A USER MODELING AND RECOMMENDATION SYSTEM BY MEANS OF SOCIAL NETWORKS

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

BY

ALİ KARAKAYA

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

DECEMBER 2014

Approval of the thesis:

A USER MODELING AND RECOMMENDATION SYSTEM BY MEANS OF SOCIAL NETWORKS

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

Prof. Dr. Gülbin Dural Ünver _____
Dean, Graduate School of **Natural and Applied Sciences**

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

Prof. Dr. Nihan Kesim Çiçekli _____
Supervisor, **Computer Engineering Dept., METU**

Examining Committee Members:

Prof. Dr. Ahmet Coşar _____
Computer Engineering Dept., METU

Prof. Dr. Nihan Kesim Çiçekli _____
Computer Engineering Dept., METU

Assoc. Prof. Dr. Halit Oğuztüzün _____
Computer Engineering Dept., METU

Dr. Ayşenur Birtürk _____
Computer Engineering Dept., METU

Gökhan Tüysüz, M.Sc _____
Central Bank of Republic of Turkey

Date: 02.12.2014

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: ALİ KARAKAYA

Signature:

ABSTRACT

A USER MODELING AND RECOMMENDATION SYSTEM BY MEANS OF SOCIAL NETWORKS

Karakaya, Ali

M.S., Department of Computer Engineering

Supervisor: Prof. Dr. Nihan Kesim Çiçekli

December 2014, 58 pages

In this thesis, it is aimed to design a system which builds user profiles to model users' preferences by tracking the activities of the users on social networks. Specifically, Facebook and Twitter are considered as the social networks. The extracted user profiles are used in a recommendation system application. The user data collected from the social networks is enriched with the concepts in Freebase which is an online and public library, and then the enriched data is used to create vector-based and graph-based user models. Content-based, collaborative and hybrid recommendation algorithms that are implemented in this thesis utilize the created user profiles. The suggestions generated by the recommender system are presented to subjects through a survey to evaluate the performance of the user models. Results show that the recommender system using the semantically enriched user profiles provides a high rate of correct suggestions to the users.

Keywords: Social Networks, User Modeling, Semantic Enrichment, Recommendation Systems

ÖZ

SOSYAL AĞLAR YARDIMIYLA KULLANICI MODELLEME VE TAVSİYE SİSTEMİ

Karakaya, Ali

Yüksek Lisans, Bilgisayar Mühendisliği

Tez Yöneticisi: Prof. Dr. Nihan Kesim Çiçekli

Aralık 2014, 58 sayfa

Bu tezde, kullanıcıların sosyal ağlardaki aktivitelerini takip ederek kullanıcıların tercihlerini modellemek için kullanıcı profilleri oluşturan bir sistemin tasarlanması amaçlanmaktadır. Özellikle, Facebook ve Twitter sosyal ağ olarak dikkate alınmıştır. Elde edilen kullanıcı profilleri bir tavsiye sistemi uygulamasında kullanılmıştır. Sosyal ağlardan elde edilen kullanıcı verileri, çevrimiçi ve açık bir kütüphane olan Freebase'deki kavramlarla zenginleştirilmiş ve zenginleştirilen veriler vektör ve çizge tabanlı kullanıcı modellerinde kullanılmıştır. Bu tez çalışmasında tasarladığımız içerik tabanlı, işbirlikçi ve hibrit tavsiye algoritmaları, oluşturulan kullanıcı modellerinden yararlanmıştır. Tavsiye sistemimizce üretilen öneriler, kullanıcılara anket olarak sunularak kullanıcı modellerinin başarımı değerlendirilmiştir. Sonuçlar anlamsal olarak zenginleştirilmiş kullanıcı profillerini kullanan tavsiye sisteminin, kullanıcılara yüksek oranda doğru tavsiyelerde bulunduğunu göstermiştir.

Anahtar Kelimeler: Sosyal Ağlar, Kullanıcı Modelleme, Anlamsal Zenginleştirme, Tavsiye Sistemleri

To my precious family

ACKNOWLEDGEMENTS

I would like to deeply thank my supervisor Prof. Dr. Nihan K. Çiçekli for her valuable supervision, guidance, useful critics and discussions throughout this work. It was a great chance to study with such tolerant, friendly and motivating supervisor.

I am also grateful to my thesis committee members Prof. Dr. Ahmet Coşar, Assoc. Prof. Dr. Halit Oğuztüzün, Dr. Ayşenur Birtürk and Gökhan Tüysüz for their criticism and advices.

I would like to thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) for providing the financial means throughout this study with the project number of EEEAG-112E11.

I am deeply thankful to my friends, Emrah Şamdan, Yunus Emre Işıklar, İbrahim Akçay, Volkan Ak and Zübeyir Hasan Gün for their incentive belief and encouragement about both thesis and real life.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVATIONS

SNS	(Social Network Site)
UM	(User Model)
API	(Application Programming Interface)
SDK	(Software Development Kit)
FQL	(Facebook Query Language)
REST	(Representational State Transfer)
SQL	(Structured Query Language)
TV	(Television)
URL	(Uniform Resource Locator)
IDF	(Inverse Document Frequency)
LOG	(Logarithm operator)
TF	(Term Frequency)
HTML	(Hyper Text Markup Language)
HTTP	(Hyper-Text Transfer Protocol)
TUMS	(Twitter-based User Modeling Service)
MQL	(Meta Query Language)
CRUD	(Create-Read-Update-Delete)

CHAPTER 1

INTRODUCTION

Developing technology and growing accessibility of Web have changed the way people can communicate with each other. A new type of information system, called social networks, is witnessed by millions of users which encourage them to share not only visible items such as photos, videos or documents but also feelings or check-ins.

Social networks are new scientific research area in Computer Science with closely related daily life experience and have huge amounts of data. Social Network Web sites such as Facebook (about 1.28B users), QZone (about 644M users), Google+ (about 343M users), LinkedIn (about 300M users) and Twitter (about 255M users) are highly popular[1]. Estimated 13M transactions per seconds for Facebook, average 58M tweets per day for Twitter and a total of 5 billion photos hosted by Flickr show that social networks are one of the most challenging areas in computer science based on scalability and robustness properties[2].

In this thesis, we propose a user modeling system which shows different aspects of the individuals' interests by processing the traces of the users on social networks. Moreover, a recommendation system is designed to evaluate the performance of the constructed user models.

[1] Number of Users Statistics: http://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/
[2] Social Networking Statistics: http://www.statisticbrain.com/social-networking-statistics

1.1. Motivation

Sociologists have worked on social networks before modern social network sites. Analyzing company organizations by means of social networks between employees [1], defining a social network support for old people derived from theory of social networks [2], and examining social support network based on characteristics of network users and relationships between them [3] are example researches conducted before social network sites have arisen on internet. Since collecting data from users was too hard, the researches were carried on limited networks. The rapid growth of social network sites provides an opportunity to study with large user networks. The huge data proposed by social networks about users can be used in different systems. In reality, each social network site uses their own users' data to help users to present personalized choices for them. Since the amount of information which has been generated is too much and the processed data can be used only in their own domain (i.e. in the same social network site), working on social network sites is still an open area for scientists.

Working on different social network sites and combining their data are challenge for me. I have interested in studying on huge and dynamic user data. The potential for future collaboration in this area is apparent. I will be glad, if users like the items that suggested by our recommendation system.

1.2. Problem Definition and Our Approach

Social Networks and their data are attracted by both academic and industrial institutions. Generally, studies are based on a simple idea which is extracting the relevant data from social networks and using it in suitable areas [4]. In this thesis, we study user modeling with the extracted data from specific social network sites. The created user models can be used in different applications such as recommendation systems.

We have considered only Facebook[3] and Twitter[4] as social network sites in this study. We have chosen these two social network sites because of their popularity. Twitter is an online micro-blogging service that enables users to send or read text based messages called Tweets. In contrast to Twitter, Facebook is a general purpose social site that is more sophisticated than Twitter. Working on these two different types of social network sites provides a great flexibility to create user models. Moreover, in order to get more expressive user models, a general purpose open library Freebase[5] is used. The extracted data is semantically enriched with the help of this library.

User model generation is done with two different methods which are vector based and graph based approaches. Generated user models are tested on an experimental recommendation system that is developed in this thesis. The performances of the created user models and the recommendation system are measured by conducting surveys with a few subjects.

1.3. Contributions

This thesis contributes to the literature in the following ways:

- Research related to Facebook user data is not widespread. We have collected the data of publicly available users with an API provided by Facebook and a web application that we have implemented.
- The collected data is enhanced with an open library with respect to its type. Enhancement style and level are varied based on domain of the data such as 'Film, Music, Television, Sport or Book'.
- We have used a graph database to store the collected data. We have implemented a recommendation module using the functionalities of graph databases in different ways.

[3] https://www.facebook.com/
[4] https://twitter.com/
[5] https://www.freebase.com/

1.4. Thesis Overview

The rest of the thesis is organized as follows:

Chapter 2 introduces the research areas related to the thesis. It includes social network types, user modeling strategies and general information about recommendation systems.

Chapter 3 presents our approach to user modeling and the recommendation algorithms developed in this thesis. We present the modules of our system, including data collection and data enhancement modules, and the structure of the graph database in detail in this chapter.

Chapter 4 presents the surveys conducted within the scope of this thesis. It also discusses the evaluation of the performance of the recommendation system.

Chapter 5 draws conclusions about the thesis work. Some parameters that affect the success of the system are discussed. Possible improvements and future plans about the system are also mentioned.

CHAPTER 2

RELATED WORK

This chapter provides general information about social networks, user modeling and recommendation systems. We discuss the importance of social networks and the techniques that can be used to crawl the social networks to create a dataset. We also review user modeling strategies and the requirements of user models. A brief survey on recommender systems and a comparison of recommendation methods are also presented in this chapter. Finally we discuss semantic enrichment methods that can be applied to enhance user profiles.

2.1. Social Networks and Data Crawling

Social network sites have become a part of daily life recently. They provide people with a fast communication opportunity. There are lots of social networking sites which have different styles and focus on different aspects. A social network site (SNS) is defined as a set of web-based services that allow individuals to

- Create a profile with different visibility levels within a bounded system,
- Share meaningful items with other people or able to see other's sharing,
- View and traverse their list of connections and those made by others within the system [5].

Although SNS's have implemented a wide variety of technical features, they basically consist of profiles which can be visited by other people. However, each

SNS has its own functionalities and target audience. In order to work on a SNS, one must know its type and how/what data can be crawled from it.

2.1.1. Social Network Classification

By means of social networks, millions of people are creating a digital social structure which is made up of nodes (individuals, groups or organizations) and connections. Connections are representing relationships between nodes. Node types and connections have different implementing hierarchies based on the social network type. The main idea is based on a simple principle: on-line socialization.

Figure 1 illustrates a social media classification. Social networking services such as Facebook, Linkedin, are platforms that people build social relations and interact with each other. These sites allow individuals to create a profile and create a friend list which can view and share connections within the site.

Social bookmarking sites allow users to add, edit or share bookmarks of web sites [6]. It is different from storing bookmarks in a local folder. Tagged documents are stored on the Web and can be accessed by other users and different computers. Users can share their bookmarks to other users and they can see the most bookmarked sites.

Due to increasing the number of social web sites, following all news by users are getting hard. Social media news sites such as Digg, Reddit and Slashdot show news and articles and users can vote or comment related to them.

Blogging sites (Twitter, Wordpress, Google Blogger etc.) are web based services that allow users to share and view small elements such as messages or html links that are typically displayed in reverse chronological order. Blogging sites can be defined as online diaries of users. In Blogger the owner of an account controls their own discussion and allows others to comment or question about the topic.

In social network sites, users can not only share textual information but also deliver media files or do real time multimedia communication with other users. Some applications called VoIP software applications provide users to communicate with their friends and colleagues easily. These applications support document sharing, voice messaging, video calling and some other beneficial features.

Users can communicate only with people who are in their contact list with VoIP applications. If they want to share their photos or videos to other people, they can use media sharing services. Media sharing web sites focus on different domains. Users share their videos with YouTube, Vimeo and Veoh and share their photos with Flickr and Instagram.

Document or file sharing over internet is also done by social network sites. Google Drive, Dropbox, iCloud and MediaFire are web sites that provide accessing to digital media by users. In order to share a document, users upload files to the web sites and share created links to peers. The peers can download the content via the link. Document sharing raises security leakages and copyright issues. All sharing services users have to accept a contract that contains warnings related to these issues, before uploading a file.

