: Experiment Results for Tag Count Extended Rating Matrix

4.6 Results for Extended SVD Recommendation with Ontology Similarity

In this sections, experiment results for ontology similarity extension are provided. Figure 4.5 shows the effects of extension regular matrix with tag count data. For this experiment, Accuracy results for both regular and extended regular matrix are also almost same. This shows that, this method also has the same drawbacks of the tag count extension method.

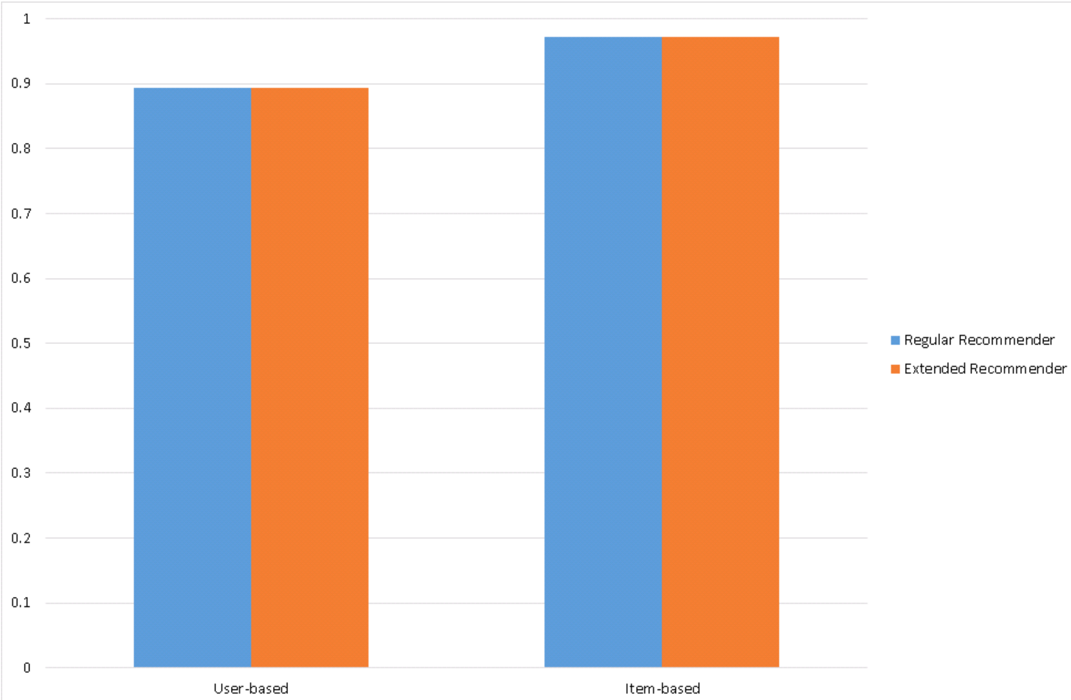

Figure 4.5: Experiment Results for Extended Rating Matrix with Ontology Similarity

4.7 Results for Extension with Meaningful Tags

In this section, experiment results for extension with meaningful tags are given. Figure 4.6 shows the effects of removal of tags whose variance is higher than 1.5 when associated with ratings. Figure 4.7 shows the results when tags that are used both 1.0 rated movies and 5.0 rated movies. Both figures show that recommendation do not get better with the use of more meaningful tags.

Table 4.1: Statistics: Meaningful Tags Variance Threshold

Number of Unique Users	588
Number of Unique Movies	1997
Number of Unique Tags	1826

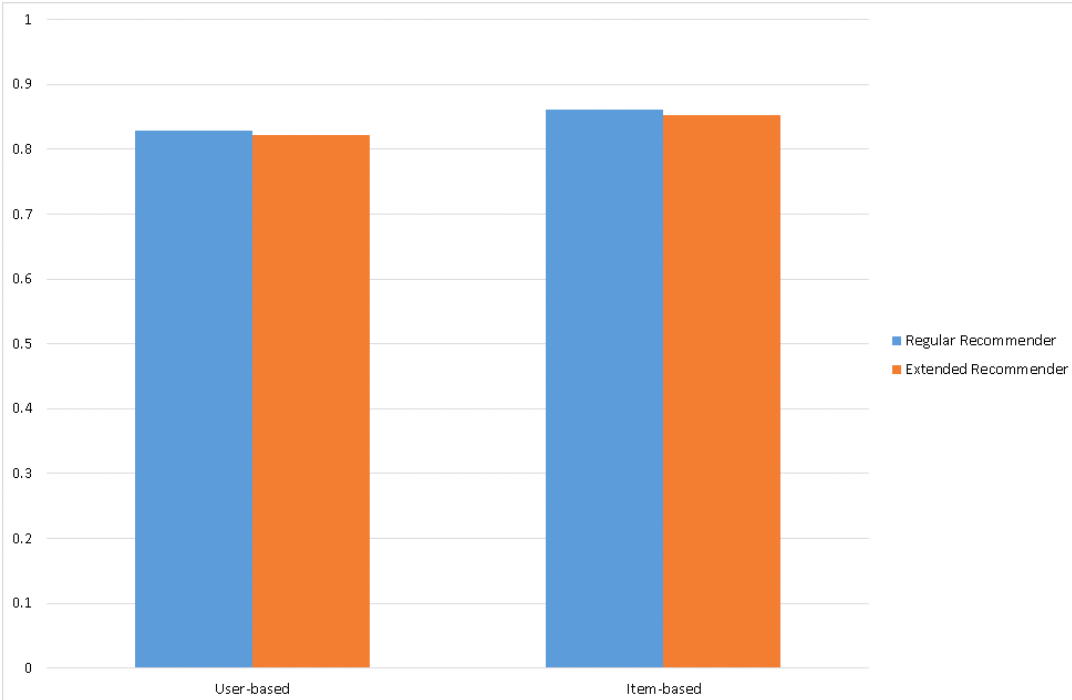

Figure 4.6: Experiment Results for Extension with Tags: Variance Threshold

Table 4.2: Statistics: Meaningful Tags 1.0 and 5.0 Rated Tags

Number of Unique Users	588
Number of Unique Movies	1997
Number of Unique Tags	1794