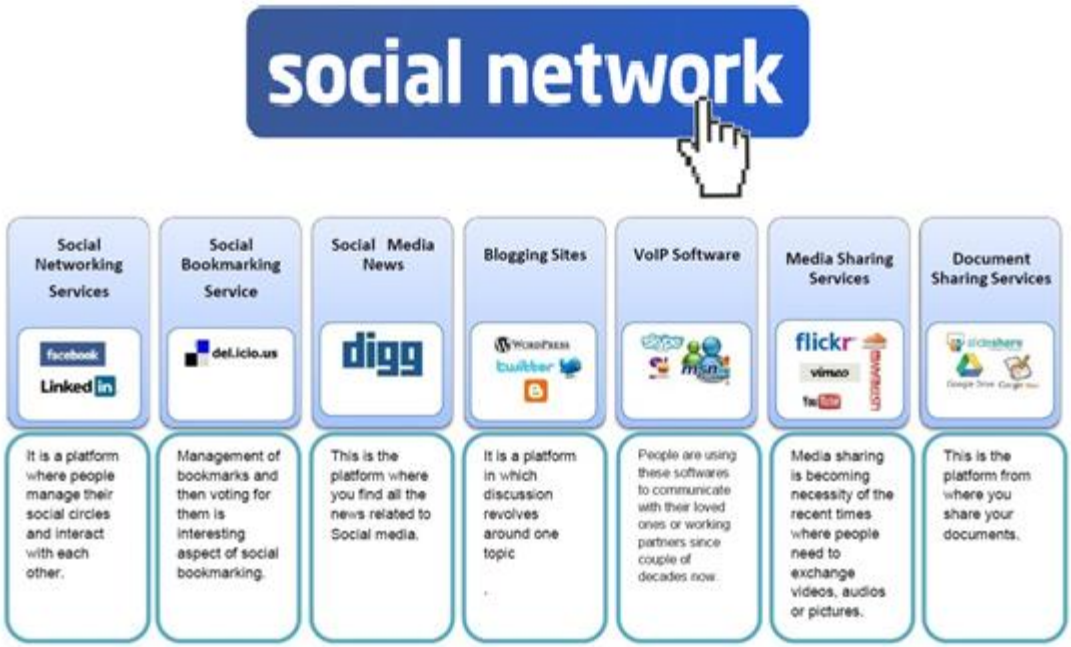

Figure 1: Social Network Classification[6]

2.1.2. Data Crawling from Social Networks

Some social network sites provide some Application Programming Interfaces (API) for developers so that they can implement their own applications. API's are libraries that contain specifications related to variables, data structures and functionality methods.

Facebook provides a general purpose Graph API and REST API. Moreover some third party APIs and SDKs are available to use Facebook functionalities. Facebook Graph API library includes a set of methods that enable reading from or writing to data on Facebook platform. It is called Graph API because the results of queries should protect the consistent view of the Facebook graph with the connections. Facebook graph is formed by nodes or objects (users, photos, events etc.) and their connections between them such as user friendships, tagging photos etc. REST API is

[6] http://whatisdigitalmarketing.wordpress.com/2013/05/27/classification-of-social-media/

a deprecated library that contains methods which use a special language called Facebook Query Language (FQL) based on REST architecture. Facebook wants developers to migrate their REST architecture based applications to new Graph API.

Twitter also provides APIs to developers such as REST API, Streaming API and new Fabric API. REST based API and Streaming API are old but simple and fully functional APIs that contain methods related to programmatic access to read and write Twitter data. Fabric is an advanced API that supports mobile application development and is used to create more stable and flexible applications with extra social features like sharing.

Linkedin, Delicious, Flickr and most of the other Social Network Sites support developers to implement their own applications by the provided APIs.

2.2. User Modeling

A user model is a representation of a user, which contains the captured information about the user for personalization needs. According to a study conducted by Nurmi et al. [5], a user model is created in order to get information such as:

- goals/tasks: what is the user attempting to achieve?
- knowledge/background/experience: what does the user know of the subject? What can we expect the user to know?
- interests: what web pages or songs the user likes (etc.)?
- traits: personality features that can influence the user's behavior and expectations. e.g., introvert or extrovert
- cognitive styles: holist or surrealist (etc.)
- context of work (platform, location, activity)

Created user models can be used in many areas such as search engines, e-commerce, social networks and digital libraries.

According to a survey conducted by Viviani et al., there is a set of requirements that has to be considered in the definition of a good user modeling system [7]. These requirements are independent from which modeling strategies are used. The following requirements are considered:

- Precision: Created models must be precise in order to be advantageous for the users themselves and the applications by the knowledge sharing process.
- Domain independence (Generality): Created models should provide compatibility with as many applications and domains as possible.
- Expressiveness: Models should be able to express many types of actions and rules about the users and their context at the same time.
- Strong inferential capabilities: Models have to be able to resolve conflicts if any contradictions are detected and have to be supporting various types of reasoning.
- Easy integration: Created models must be integrated with other systems easily.
- Quick adaptation: Models have to quickly adapt services to new users, personalization functionalities, applications, and domains.
- Scalability: Created models must support and manage in an efficient way many other applications at the same time.
- Privacy: Models may contain private information about users. User Modeling System has to implement well-defined privacy policies and conventions. System has to enforce concrete policies when using or sharing models through applications.

According to a survey conducted by Xu et al [8], there are four ways to construct user models:

- Traditional Bag of Words (BOW): BOW is a simplest way to create user models. Explicit user interests are listed in a vector. Each user represented as a vector and models are created by using the whole or partial vector [9][10].

- Concepts Based: In addition to the explicit user data, implicit user data is used in modeling. Implicit data also called concepts are obtained from text mining methods, mining patterns or can be external sources such as online libraries [11].
- Tag Based: In this modeling strategy, models can be represented as user-tag-resource-relation quadruple. For multi-resource and highly distributed tags, this modeling strategy generates better models [12].
- Topics Based: This modeling strategy is used in order to represent user interest as topics rather than tags. Instead of simple tags, models contain more comprehensible topics [13][14].

These models are represented as relational matrix or as graph based on data and relation types.

2.3. Recommendation Systems

Recommendation systems are a subclass of information filtering systems, that suggest most relevant items to user [15]. Items can be book/music/film in an e-commerce site, news/event in a news site, or other user/groups in a social web site. The main purpose of a recommendation system is to provide users with personalized services.

Basically, recommendation systems offer most reasonable items in a finite set of items. Generally, the most reasonable item can be the most weighted or closest one based on the representation and implementation. In order to calculate the weight of an item or the distance between items, not only raw item data but also derived item groups or models can be used [16].

Recommendation methods are information agents that try to predict which items out of a large set of items; the target user may be interested in and recommend the most suitable ones to the user. Each technique has its own advantages and disadvantages.

Depending on the domain, the properties of items and the extracted knowledge about users/items, one or more of these techniques can be used in recommendation system.

Basically, recommendation methods can be divided into two sub-categories which are personalized recommendation or non-personalized recommendation [17]. Non-personalized recommendation is identical for each user. The bestselling items for e-commerce sites, the highest average rating for a movie, and the most viewed accounts for a social web site are examples of non-personalized recommendations. Recommendation systems generally do not focus on non-personalized recommendations. Personalized recommendation methods can be divided into three main categories which are content-based, collaborative-based and knowledge-based methods [18]. Moreover, there are some derived methods which are used in recommendation systems such as hybrid approaches, graph-based recommendation etc [19].

2.3.1. Content Based Filtering Methods

Content-based filtering, also referred to as cognitive filtering [20], recommends items based on a comparison between the content of the items and a user profile. Content-based recommender systems make recommendations by analyzing and interpreting the content of the data and finding relationships between them [18]. The content of each item is represented as a set of descriptors or simple terms. The user profile is represented with the same or related terms and it is constructed by analyzing the content of items which have been seen by the user. At this point, the same terms can be found easily. The main problematic part is determining the related terms.

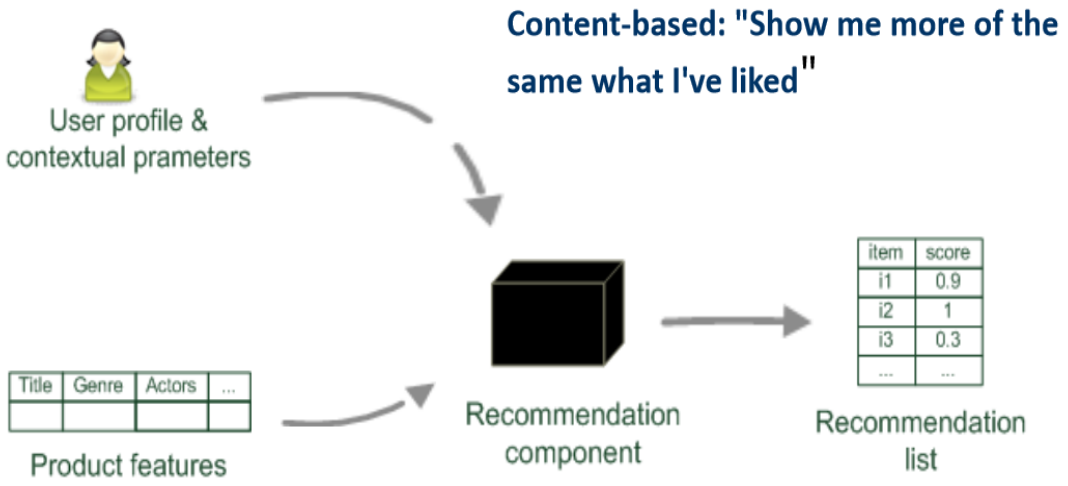

Figure 2: Content-based recommendation [21]

In order to find related terms, there are several methods with respect to the domain. If the domain is a set of documents which contain only words or word groups, term frequency – inverse document frequency ($TF_{(t,\,d)}$ x $IDF_{(t)}$) methods can be used [22]. $TF_{(t,\,d)}$ means a specific term *t* in a given document *d*. It is expected that a term will appear many more times in long documents than shorter ones. Thus, the normalized version of TF is often used.

$$tf(t, D) = \frac{N_t}{L_D}$$

where *tf(t,D)* denotes the term frequency, N_t is the number of terms in the document D, and L_D is the length of the document or total number of words in the document.

IDF means inverse document frequency which is a function shows whether the term is common or rare across all documents. While computing TF, all terms are considered equally important. By multiplying two concepts, the importance of the terms can be calculated more reasonably. The multiplication causes weighting down the frequent terms while scaling up the rare ones. The formula for IDF is:

$$idf(t, D) = \log \frac{N_D}{|\{d \in D : t \in d\}|}$$

where idf(t,D) is inverse document frequency, N_D denotes the number of all documents in the corpus, and the denominator denotes the number of documents with term *t* in it.

For example, the term "the", is a very common English word in documents. Therefore, its Term Frequency is high. On the other hand, since it is observed in all documents, IDF is so low, or 0. Therefore TF x IDF is very close to 0, which means "the" word is not an important term in general purpose documents.

Another content based method for recommendation systems is the decision tree method [23]. The decision tree forms a tree model which maps the input to a predicted item with respect to a user model. According to this representation, arcs from parent node to child node represent an item or a set of items. Interior nodes, including the root node, contain a hypothesis related to the item and subtrees are formed with respect to the hypothesis. The construction of the decision tree starts with a root node and the set of items. A sample decision tree representation is shown in Figure 3. An item or a user model name is assigned to the root and arcs and child nodes for each set of values are created. Then, the set of items divided into child nodes based on attribute values as specified by the arcs. This division process repeats itself recursively, until the count of set of item is 1 or division of set is not feasible [24].