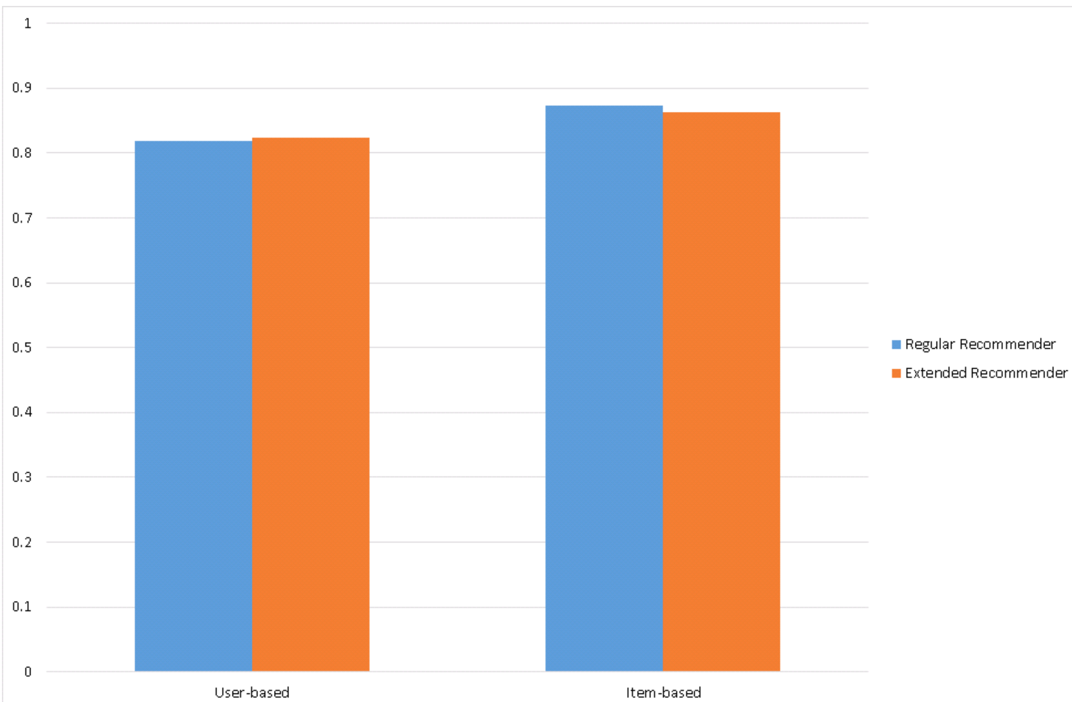

Figure 4.7: Experiment Results for Extension with Tags: 1.0 and 5.0 Rated Tags Removed

4.8 Experiment Results for Hybrid Recommendation Using Tags Set Distance

We have evaluated the effect of Hybrid Recommendation where Jaccard distance is used while calculating user similarities via their tag sets and is merged with Collaborative Filtering (CF) recommender distancein order to generate recommendations. Figure 4.8, 4.9 and 4.10 show the results and effects of hybrid recommendation with the ratio change: i.e. 80% CF - 20% Jaccard, 70% CF - 30% Jaccard, 50% CF - 50% Jaccard respectively. It can be concluded that the higher the ratio of Jaccard distance is the worse recommendation results become. This might indicate that quality of the tags in the dataset is not enough representative to improve recommendation.

Table 4.3: Statistics: Hybrid Recommendation Using Tags Set Distance

Number of Unique Users	588
Number of Unique Movies	1997
Number of Unique Tags	1933

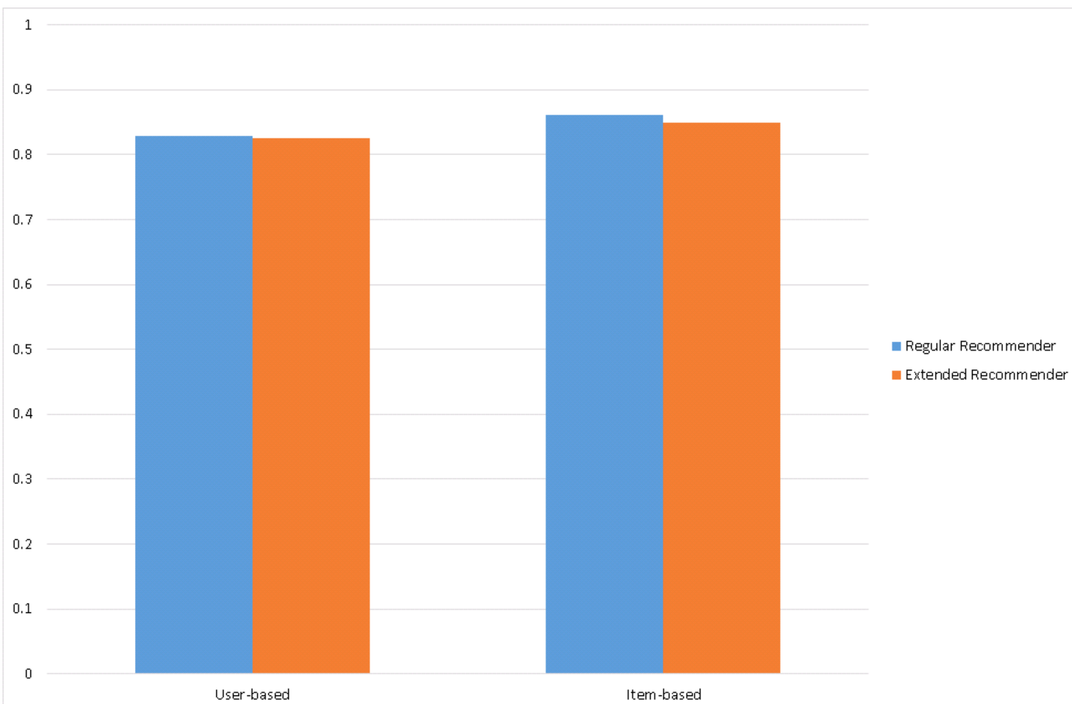

Figure 4.8: Experiment Results for Hybrid Recommendation: 80% - 20%

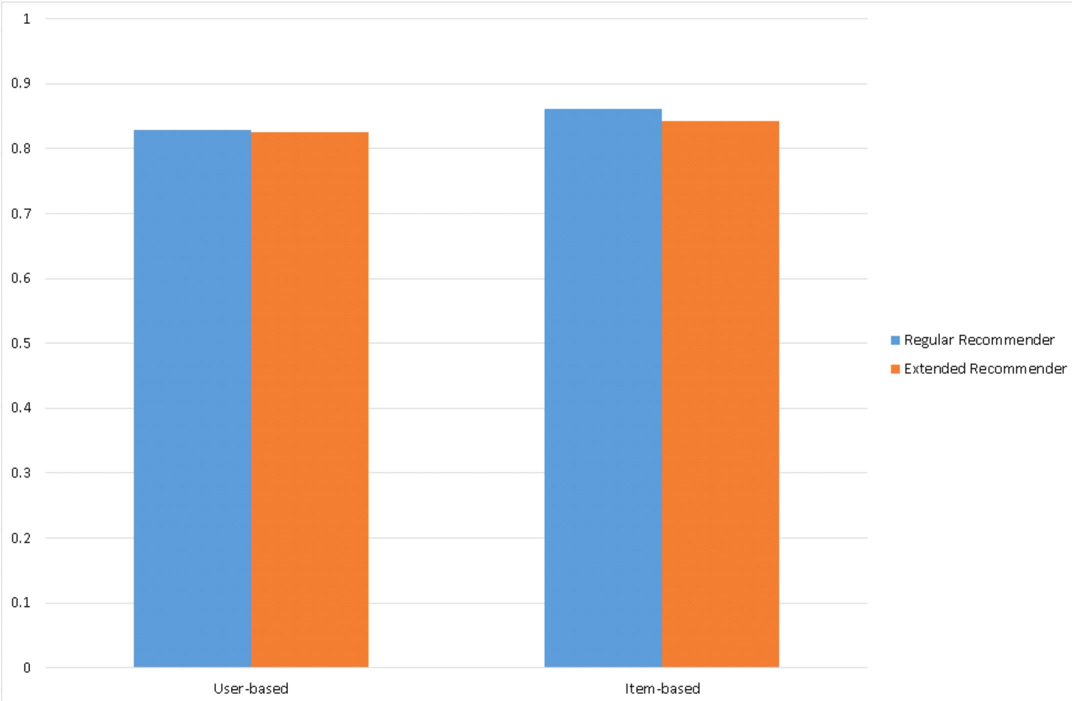

Figure 4.9: Experiment Results for Hybrid Recommendation: 70% - 30%

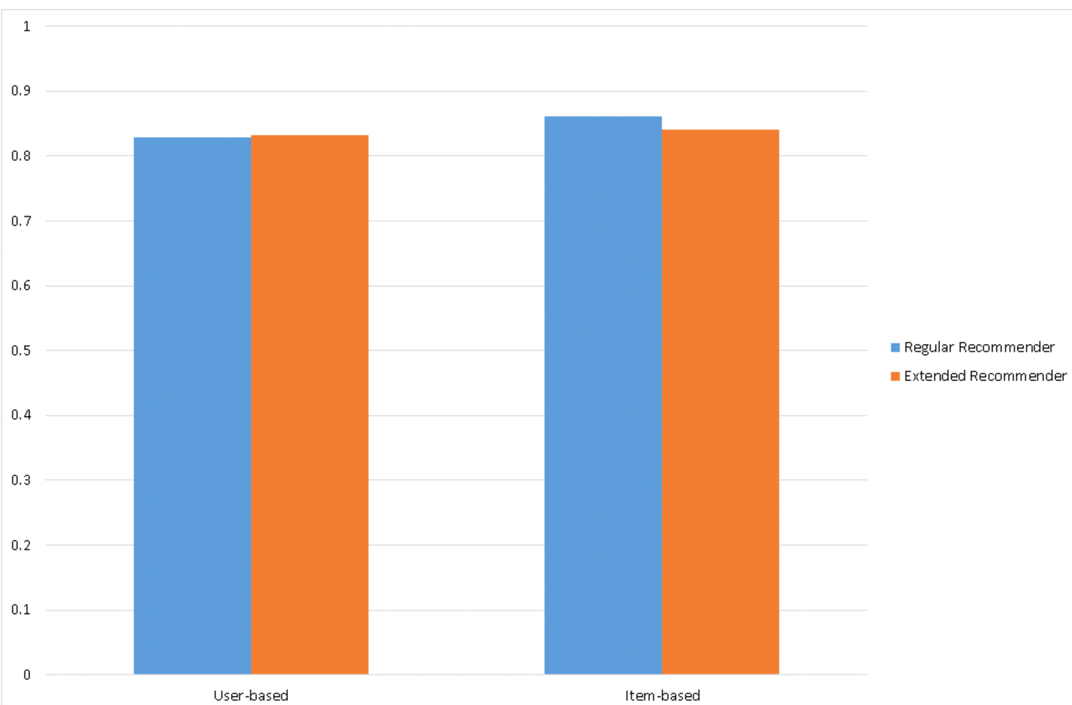

Figure 4.10: Experiment Results for Hybrid Recommendation: 50% - 50%

4.9 Experiment Results for Pre-Clustered Recommendation

In this section, results obtained when users are clustered based on their tags using K-medoids clustering with Jaccard distance are included. Results are depicted in the figure 4.11 and they show that clustering using tags resulted in worse recommendation. Since users are separated based on their tags first, it means that tags are a lot less representative than ratings when user similarity is concerned.

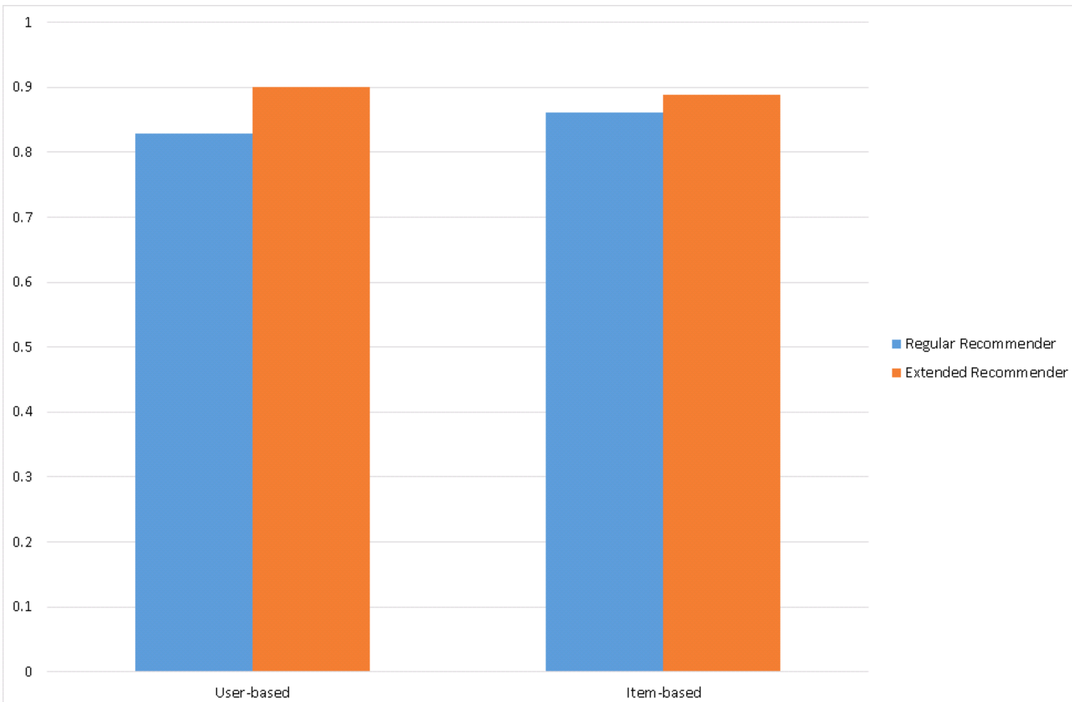

Figure 4.11: Experiment Results for Pre-Clustered Recommendation

4.10 Experiment Results for Higher Tag Dense Data

For this section, data used in experiments are filtered more to ensure that tag density in the data is high enough to affect the results. For each step, data is percolated from previous step and how the accuracy of recommendation changes is observed. Figure 4.12 shows the results when users and movies that have less than four tags associated with them removed. The dimensions of the generated matrix after filtering is 4007×5426 which is quite small and dense compared to original 100k rating and 10M tag data. Next tags that are used less than four times have been removed from previous data whose dimensions are dimensions 2905×4311 and its results are depicted in Figure 4.13. After that tags that do not have any ontology relation with any of the other tags are filtered and experiments conducted with this data gave the results shown in Figure 4.14. After percolations of the tag data, resulted matrix dimensions became 2595×4004. Also for each step of removal, if no tags left for a user or movie, then that rating data are also removed to ensure tag density level.