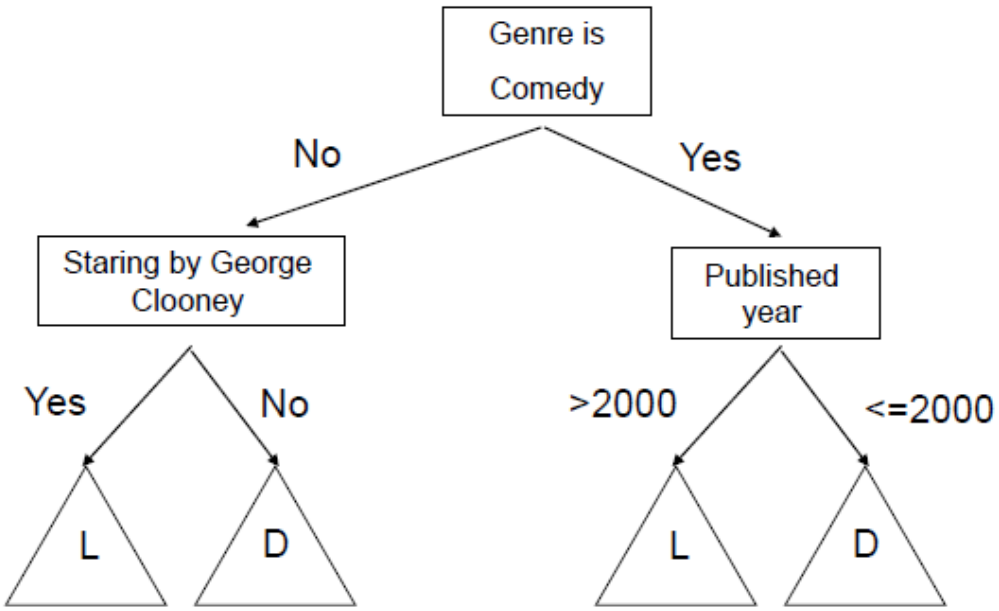

Figure 3: Decision Tree approach of Content-based Recommendation [24]

Cold start is a general problem in recommendation systems for novel users or new items. The system cannot draw any inferences for new users or items about which it has not yet gathered sufficient information [25]. Cold start especially problematic for content-based recommendation systems. In order to solve this problem, hybrid methods which combine content-based and other recommendation methods or semantic enrichment methods can be used.

Synonyms and homonyms are also important problems in information systems. Synonymy is defined as the tendency of the same or similar items to have different names. In a recommendation system the word “unhappy” has to be with the same properties as the word “sad”. The word “miserable” can also be with the same properties. If the words which have same meanings, are stored repeatedly, the performance of the system may reduce. On the other hand, homonymy is used for words that have the same written form but different meaning. For example, the word “Batman” may refer to a book name, a science fiction character, a series or a city in

Turkey. During recommendation phase, the meaning of the word can be very important.

2.3.2. Collaborative Filtering Method

Most of the people rely on recommendations from friends by spoken words, social networks, news etc. People care their friends' opinion since they have similar styles, ideas or manners. Collaborative filtering in a recommendation system is based on this simple idea. If two people have similar past experiences, they might have similar preferences in the future. Collaborative filtering techniques use a database of items obtained from users' past experiences to predict additional topics or products a new user might like.

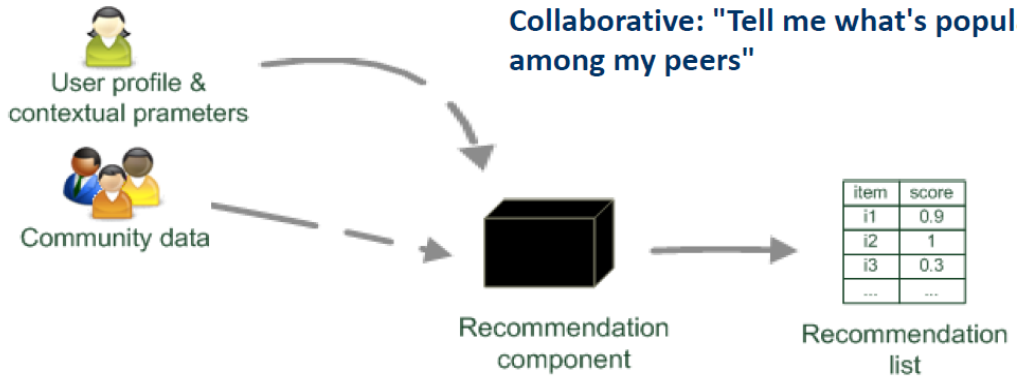

Figure 4: Collaborative-based recommendation [21]

For collaborative recommendation, three methods are generally used, which are clustering approach, similarity based approach, and the decision tree method [26].

Clustering approach can be divided into two subgroups which are user clustering and item clustering. User clustering techniques identify groups (called partitions) of users who are related to same or relevant items. Once the clusters are created, predictions for a target user can be made by averaging the opinions of the other users in that

cluster. Then, target user is pushed into a suitable cluster. The cluster that the target user is pushed is reorganized with respect to the new user and its properties. There may be some users that are a part of different clusters. The main idea dividing users into clusters is shown in Figure 5. In this figure, centers (center1, center2 …) show average properties of the clusters, R_{ij} denotes the rating of the user *i* to the item *j*, a_{ij} denotes the average rating of the cluster center *i* and item *j*. The clustering algorithm may generate fixed sized partitions, or based on some similarity threshold it may generate a requested number of partitions of varying size [27].

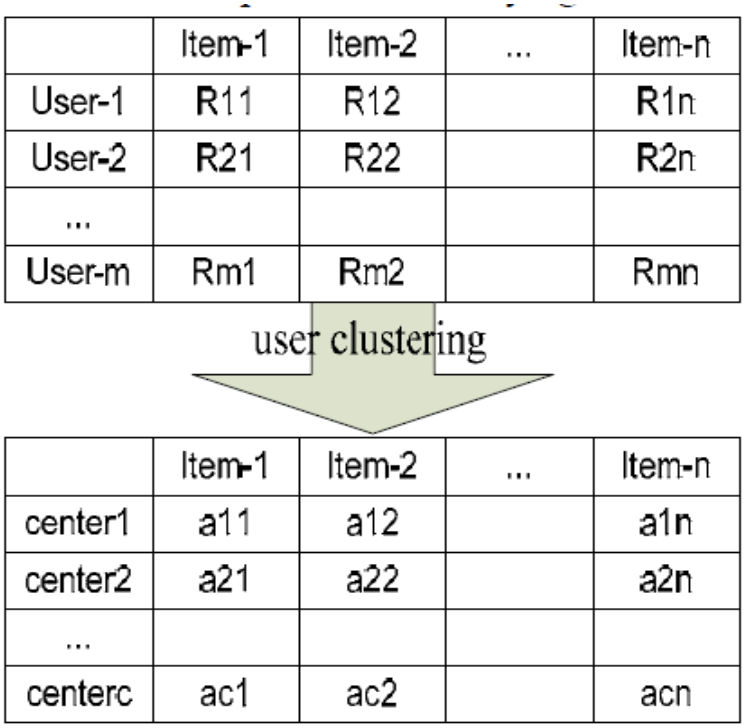

Figure 5: User clustering method for Collaborative Filtering [27]

Similar to user clustering technique, item clustering techniques work by identifying groups of items which are closely related to each other. Clustering is based on averaging of ratings of the items. The prediction of which cluster is suitable for the target item is based on the properties of other items in that cluster. The items that have similar properties, are held in same cluster. The main idea is to divide all items into clusters as shown in Figure 6. In this figure, centers (center1, center2 …) show

average properties of the clusters of items. R_{ij} denotes the rating of user i to the item j; a_{ij} denotes the average rating of the user *i* and item cluster *j*. The clustering algorithm may generate fixed sized partitions, or based on some similarity threshold it may generate a requested number of partitions of varying size.

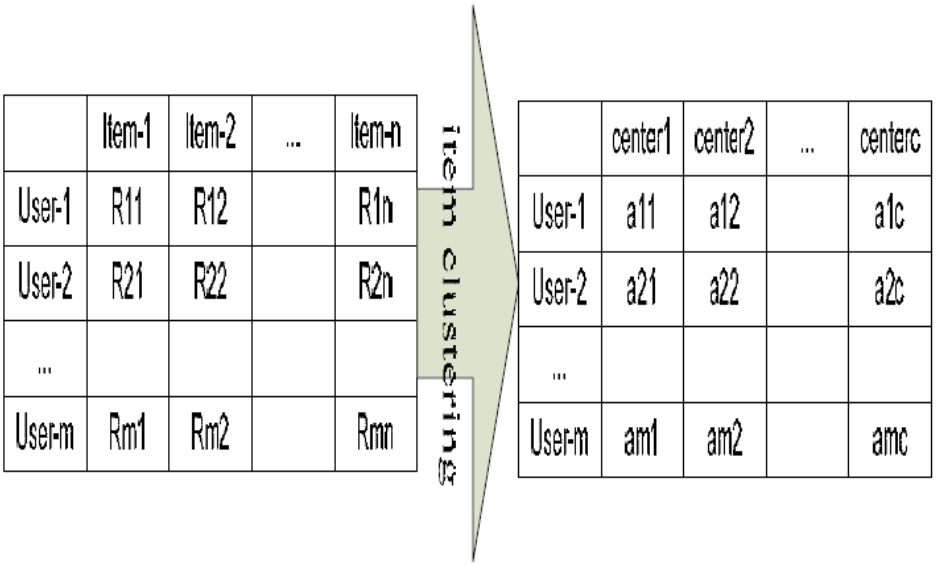

Figure 6: Item clustering method for Collaborative Filtering [27]

Another collaborative filtering method for recommendation systems is cosine similarity [28]. Simply, it is a metric to determine the similarity between two documents. In this similarity method, the words (or items, in case of recommendation) are stored as a vector to find the normalized dot product of two documents (or users, in case of recommendation). The dot product in mathematics is defined as the multiplication of the length of two vectors and the cosine of the angle between them [29]. For Collaborative Recommendation Systems, the dot product can be defined by using the matched items. If two users do not share any item, the angle between them is 90^o (vectors are perpendicular) which causes the resulting dot product value of 0 (since $\cos(90^o) = 0$). Similarly if a user vector is a sub vector of other vector (vectors are parallel), dot product value of maximum will be achieved (since $\cos(0^o) = 0$).

Decision Tree method is also used for collaborative based recommendation system. Breese et al. [23] use this method to create a collaborative recommendation system. A dedicated decision tree is built for each item according to user feedback. The arcs for recommended items can be predicted from the known items and attributes [24]. The method is collaborative since similar items based on users' feedbacks are collected in same subtree. Figure 7 shows an example of a decision tree with some films. In the figure, the related films (i.e. films that are liked by users) come together in the right sub-trees.

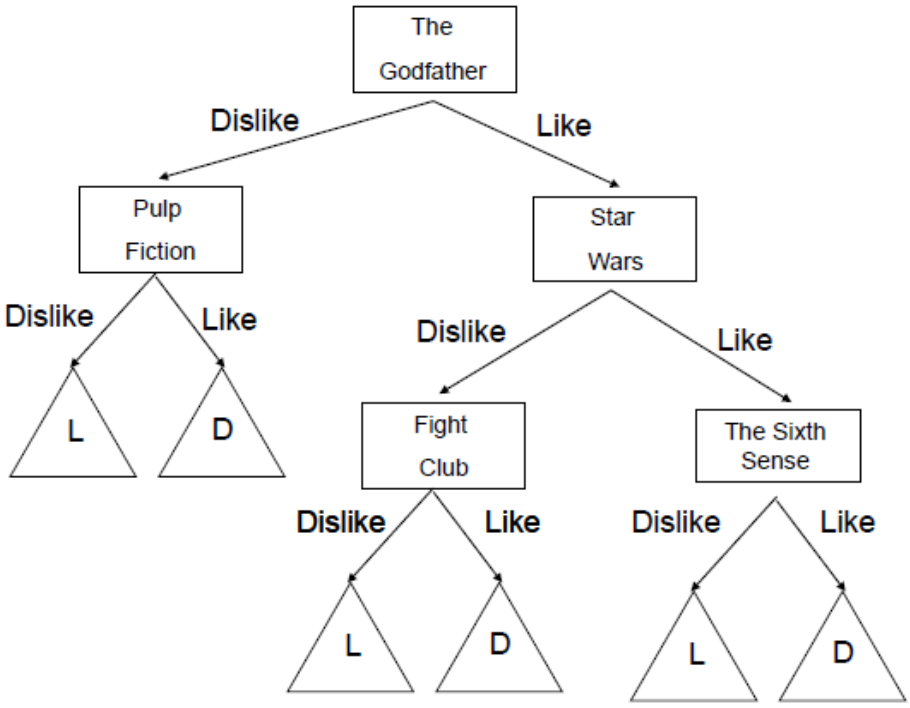

Figure 7: Decision Tree approach of Collaborative-based Recommendation [24]

Sparsity is an important problem in the collaborative filtering approach. If we have too many items in an item cloud, most of the ratings in the user–item rating matrix will be 0. Due to same reasons as in the content based approach, cold start [18], synonymy and homonymy [26] are also important problems of collaborative based recommendation systems.

2.3.3. Knowledge-based Recommendation

A third type of a recommender system is knowledge-based recommendation systems which use knowledge about not only users but also items to generate a recommendation list by reasoning about what items meet the user's requirements. The system suggests items based on inferences about the user's needs and preferences. Inferences can be a case based reasoning system, an adapted similarity metric or a part of an ontology [30]. Knowledge-based systems do not depend on user ratings. For example, a restaurant recommender system called as Entree [31] keeps track of user "knows" and "likes" to recommend restaurants in a new city. Users able to navigate by stating their preferences with respect to a given restaurants and the system refine their search criteria. The Entree system eliminates unrelated restaurants by users' past search criteria. The recommendation method is generally used by e-commerce sites. Content-based and collaborative based recommender systems may suffer due to sparsity and cold start problems. Knowledge based systems do not have to gather information about user tastes. This characteristic feature of knowledge based systems makes them not only important systems on their own, but also highly preferable complementary systems to other recommendation systems [31].

2.3.4. Graph-based Recommendations

Graph based recommender systems recommend items based on not only properties of items but also the connectivity between them [32]. Users and items are represented as nodes and relations. Relations between user-user, item-item and user-item are represented with weighted or unweighted bidirectional or unidirectional edges. Although edges can be built between all possible nodes, they are generally formed between users and items. The edges show relations between users and items and they are used for collaborative filtering. A graph based approach named Adsorption conducted by Baluja et al., tries to find unlabeled nodes from labeled and unlabeled nodes via random walk method [33]. In this research, unlabeled nodes are found by

collaborative filtering in terms of propagating the labeled information by randomly moving between nodes.

A study conducted by Mirza [34], proposes the calculation of average path length to be used in graph based recommendation systems. Path length is simply defined as the calculated distance between two nodes. There may be more than one path available between the nodes. This time, the minimum cost (distance) path is assumed to be the path length. The average path length is the mean of the shortest path lengths over all node pairs. In this thesis, users and items are stored as nodes. The relations between them are represented as edges. In our implementation, we use this approach to calculate the distance between users. We calculate the distances between all users and this measurement is a good indication whether the target users are similar or not. After the calculation of the distance, the recommendation set is prepared based on the idea “similar users may like same things”.

2.3.5. Hybrid Methods

Hybrid recommendation systems are information systems that are formed combination of recommendation techniques to achieve better performance [35]. All of the recommendation techniques have strengths and weaknesses, and many different ways to combine recommendation techniques. Burke et al [35], classified hybrid recommendation systems to the following sub categories:

- Weighted hybrid: Each recommendation techniques have constant weight and recommendation set is filled with respect to these weights.
- Switching hybrid: Based on criteria, past experience or performance issues, recommendation set is created with switching defined simple recommendation systems.
- Mixed hybrid: The recommended items of different systems are merged and ranked to create a unique list. The issue of the system is how the ranking

should be done. The cardinality of simple recommendation system or adding each rank score are used to rank items.

- Feature combination hybrid: Main recommendation system uses as a source other simple recommendation component to generate single recommendation list.
- Feature augmentation hybrid: The system is similar to feature combination hybrid but different in that source recommendation system generates new items to feed main recommendation system.
- Meta-level hybrid: The simple recommendation system prepares the data for the latter complete recommendation system.

In these systems, simple recommendation system pretends as a part of main recommendation system.

In another research conducted by Öztürk and Çiçekli, Baluja's Adsorption algorithm is extended with content based approach. This work focuses on video recommendation [36]. Also, Phuong et al. use the graph based approach for combining the content based and collaborative recommendation approaches. They have proved that their graph based hybrid approach outperforms pure collaborative filtering and pure content-based filtering methods [37].

In this thesis, we have also used a hybrid method as a combination of content-based and collaborative based recommendations. We store items together with their semantic enrichments in the database. Therefore we have not only items that directly come from the user profiles but also similar items obtained from Freebase. This property provides us with a way to use content-based recommendation. Collaborative filtering approach is implemented by forming clusters of users.

2.4. Semantic Enrichment

Semantic enrichment aims to enhance data by adding additional information such as new tags, new relations or new categories to enhance its meaning. According to a research published by Heflin and Hendler [38], without semantically enriched content, the Web cannot be used in its full potential. Many studies in the literature have used different methods to enrich the content. For instance, Fabian et al. [39], apply semantic enrichment to tweets. They use a library called OpenCalais[7] to detect and identify many different entities such as persons, events or products. Then the extracted entities are enriched and linked to news web URL's.

In another research conducted by Abel et al. [40], a people modeling service called U-Sem context is enriched semantically by adding click data in the social networks. By adding this enrichment, the U-Sem system monitors user activities. Moreover, the created models generated by U-Sem are also enriched with Dbpedia[8] library based on the corresponding entities.

[7] OpenCalais Project: https://www.drupal.org/project/opencalais

[8] Dbpedia – Wikipedia based online library http://wiki.dbpedia.org/About

CHAPTER 3

USER MODELING AND RECOMMENDATIONS

3.1. System Architecture

The overall architecture of our system is shown in Figure 8. The system consists of four main modules which are responsible for different tasks. The data collection module gathers data from Facebook and Twitter with API support. In order to enhance the collected Facebook data, the enhancement module is used. In this module the collected data is semantically enhanced by Freebase library. The collected and enhanced data are stored in a graph database named Neo4j[9]. Using these data, user models are generated by the user modeling module. The created user models include both raw data from Social Network Sites and derived data with the support of Freebase Library. A recommendation system that uses content based, collaborative based, hybrid and graph based approaches is implemented to evaluate the success of the user models.

[9] Neo4j – A graph based database: http://neo4j.com/

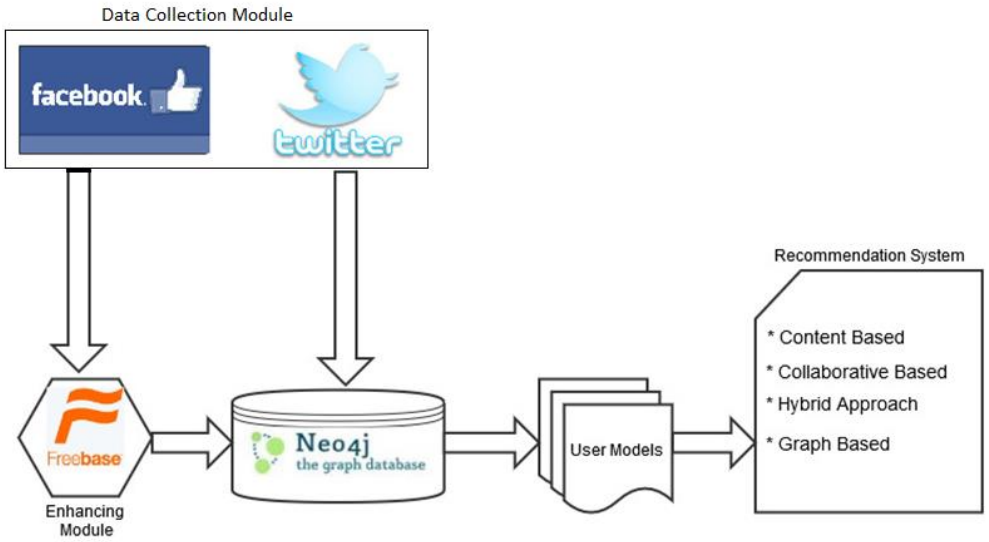

Figure 8: Overall System Architecture

In the following, we describe these modules in detail.

3.1.1. Data Collection Module

The data collection module is responsible for gathering user profile data from social network sites which are Facebook and Twitter. Facebook data is collected via Facebook Graph API and a web based application that we have implemented.

Facebook Graph API is the first and well known way to get data in and out of Facebook‘s social graph. It contains low level HTTP based methods to run queries. In our implementation, we get user page likes via Graph API. Publicly available users’ likes are crawled with queries. A sample result of a query is as follows:

```
{ "data": [
  {"category": "Movie", "name": "Focus","id": "1431466237123842"},
  {"category": "Producer", "name": "Overbrook Entertainment", "id": "50569073841"},
  {"category": "Actor/director", "name": "Queen Latifah", "id": "90061353027"},
  {"category": "Musician/band", "name": "Trey Smith", "id": "268901369814097"},
  {"category": "Actor/director", "name": "Jaden Smith", "id": "151182238257632"},
  {"category": "Musician/band", "name": "Willow Smith", "id": "110281552358202"},
  {"category": "Actor/director", "name": "Jada Pinkett Smith", "id": "51346591319"} ] }
```


Figure 9: A Sample Result

Facebook Graph API limits the query results and does not provide all of page like information. Therefore we implemented a web based application that brings page likes via HTTP connection. This application gathers page likes of users as a real user's view from a web browser. The application logins to Facebook as a regular user and sends a request via HTTP connection to see page likes of the user. Then Facebook server sends the response as a HTML file that contains page likes. A sample page like information of a publicly available user is shown in Figure 10.