Table 4.4: Statistics: Users and Movies with at least 4 Tags

Number of Unique Users	588
Number of Unique Movies	1997
Number of Unique Tags	3212

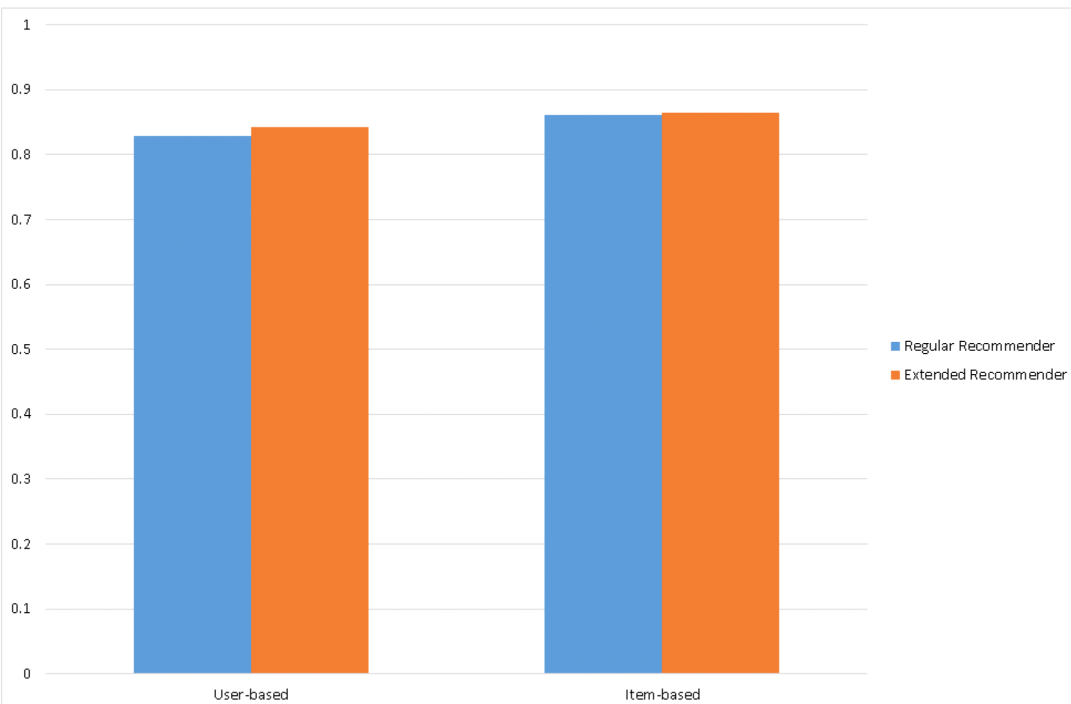

Figure 4.12: Experiment Results: Users and Movies with at least 4 Tags

Table 4.5: Statistics: Tags Used Less Than 4 Removed

Number of Unique Users	588
Number of Unique Movies	1997
Number of Unique Tags	1933

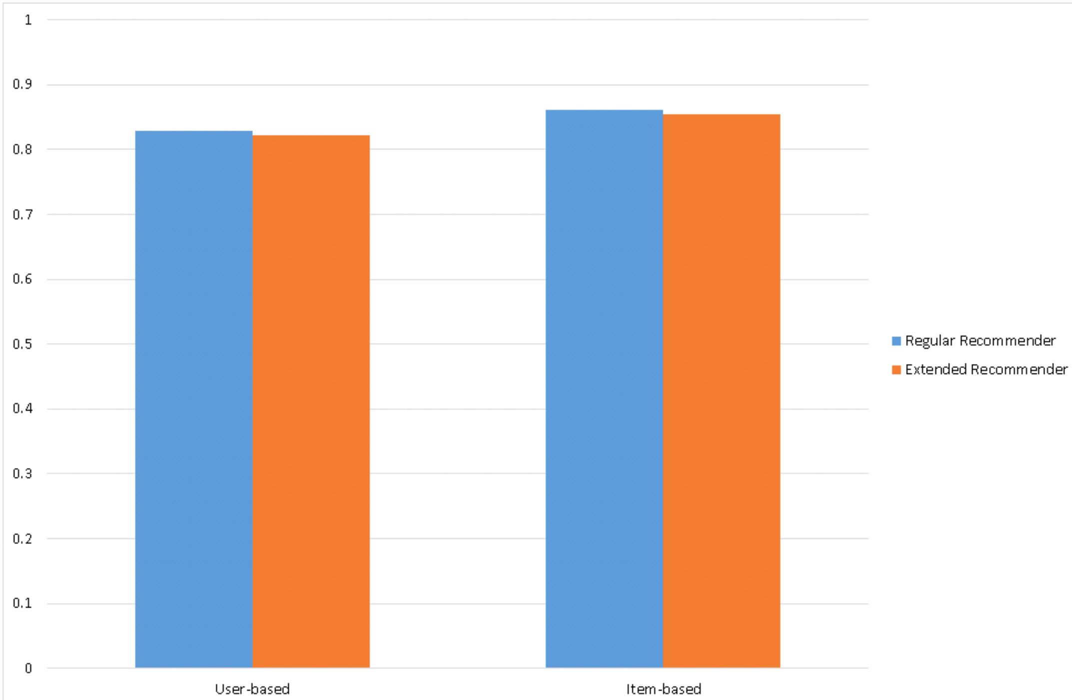

Figure 4.13: Experiment Results: Tags Used Less Than 4 Removed

Table 4.6: Statistics: Tags with No Ontology Relation Removed

Number of Unique Users	588
Number of Unique Movies	1997
Number of Unique Tags	1610