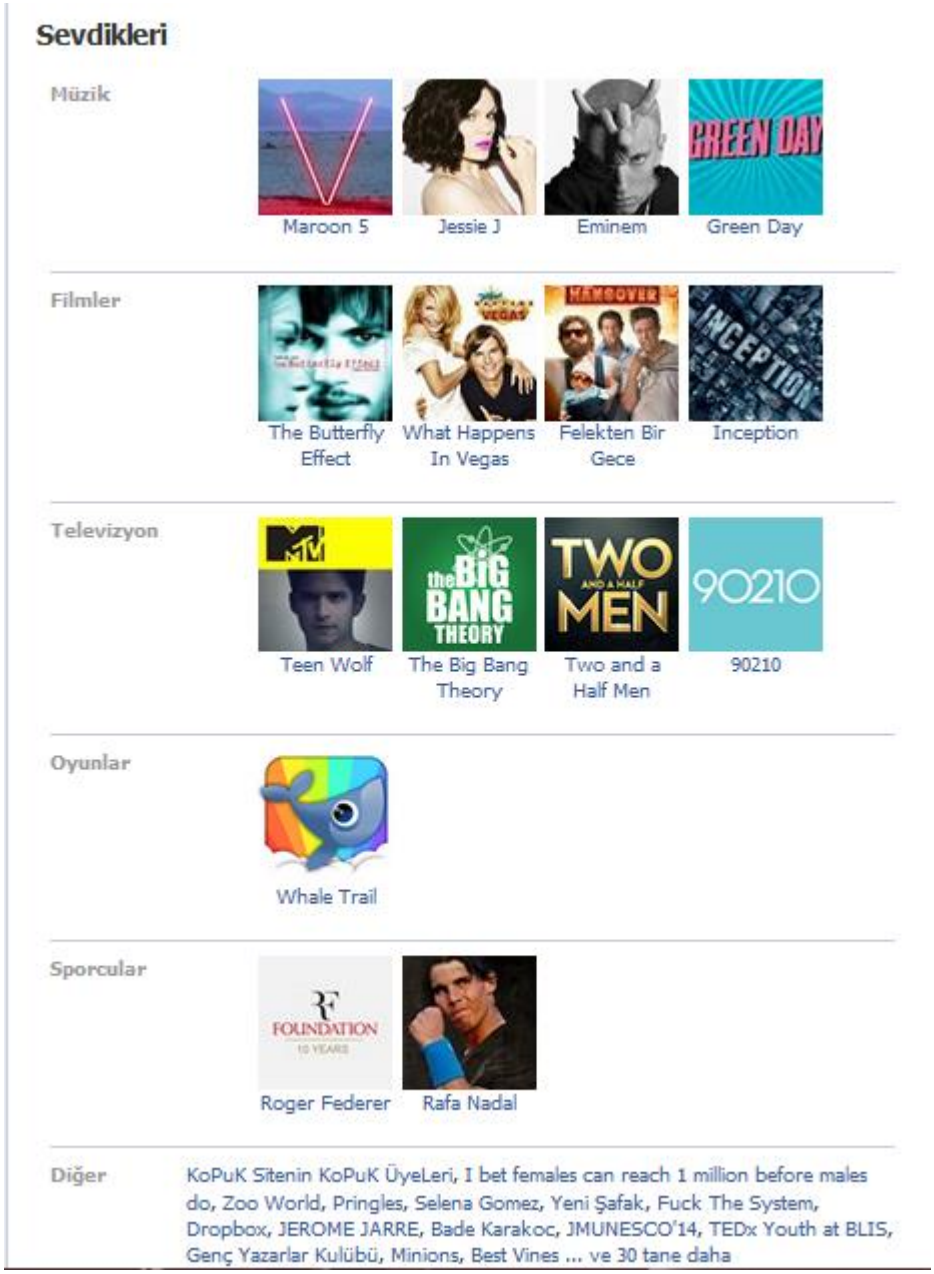

Figure 10: Likes Of a Publicly Available User

The web application parses the web page and returns page likes with category information as in Figure 11. The system sends them to the data enhancer module and saves in the database.

```
Sevdikleri---

Müzik:Maroon 5---Jessie J---Eminem---Green Day---Queen---
The Beatles---Coldplay---P!nk---Bruno Mars---Eminem---
LMFAO---Black Eyed Peas---Adele---Manga---Michael Jackson

Filmler:The Butterfly Effect---What Happens In Vegas---
Felekten Bir Gece---Inception---Sherlock Holmes---
27 Dresses Movie---The Hunger Games---Grease---
Pitch Perfect---Bridesmaids---Easy A---
The Devils Wears Prada---Troy---Indiana Jones---
Karayip Korsanları---

Televizyon:Teen Wolf---The Big Bang Theory---
Two and a Half Men---90210---
South Park---Family Guy---MythBusters---
New Girl---Suburgatory---What''s new, Scooby Doo?---
Scooby Doo Gizem Avcıları---Yalan Dünya---2 Broke Girls---
Dedikoducu Kız---Muhteşem Yüzyıl---Memphis Beat---Lost---
Nigahiga---

Kitaplar:Diary of A young Girl---

Oyunlar:Whale Trail---

Sporcular:Roger Federer---Rafa Nadal---
```

Figure 11: Results Obtained From Data Collection Module

The profiles of Twitter users are gathered via Twitter-based user modeling service abbreviated as TUMS [41]. This service takes a twitter username as an input and returns entities and hashtags related to the user's tweets. It also shows the weight of the terms calculated from user tweets. The data collection module calls the service for each user, and the TUMS returns a JSON object file that contains entities, hashtags and calculated weight of them. Then the JSON file is parsed to obtain hashtags and entities. The twitter username and all parsed hashtags and entities are inserted to the database and relations are constructed between user node and parsed-item nodes. A part of the sample result is shown in Figure 12.

```
<wi:preference>
  <wi:WeightedInterest>
    <wi:topic>Turkey</wi:topic>
    <wo:weight>
      <wo:Weight>
        <wo:weight_value rdf:datatype="&xsd;double">
          0.07457627118644068</wo:weight_value>
        <wo:scale rdf:resource="&tums;Scale"/>
      </wo:Weight>
    </wo:weight>
  </wi:WeightedInterest>
</wi:preference>

<wi:preference>
  <wi:WeightedInterest>
    <wi:topic>Istanbul</wi:topic>
    <wo:weight>
      <wo:Weight>
        <wo:weight_value rdf:datatype="&xsd;double">
          0.027118644067796612</wo:weight_value>
        <wo:scale rdf:resource="&tums;Scale"/>
      </wo:Weight>
    </wo:weight>
  </wi:WeightedInterest>
</wi:preference>


<wi:preference>
  <wi:WeightedInterest>
    <wi:topic>CNN</wi:topic>
    <wo:weight>
      <wo:Weight>
        <wo:weight_value rdf:datatype="&xsd;double">
          0.0033898305084745763</wo:weight_value>
        <wo:scale rdf:resource="&tums;Scale"/>
      </wo:Weight>
    </wo:weight>
  </wi:WeightedInterest>
</wi:preference>
```

Figure 12: Twitter Data Collection Result

3.1.2. Data Enhancer Module

Some user profiles do not contain enough information to model the user. For example, if a user model contains very limited music performer items, then our system cannot draw any inference about music recommendation. In order to populate

data, our system query some important items related to the data. This module uses Freebase[10] library. Freebase provides an API and a special programming language called MQL.

The enriching strategy and its level depend on data types. Table 1 shows how data is enriched with different properties. Before the enhancement, we only know the name of the data and its type. Using Freebase, the system detects item category of the data. Based on data type and item category, the system creates MQL queries to get Level-1 items. Then the system creates new queries to get Level-2 items with using results of Level-1 queries.

Table 1: Enhancement Table

Data Type	Item Category	Enrichment Level 1	Enrichment Level 2
Film	Movie	Actors	Other Movies of the Actors
		Director	Other Movies directed by the Director
		Genres	
		Same series movies if the film is a member of series	
	Actor	Other Movies of the Actors	
	Director	Other Movies directed by the Director	
Music	Performer	Genres	
		Albums of the performer	Songs in the albums
		Coperformers	
		Group members if the performer name is a band	
	Song	Performer of the song	Other song of the performer
		Album name	Other song of the album
		Genres	
Book	Book Name	Author of the book	Other books written by the author
		Subject	
		Genres	
		Same series books if the book is a member of series	
		Film name if book projected to cinema	
	Author	Books of the author	
		Nationality of the author	
		Genres	
Sport	Sport Name	Famous teams if the sport is team game	Famous sportmans of the teams
		Famous sportmans	
		Global organizations related to sportname	
	Athlete	Sport name related to athlete	Famous sportmans related to sport name
		Teams of the athlete	
	Coach	Sport name related to coach	
		Teams of the coach	Famous sportmans of the teams
		Nationality of the coach	
Television	Program	Genres	
		Actor names	Other programs of the Actors
		Category	
	Actor	Other programs of the Actors	

[10] Freebase – A community-curated database: https://www.freebase.com/

For example, if a user likes a movie, then the actors of the movie, other movies of those actors, genres of the movie, the director of the movie, movies of this director are obtained from Freebase and inserted to the graph if they do not exist already. In case of a new node insertion, only new connections to the existing nodes are established.

Freebase's special Meta Query Language (MQL) allows developers to get data programmatically. For example, if a user likes a book called "Ignited Minds", the data collection module handles it, and then it sends the book name to the data enhancer module. The data enhancer module sends an MQL query via category information as shown in Figure 13. This MQL query only returns subjects of the book. For all cells in table 1, data enhancer module sends a query to get data from freebase and returned objects are inserted to the graph database.

```
{
"name": "Ignited Minds",
"type": "/book/book",
"/book/written_work/subjects": []
}
```

Figure 13: A Sample Query in Freebase

Freebase returns the result of the query as in Figure 14.

```
{
"result":
{
"name": "Ignited Minds", "type": "/book/book",
"/book/written_work/subjects":
["Asia", "India", "History", "Patriotism"]
}
}
```

Figure 14: Result of the Query

The result shows that, the subjects of the book "Ignited Minds" are "Asia, India, History and Patriotism". After the query is executed, the nodes "Ignited Minds", "Asia", "India", "History" and "Patriotism" are created and inserted to the database if they do not exist already with equal weight.

3.1.3. User Modeling Module

In our implementation, the data directly gathered from social networks and the enhanced data by means of Freebase library are stored in a graph database called Neo4j[11]. This module is responsible for creating models using the stored data. We have implemented two strategies for modelling users, which are vector based and graph based strategies.

3.1.3.1. Vector Based User Modeling

In this modeling strategy, each user is represented as a vector that contains a set of items related to the user. The profile data that directly comes from the social networks and two level depth data are represented in the vector. Each vector element also contains the source of the data whether it is derived or not. The vector contains thousands of data relevant to the user. A sample vector is shown in Table 2. Depth 0 elements are directly extracted from user likes. Depth 1 elements show data related to depth 0 elements. For example, the director and actors of the film "The Shawshank Redemption" are depth 1 elements. The films that are directed by "Frank Darabont", who is the director of "The Shawshank Redemption", and the films acted by "Tim Robbins", who is an actor of the film, form depth 2 elements.

[11] Neo4j – A graph based database: http://neo4j.com/

Table 2: A Sample Vector Based User Model

Item Name	Depth	Type	Item Name	Depth	Type
The Shawshank Redemption	**0**	**Film**	Thanks for Sharing	2	Film
Frank Darabont	1	Director	Green Lantern	2	Film
Tim Robbins	1	Actor	Cinema Verite	2	Film
Morgan Freeman	1	Actor	City of Ember	2	Film
Bob Gunton	1	Actor	The Lucky Ones	2	Film
William Sadler	1	Actor	Noise	2	Film
Clancy Brown	1	Actor	Tenacious D in The Pick of Destiny	2	Film
The Green Mile	2	Film	Catch a Fire	2	Film
The Majestic	2	Film	Zathura	2	Film
Fahrenheit 451	2	Film	Embedded: Live	2	Film
The Mist	2	Film	The Secret Life of Words	2	Film
Buried Alive	2	Film	War of the Worlds	2	Film
The Woman in the Room	2	Film	Anchorman: The Legend of Ron Burgundy	2	Film
Maroon 5	0	Music	Holiday Gift	1	Album
Songs About Jane	1	Album	Goodnight Goodnight	1	Album
This Love	1	Album	Never Gonna Leave This Bed	1	Album
If I Never See Your Face Again	1	Album	Daylight	1	Album
Won't Go Home Without You	1	Album	Love Somebody	1	Album
Give a Little More	1	Album	The Tears Of Medusa	1	Album
Is Anybody Out There	1	Album			

3.1.3.2. Graph Based User Modeling

Neo4j database stores users and their related items as nodes and define their relations as edges. In this modeling strategy, user-centered graph shows user's likes and enhanced items connected to the likes. The items that directly come from user "likes" have depth 1 from the user node. For example, films that are liked by a user are represented with 1 level depth edges. Derived information that comes from Freebase library such as director, actors or genres of the films is connected to the film nodes. Models are obtained by graph traversal algorithms. In order to traverse the graph, initially all possible outgoing edges are detected from a user node. Then, similarly all outgoing edges connected to the nodes are inserted into the model. The insertion of nodes continues until the user node is reached. If a node marked as user node, then the insertion operation is done to this outgoing node and the operation backtracks to

other outgoing edges. When all outgoing edges reach a user node, creating the model is done for the user.

3.1.4. Database Details

Users and directly or indirectly obtained items are stored in Neo4j as nodes. User nodes do not connect to other user nodes. Items can be connected to either user nodes or other item nodes. Edges between nodes represent relations based on data. The edge between a user and an item represents “LIKES” relation. Item-item edges can be “MUSIC PERFORMED”, “DIRECTED”, “ACT PERFORMED”, “PLAY” etc. based on the type of the item. A very limited snapshot is shown in Figure 15. The orange node shows the ROOT element. The root element is a virtual node which is the ancestor of all user nodes. If we want to insert a new user, then we describe it as a child of the root element. The purple nodes represent users and blue nodes represent items. The edges between users and items, and the edges between items and items show relationship between them.

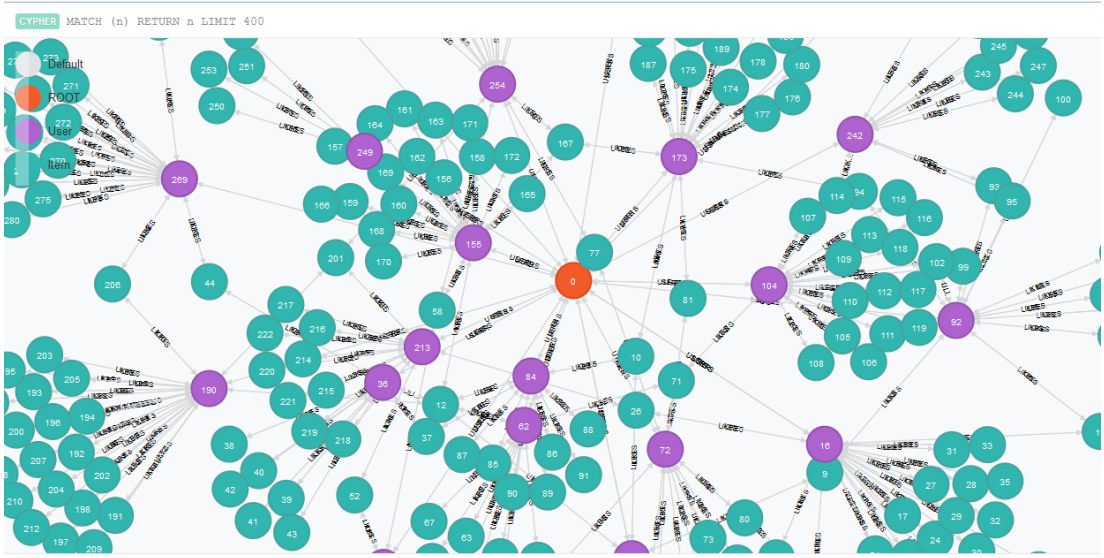

Figure 15: A Snapshot from the Database

Since our database is not a standard relational database, we cannot use SQL or other similar languages for CRUD[12] operations. A special language called Cypher Query Language is used to insert or select nodes in Neo4j. This language focuses on *what* to retrieve from a graph, not *how* to retrieve it. A java based API that supports Cypher language is used in our implementation.