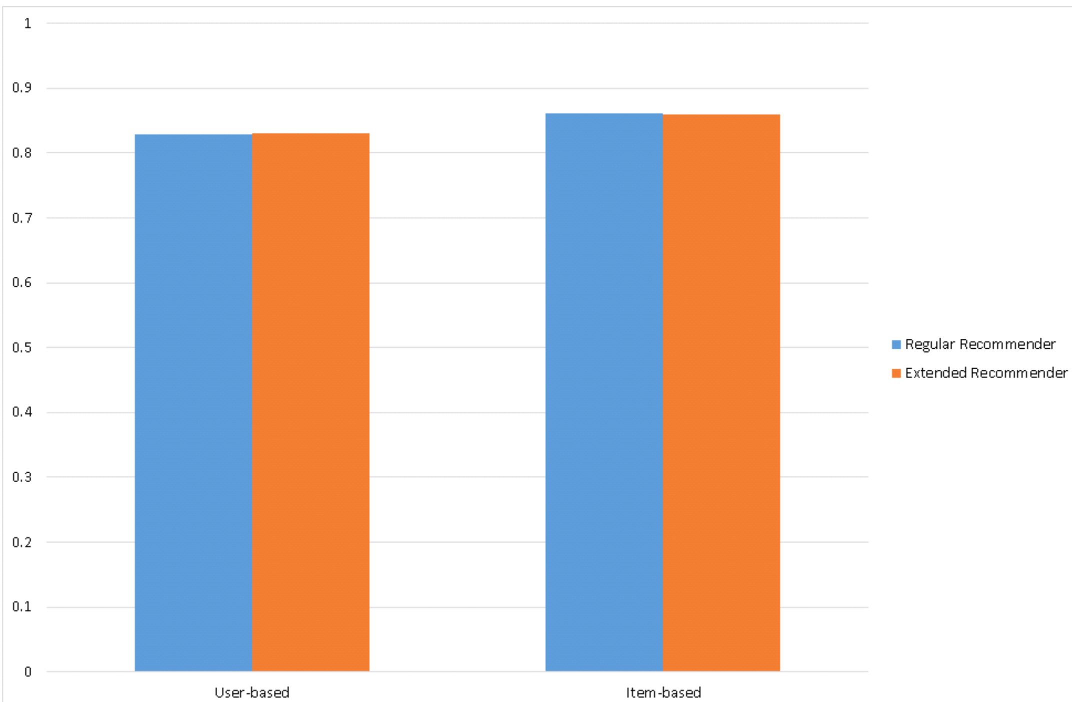

Figure 4.14: Experiment Results: Tags with No Ontology Relation Removed

4.11 Experiment Results for HOSVD-based Recommendation

In this experiment, a 3-dimesional tensor is constructed with rating data, tag data and ontological distances between these tags and HOSVD-based recommendation technique is applied. Results are depicted in the Figure 4.15. There is a slightly improvement in the results, however it is not significant enough to be considered as an enhancement to the process. With the third dimension, the working data become very large that we needed to reduce it in order to complete experiments. This may influenced experiments in a bad way. This technique also suffers from same problems described above such as quality of tag data.

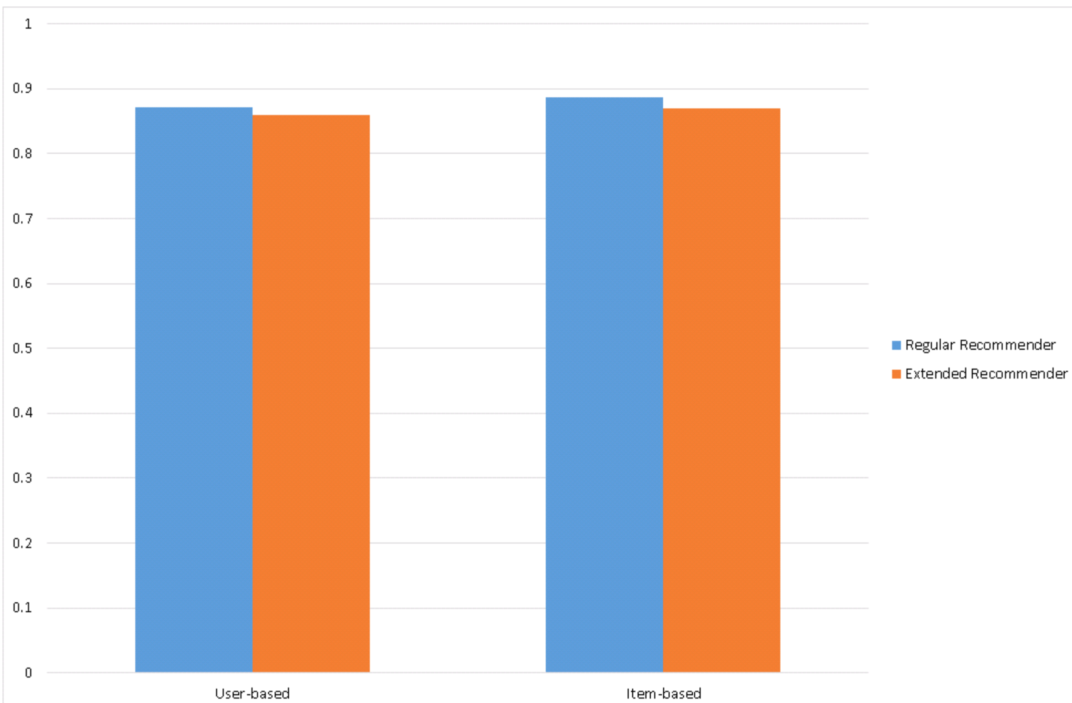

Figure 4.15: Experiment Results for HOSVD-based Recommendation

4.12 Summary of Results

In this section, all results are summarized. Figure 4.16 shows the results of all algorithms when user-based recommendation is performed and Figure 4.17 depicts the results of item-based recommendation for all proposed algorithms.

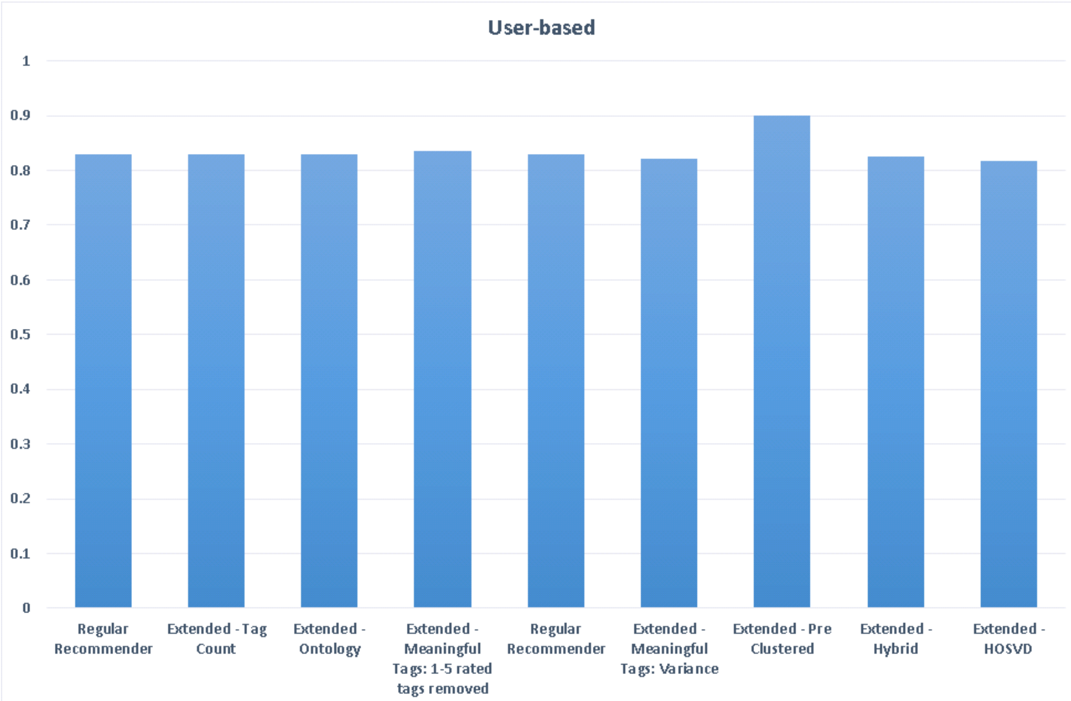

Figure 4.16: Experiment Results: All Algorithms - User-based

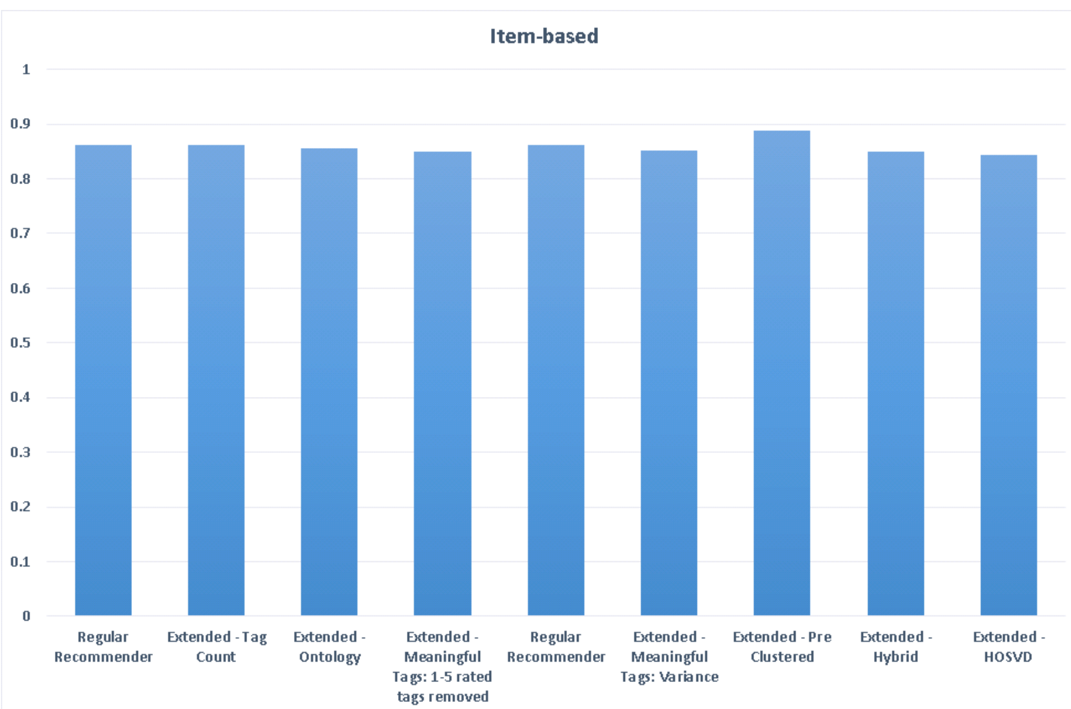

Figure 4.17: Experiment Results: All Algorithms - Item-based

CHAPTER 5

CONCLUSIONS

In conclusion, recommender systems are a hot research topic nowadays. Several online firms, including Yahoo!, Amazon.com and Movie Critic, recommend documents and products to consumers. However, it suffers from performance, scalability and accuracy problems. In this thesis, Extended SVD Recommendation was introduced. Most of the recommender systems do not tags take into consideration while recommendation, however they may give a valuable information, as a result increase recommendation accuracy. We tried to incorporate tags into recommendation process and used ontology to identify relationships between tags. We have compared our algorithms with a reference SVD-based recommendation algorithm. Most of the time, proposed approaches did not perform better, since tags are unstructured and hard to relate with rating information because a wide variety of information is contained in them most of which is irrelevant to user preferences. Also extension of rating matrix with tag data does not actively change the working data resulted in negligible differences in the accuracy of recommendation results. Another problem is that the data contains freely entered tags where no guidance performed while tagging. This might results in lower quality of tag data. Moreover, tags are not as representative as rating data so that their effects are almost negligible. Furthermore quantity of tags compared to ratings are quite small in the used dataset. As a result, methods that try to improve recommendation with the help of tags entered by users and pre-defined ontologies such as WordNet are introduced however they did not produce better recommendations.