Cypher language differs from native SQL with some properties. Its structure is borrowed from SQL, i.e. the queries are built up using clauses. A sample SQL query which returns items that are liked by user "Derya" is shown Figure 16. In the same figure, equivalent cypher query is also shown.

```
SELECT "Item".*
FROM "User"
JOIN "Item" ON "User".name = "Item".userName
WHERE "User".name = 'Derya'

MATCH (user:User { name: 'Derya' })-[:likes]->(item)
RETURN item
```

Figure 16: Equivalent SQL and Cypher Queries

Keywords *Create* and *Create Unique* are used to create new nodes. Keywords *Match, Optional Match, Start, Foreach* are used to get results similar to native SQL's *Select* word. *Merge* and *Set* methods are used to update graph nodes and edges. Similar to native SQL, some keywords such as *Order By, Union, Where* and *Limit* are also used in Cypher Language. In our implementation, we used Cypher Language to create, retrieve and update operations in the database.

User Clustering Algorithms we have designed and implemented a recommender system for Facebook users in order to evaluate the success of both the vector-based and graph-based user models. In our implementation, the recommender system is designed based on clustering approach where similar users are placed in the same cluster. The number of clusters and the number of users in a cluster dynamically change during calculations. Users are clustered differently for vector based and graph

[12] The acronym CRUD refers to Create, Read, Update and Delete operations in persistent storage.

based approaches. For each user, the set of items which are connected to the user in the same cluster are recommended.

3.1.4.1. Clustering Vector Based User Models

In this approach, the distance between all vectors is calculated with cosine similarity. All calculated and normalized distances are stored in a two dimensional matrix. Then, the most similar two users (the maximum value in the matrix) are stored in the first cluster. Then, a new vector is created for the cluster which includes all the items of the users in the same cluster. The vector representing the cluster is added to the matrix as a new user. The distances are recalculated with the new user. The procedure continues with new distances until every user is assigned to a cluster. The pseudocode of the process is shown in Figure 17.

```
PROCEDURE Calculate DistanceMatrix
Input (vector u1 ... uN)
FOR X = 1 to N
    FOR Y = 1 to N
        DistanceMatrix[X][Y] <- cosineSimilarity (uX,uY)
    END FOR
END FOR


PROCEDURE Find Clusters
Input(DistanceMatrix[N][N])
ElementCount <- N
REPEAT
  maxSimilar <- Find Maximum(DistanceMatrix[][])
  User1 <- firstIndex(maxSimilar = DistanceMatrix[][])
  User2 <- secondIndex(maxSimilar = DistanceMatrix[][])
  DistanceMatrix[User1][User2] <- 0
  newUser <- User1 + User2
  INCREMENT ElementCount
  Insert newUser to DistanceMatrix
  FOR Y=1 to N
    IF uY NOT IN newUser
      DistanceMatrix[newUser][Y] <- cosineSimilarity (newUser,uY)
    END IF
  END FOR
  FOR X=1 to N
    IF uX NOT IN newUser
      DistanceMatrix[X][newUser] <- cosineSimilarity (uX,newUser)
    END IF
  END FOR
UNTIL DistanceMatrix[1 to N][1 to N] <> 0
RETURN index_of(DistanceMatrix[N to ElementCount])
```

Figure 17: Pseudo code of vector based approach

3.1.4.2. Graph Based Calculation

In this approach, the distance between users is calculated by considering the path length. In order to calculate the distance between users A and B, initially all possible paths are listed between them. The paths may contain cycles. Using Floyd cycle detection algorithm, possible cycles are eliminated and distances are reduced. The edge weights between the nodes are assumed to be 1 initially. The average path length between A and B is calculated by dividing the total distance between A and B to the number of possible paths between A and B. All calculated distances are stored in a two dimensional matrix same as the vector based approach. Similar to vector

based approach, the closest users form a new cluster. The new cluster is inserted to the database as a new user. The items related to clustered users are connected to the new node with new constructed edges. The average path length is calculated again with the new node. The calculation of the average path length continues iteratively until every user is assigned to a cluster. The pseudo code of this process is shown in Figure 18.

```
PROCEDURE BreadthFirst Distance Calculator
Input (Item node,User targetUser)
  distance <- 0
    create a queue Q
    create a set V
    add node to V
    Q.enqueue(node)
    while Q is not empty or node is not equal targetUser LOOP
    INCREMENT distance
      t ← Q.dequeue()
      visit(t)
      if t is equal to node
        return
      end if
      for all edges e in node.adjacentEdges(t) LOOP
        u ← node.adjacentVertex(t,e)
        if u is not in V then
          add u to V
          Q.enqueue(u)
        end if
      end LOOP
    end LOOP
  return distance
end PROCEDURE

PROCEDURE Calculate Average Path Length
Input(user u1, user u2)
  pathLength <- 0
  avgPathLength <- 0
FOR
  All nodes connected with outgoing edges to u1 as ItemX
    pathLength <- pathLength + BreadthFirst Distance Calculator
                               (ItemX,u2)
END FOR
  avgPathLength <- pathLength / outgoing edges of u1
```

Figure 18: Pseudo code of graph based approach

```
PROCEDURE Find Clusters
ElementCount <- N
FOR X = 1 to N
    FOR Y = 1 to N
        AvgPathDistanceMatrix[X][Y] <-
          Calculate Average Path Length (uX,uY)
    END FOR
END FOR
REPEAT
  maxSimilar <- Find Maximum(AvgPathDistanceMatrix[][])
  User1 <- firstIndex(maxSimilar = AvgPathDistanceMatrix[][])
  User2 <- secondIndex(maxSimilar = AvgPathDistanceMatrix[][])
  AvgPathDistanceMatrix[User1][User2] <- 0
  INSERT newUser to database and rebuild items
  INCREMENT ElementCount
  Insert newUser to AvgPathDistanceMatrix
  FOR Y=1 to N
    IF uY NOT IN newUser
      AvgPathDistanceMatrix[newUser][Y] <-
         Calculate Average Path Length (newUser,uY)
    END IF
  END FOR
  FOR X=1 to N
    IF uX NOT IN newUser
      AvgPathDistanceMatrix[X][newUser] <-
        Calculate Average Path Length (uX,newUser)
    END IF
  END FOR
UNTIL AvgPathDistanceMatrix[1 to N][1 to N] <> 0
```

Figure 18: Pseudo code of graph based approach (continued)

3.2. Proposed Recommender System Algorithms

Each user becomes a member of a cluster based on the cosine similarity of the vector approach or the average path length of graph approach. In our implementation, the recommendation system works with two simple ideas, which are item-item similarity and user-user similarity.

In item-item similarity, the recommended item set consists of similar or related items in the user model. For example, if a user likes a book that is written by some author, then s/he may also like other books of the same author. The other books come from the data enrichment module. Similarly, if a user likes a film, then s/he may like the actors of the film or genres of the film. To construct the set of recommended items,

the graph based approach is used. Based on the data type, items that have path lengths of 1 or 2 are recommended to the user.

In Section 3.1.5 we define how user clusters are generated in terms of vector and graph based approaches. Each cluster contains users whose models have the same or similar items. In user-user similarity method, the recommended item set consists of items that are already seen in the same-clustered user model. This is the basic idea behind collaborative filtering. People generally come together with other people who have similar opinions and compose groups. They may be influenced by other group members.

The set of recommended items is constructed based on the following considerations for each item category:

- Film Category
    - Films liked by people in same cluster
    - The most repetitive director of the liked films
    - The other films directed by previously mentioned director
    - The most repetitive actor of the liked films
    - The other films performed by previously mentioned actor
    - The most repetitive genres of the liked films
- Music Category
    - Songs/musicians liked by people in same cluster
    - The most repetitive genres of the liked songs/musicians
    - The other songs related to previously mentioned genres
- Book Category
    - Books/authors liked by people in same cluster
    - The most repetitive book category of the liked books/authors
    - The most repetitive author of the liked books
    - The other books written by previously mentioned author
- Television Category
    - TV programs liked by people in same cluster

    - The most repetitive TV program category of the liked programs
    - The most repetitive actor of the liked TV programs
    - The TV programs that acted by previously mentioned actors
- Sport Category
    - Sport Types/teams/players liked by people in same clusters
    - Sport types based on liked teams/players
    - Known players of the liked teams
    - Known players of the liked sport types

In the above, questions that contain the phrase “same cluster”, aim to recommend items based on user-user similarity. The rest of the questions fill the recommendation set by item-item similarity. There is no difference between the methods while forming the set. Both methods comprise a unique recommendation set.

Graph based and vector based modeling approaches create two recommendation sets. Although most of the items in the sets are same, they may contain different items. Final recommendation set contains the union of the two sets. The intersected items are signed and have higher rank in the unique recommendation set.

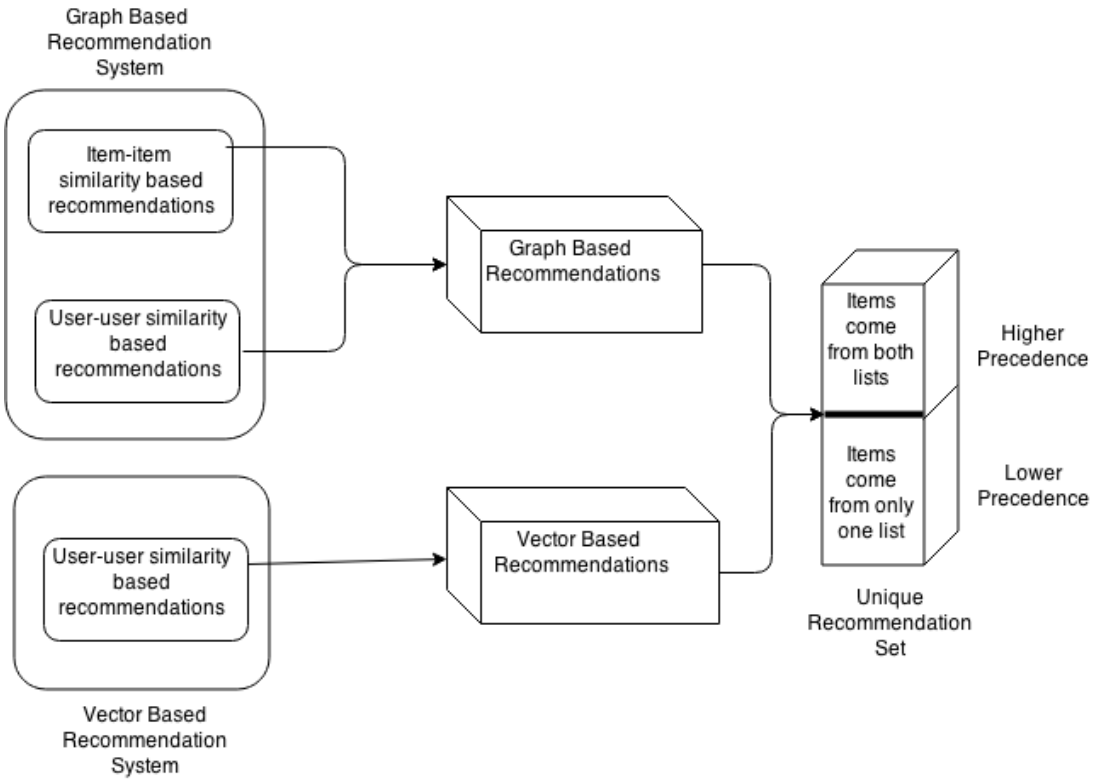

Figure 19: Recommendation Module

In our implementation, the recommendation system consists of two separate systems which are graph-based and vector based. The graph based system recommends items based on item-item and user-user similarity with equal weights. Both systems generate their own recommendation lists. Then both lists are merged and the items come from both lists are marked. The marked items have higher precedence and served in higher ranks.

CHAPTER 4

EXPERIMENTS AND RESULTS

All modules are tested with unit testing strategies. Then the whole system is tested with integration testing methods. The crawled data includes more than 250 Facebook and Twitter user data which are publicly available. Since the data size is small, neither the graph based approach nor the vector based approach caused any performance problems.

In order to measure the success of models and the recommendation system, we prepared two types of surveys. We need to verify our models and recommendation sets by using user feedbacks. Since there are no benchmark data sets for social networks to compare our implementation with others, we decided to evaluate the system with subjective surveys. We collected user feedback about the recommendation system, by conducting surveys on a user set. We applied the surveys to 10 randomly selected users. All of them are located in Turkey. Six of them are male and four of them are female. The users' ages are between 15 and 50. One of the users is a high school student, and the rest of the users are university students or have graduate degree. We collected the traces of their activities on Facebook and build their semantically enhanced user profiles. Then we asked them to evaluate the recommendations that are generated by our recommender system which uses their user profiles. We applied two kinds of surveys. Both surveys contain closed-ended multiple choice questions. In one of the surveys the subject could mark more than one answer for a question; in the other survey the subject can mark only a single answer for each question. The user may get recommendations under different categories such as Music, Film, Television, Books and Sport. If a user model

contains enough information in a category, then the system can generate a recommendation set under that category.

The questions in surveys are similar for each category, but the listed choices are personalized for each user. For single answer multiple choice questions, one element is chosen from the recommendation set and the remaining elements are chosen randomly from the other set of items under the same category. In multiple answer multiple choice questions, all (if set contains too many items then top 10) elements in the recommendation set are listed and the user is expected to select all or most of the items in the list. The recommendation set contains both item-item similarity and user-user similarity based recommended items. Both methods contribute to the recommendation set with equal weights.

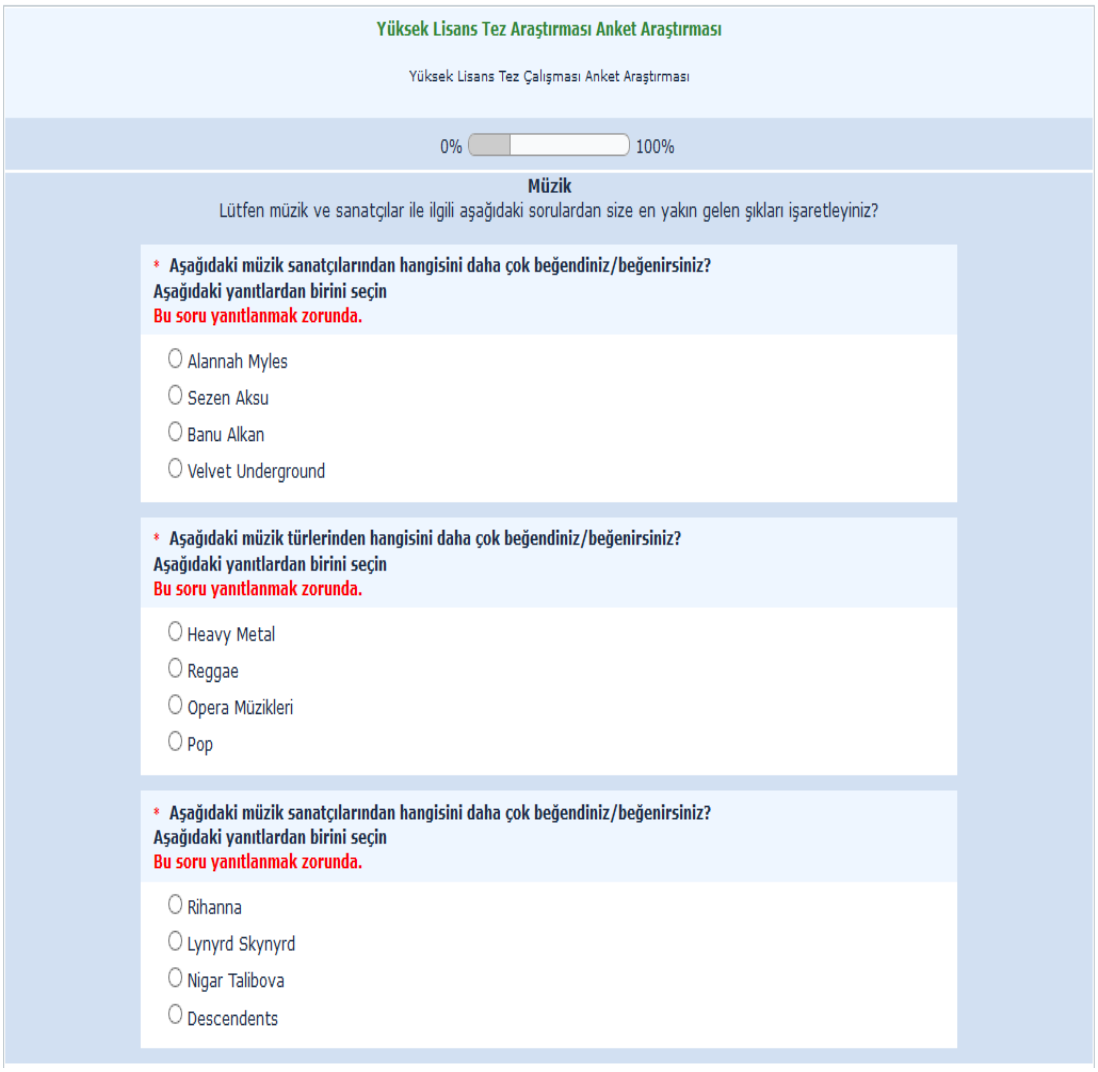

Figure 20: An Example of Single Answer Multiple Choice Questions

Figure 20 shows a snapshot of the survey screen which includes single answer multiple choice questions. There are four choices and the user is expected to select only one of them. The single answer survey contains six questions for film category, three questions for music category and four questions for each television, books and sport categories.

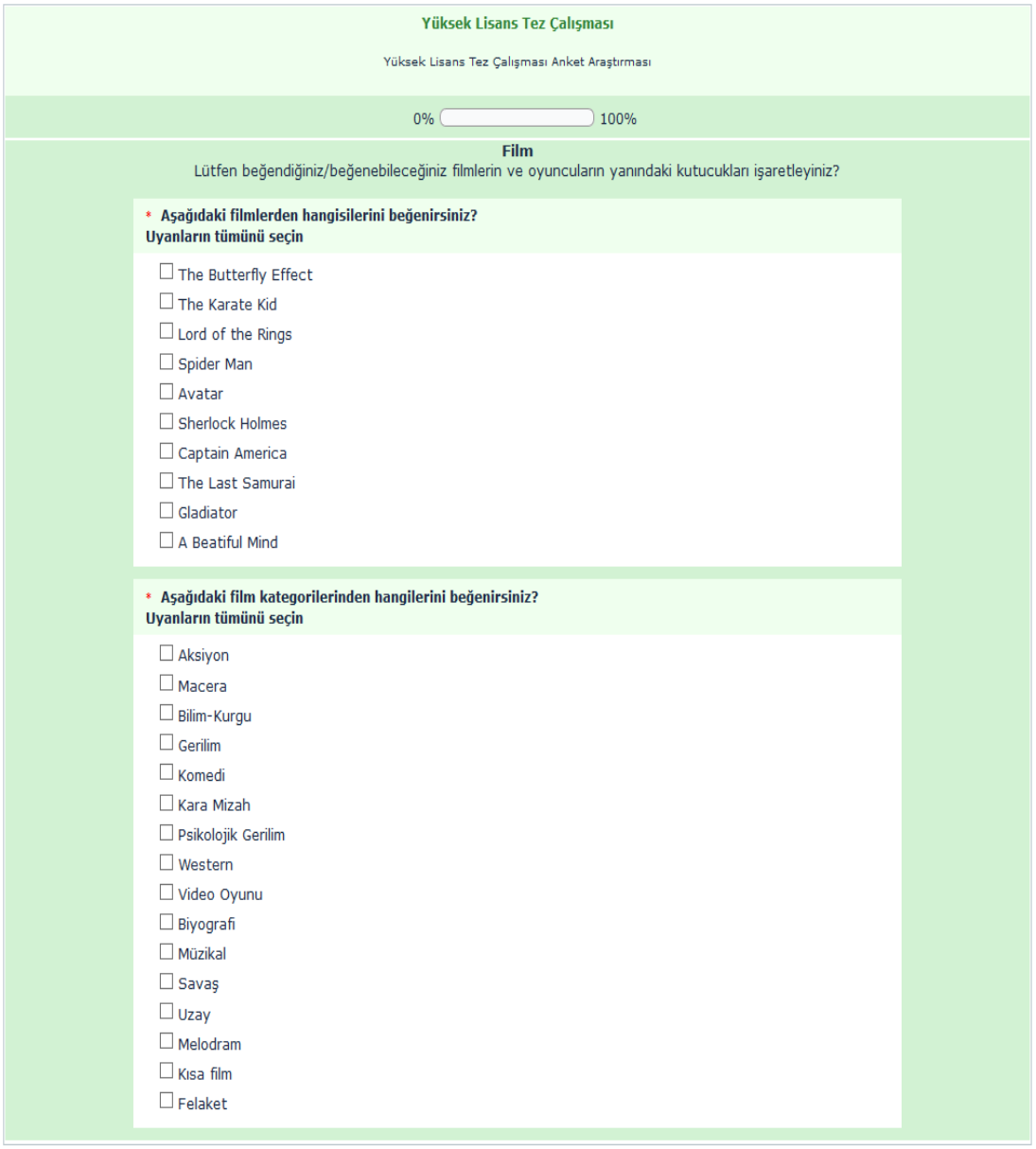

Figure 21: An Example of Multi Answer Multiple Choice Questions

Figure 21 shows a snapshot of the survey with multi answer multiple choice questions. The choices listed under the first question are generated from the recommendation set that is calculated for the subject. The second and following questions are related to genres or types depending on the category. These category specific items are generated by the data enhancing module. These questions are aimed to measure the success of the data enhancing module. In these questions, the system suggests only derived items. The derived items are obtained by data enhancer

module. With the user feedbacks, we can decide whether the data enhancer module is successful or not. The list under the question contains all genres or types related to the category. The user is expected to select those choices that are predicted as a result of item-item similarity calculations. The multi answer survey contains four questions for each of the film and book categories and three questions for each of the music, sport and television categories.

The success of the system for single answer multiple choice questions is shown in Figure 22.