REFERENCES

[1] Jaspell :: Java spelling checking package. http://jaspell.sourceforge.net/. Accessed: 2012-05-06.

[2] Mean absolute error. http://en.wikipedia.org/wiki/Mean_absolute_error. Accessed: 2013-12-22.

[3] Movielens. http://grouplens.org/datasets/movielens/. Accessed: 2011-03-12.

[4] Wordnet - a exical database for english. http://wordnet.princeton.edu/. Accessed: 2011-10-12.

[5] A. Y. an Chen and D. Mcleod. Collaborative filtering for information recommendation systems.

[6] M. W. Berry, S. Dumais, G. O'Brien, M. W. Berry, S. T. Dumais, and Gavin. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37:573–595, 1995.

[7] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithm for collaborative filtering. In *Proceedings of the 14 th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

[8] M. Bunge. *Treatise on Basic Philosophy. Ontology I: The Furniture of the World.* Boston, Riedel, 1977.

[9] R. Burke. Knowledge-based recommender systems, 2000.

[10] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, Nov. 2002.

[11] Y. H. Cho, J. K. Kim, and S. H. Kim. A personalized recommender system based on web usage mining and decision tree induction. *Expert Syst. Appl.*, 23(3):329–342, 2002.

[12] D. W. Embley, D. M. Campbell, R. D. Smith, and S. W. Liddle. Ontology-based extraction and structuring of information from data-rich unstructured documents. In *Proceedings of the Seventh International Conference on Information and Knowledge Management*, CIKM '98, pages 52–59, New York, NY, USA, 1998. ACM.

[13] S. P. Gardner. Ontologies and semantic data integration. *Drug Discovery Today*, 10(14):1001 – 1007, 2005.

[14] G. H. Golub and C. F. Van Loan. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[15] T. R. Gruber. A translation approach to portable ontology specifications. *KNOWLEDGE ACQUISITION*, 5:199–220, 1993.

[16] J. Herlocker, J. A. Konstan, and J. Riedl. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Inf. Retr.*, 5(4):287–310, Oct. 2002.

[17] J. L. Herlocker, J. A. Konstan, L. G. Terveen, John, and T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22:5–53, 2004.

[18] S.-L. Huang. Designing utility-based recommender systems for e-commerce: Evaluation of preference-elicitation methods. *Electron. Commer. Rec. Appl.*, 10(4):398–407, July 2011.

[19] V. Ioannis. Semantic similarity methods in wordnet and their application to information retrieval on the web, 2005.

[20] P. Jaccard. The distribution of the flora in the alpine zone.1. *New Phytologist*, 11(2):37–50, 1912.

[21] G. Karypis. Evaluation of item-based top-n recommendation algorithms. pages 247–254, 2000.

[22] L. Kaufman and P. Rousseeuw. Clustering by means of medoids. In Y. Dodge, editor, *Statistical Data Analysis Based on the L1-Norm and Related Methods*, pages 405–416. North-Holland, 1987.

[23] L. D. Lathauwer, B. D. Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl*, 21:1253–1278, 2000.

[24] O. Osmanli and I. Toroslu. Using tag similarity in svd-based recommendation systems. In *Application of Information and Communication Technologies (AICT), 2011 5th International Conference on*, pages 1–4, 2011.

[25] O. N. Osmanlı. A singular value decomposition approach for recommendation systems. Master's thesis, METU, 2010.

[26] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *THE ADAPTIVE WEB: METHODS AND STRATEGIES OF WEB PERSONALIZATION. VOLUME 4321 OF LECTURE NOTES IN COMPUTER SCIENCE*, pages 325–341. Springer-Verlag, 2007.

[27] R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. In *IEEE Transactions on Systems, Man and Cybernetics*, pages 17–30, 1989.

[28] S. Rendle. Pairwise interaction tensor factorization for personalized tag recommendation.

[29] J. Salter and N. Antonopoulos. Cinemascreen recommender agent: Combining collaborative and content-based filtering. *IEEE Intelligent Systems*, 21(1):35–41, Jan. 2006.

[30] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2Nd ACM Conference on Electronic Commerce*, EC '00, pages 158–167, New York, NY, USA, 2000. ACM.

[31] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. pages 285–295, 2001.

[32] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Incremental singular value decomposition algorithms for highly scalable recommender systems. In *Fifth International Conference on Computer and Information Science*, pages 27–28, 2002.

[33] M. SATTARI. A hybrid geo-activity recommendation system using advanced feature combination and semantic activity similarity. Master's thesis, METU, 2013.

[34] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos. Tag recommendations based on tensor dimensionality reduction. In *In RecSys '08: Proc. of the ACM Conference on Recommender systems, 43–50*, 2008.

[35] K. H. L. Tso-Sutter, L. B. Marinho, and L. Schmidt-Thieme. Tag-aware recommender systems by fusion of collaborative filtering algorithms. In *Proceedings of the 2008 ACM Symposium on Applied Computing*, SAC '08, pages 1995–1999, 2008.