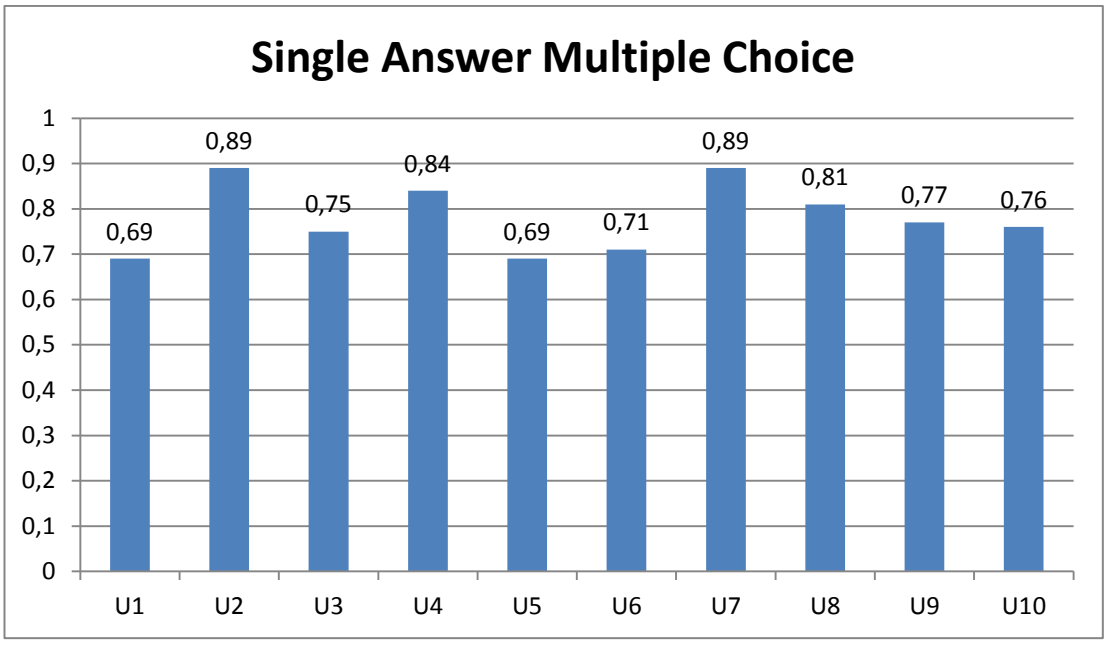

Figure 22: Results of Single Answer Multiple Choice Surveys

According to Table 3, the users liked 78% of our recommendations on the average. However this survey type has some disadvantages. We pick only one of the recommended items and put it in the list of multiple choices. The remaining answers are randomly chosen from the item cloud. If a user likes one of these randomly picked items, then the success of our system reduces as well. Since there are three random choices, the odds that the user will select one of the random items are high. Another disadvantage of this survey is that, if a user has no or little information

about the recommended item, then s/he may choose one of the randomly selected items.

Table 3: Single Answer Multiple Choice Results

Max	Min	Avg
0.89	0.69	0.78

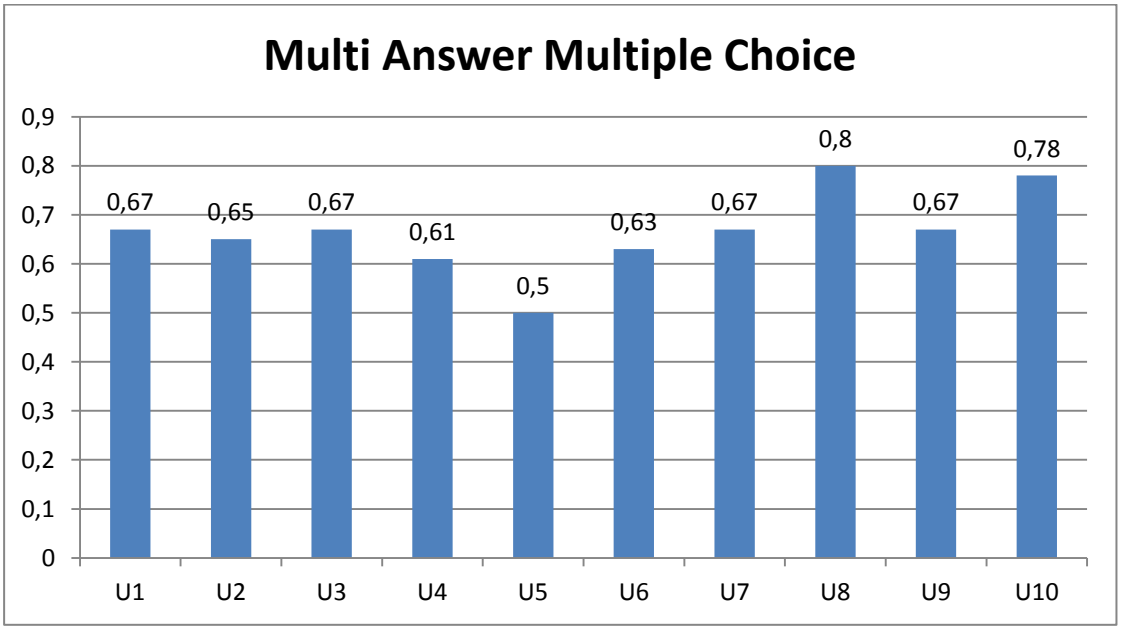

Figure 23: Results of Multi Answer Multiple Choice Surveys

Figure 23 shows the results that are obtained based on the multi answer multiple choice questions. Table 4 shows that the users liked approximately 67% of the items recommended by our system.

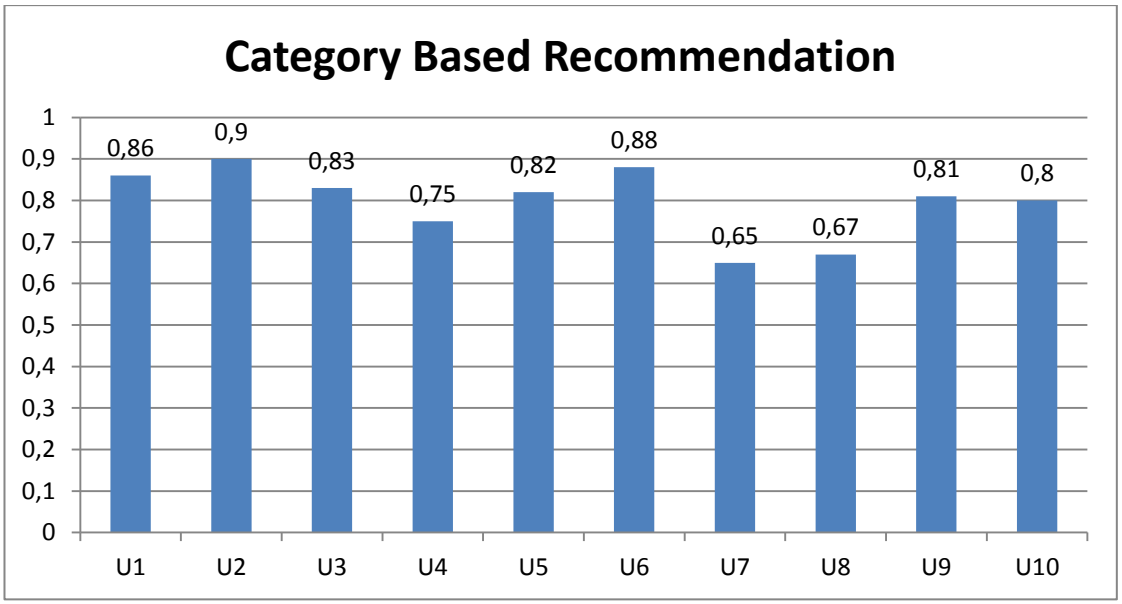

Figure 24: Results of Category Based Recommendation

The multi answer multiple choice survey contains an extra question related to the category. This question measures the success of the data enhancing module with respect to category. Using the item-item similarity approach, we create an extra set different from the main recommendation set and based on the answers of users, we were able to see whether our data enhancer is successful or not. By this approach, the system only suggests items based on derived items that directly come from data enhancer module. For example, for film category, genres of the films do not come from the basic user profile. These are derived data and obtained from Freebase library. In multi answer multiple question survey, the system suggests genres of the films in the user profile. Based on answers of users, we can measure the success of data enhancer module. Table 4 shows that the users agreed with approximately 80% of the recommended category set.

Table 4: Multi Answer Multiple Choice Results

	Avg	Min	Max
Recommendation	0.665	0.50	0.80
Category	0.797	0.65	0.90

CHAPTER 5

CONCLUSION AND FUTURE WORK

In this thesis, we present a user modeling strategy and a recommendation system to use the generated model. The user models are empowered with semantic enrichment. The recommendation system uses these models and generates a set of recommended items. We have implemented different approaches for recommendation and described them in detail.

The targeted social networks are Facebook and Twitter. For Twitter, we use a service called TUMS [21]. Twitter data stored in graph database and twitter based input is also used in recommendation. But we couldn't measure the success of the twitter-based user models because of the difficulty to reach publicly available Twitter users. We collected Facebook data in two different ways and stored both Facebook and Twitter data in a graph database named Neo4j. Most of the collected raw data are enriched via an online library named Freebase.

Using the functionalities of the graph database, our recommendation system uses two different approaches which are vector based and graph based. In order to evaluate our models and the recommendation set, we have conducted two surveys with 10 subjects. Our system hit the possibly liked item by the user approximately %78 for single answer and %67 for multi answer multiple questions. When we review results, we see that the director feature in film category and the author feature in book category reduce the performance of our recommendation system. When asked, the subjects have reported that they generally do not pay attention to directors or authors. As a future work we can use user feedback, to determine the most suitable features for our recommendation system.

The recommendation system uses only Facebook and Twitter data. The addition of other social networks and integrating the data of the same users may dramatically increase the success of the system.

REFERENCES

[1] N. M. Tichy, M. L. Tushman, and C. Fombrum, “Social network analysis for organizations,” *Acad. Manag. Rev.*, vol. 4, pp. 507–519, 1979.

[2] Bowling A and Browne PD., “Social networks, health, and emotional well-being among the oldest old in London,” *J. Gerontol. Soc. Sci.*, vol. 46, pp. S20–S32, 1991.

[3] H. Antonucci, T. C., & Akiyama, “Social networks in adult life and a preliminary examination of the convoy model,” *J. Gerontol. Soc. Sci.*, vol. 42(5), pp. 519–527, 1987.

[4] D. M. Boyd and N. B. Ellison, “Social Network Sites: Definition, History, and Scholarship,” *J. Comput. Commun.*, vol. 13, no. 1, pp. 210–230, Oct. 2007.

[5] P. Nurmi and T. Laine, “Introduction to User Modeling Introduction : What is User Modeling ?,” 2007.

[6] M. G. Noll and C. Meinel, “Web Search Personalization via Social Bookmarking and Tagging.”

[7] M. Viviani, N. Bennani, and E. Egyed-Zsigmond, “A Survey on User Modeling in Multi-application Environments,” *2010 Third Int. Conf. Adv. Human-Oriented Pers. Mech. Technol. Serv.*, no. Section II, pp. 111–116, Aug. 2010.

[8] A. Abdel-Hafez and Y. Xu, “A Survey of User Modelling in Social Media Websites,” *Comput. Inf. Sci.*, vol. 6, no. 4, pp. 59–71, Sep. 2013.

[9] J. Hannon, M. Bennett, and B. Smyth, “Publisher Recommending Twitter Users to Follow using Content and Collaborative Filtering Approaches,” 2010.

[10] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. H. Chi, “Short and Tweet : Experiments on Recommending Content from Information Streams,” 2010.

[11] C. Lu and W. Lam, “Twitter User Modeling and Tweets Recommendation Based on Wikipedia Concept Graph ∗,” pp. 33–38, 2012.

[12] C. Hung, Y. Huang, J. Y. Hsu, and D. K. Wu, “Tag-Based User Profiling for Social Media Recommendation,” pp. 49–55.

[13] E. Zhong, W. Fan, J. Wang, L. Xiao, and Y. Li, “ComSoc : Adaptive Transfer of User Behaviors over Composite Social Network ∗ Categories and Subject Descriptors,” pp. 696–704.

[14] A. Ahmed, Y. Low, M. Aly, V. Josifovski, and A. J. Smola, “Scalable Distributed Inference of Dynamic User Interests for Behavioral Targeting,” pp. 114–122.

[15] D. Rodriguez, “Recommender Systems,” vol. 2011, 2013.

[16] Y. Wang, N. Stash, L. Aroyo, P. Gorgels, and L. Rutledge, “Recommendations Based on Semantically-enriched Museum Collections.”

[17] M. M. Uddin, M. T. Hassan, and A. Karim, “Personalized Versus Non-Personalized Tag Recommendation : A Suitability Study on Three Social Networks.”

[18] X. Su and T. M. Khoshgoftaar, “A Survey of Collaborative Filtering Techniques,” *Adv. Artif. Intell.*, vol. 2009, no. Section 3, pp. 1–19, 2009.

[19] and P. S. Y. C. C. Aggarwal, J. L. Wolf, K. Wu, “Horting hatches an egg: a new graph-theoretic approach to collaborative filtering,” *5th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. pp. 201–212, 1999.

[20] N. Churcharoenkrung, Y. S. Kim, and B. H. Kang, “Dynamic Web Content Filtering based on User ’ s Knowledge.”

[21] D. Jannach and G. Friedrich, “Tutorial : Recommender Systems,” 2013.

[22] F. Abel and S. Ara, “Analyzing Cross-System User Modeling on the Social Web.”

[23] and C. K. J. S. Breese, D. Heckerman, “Emperical analysis of predictive algorithms for collaborative filtering,” *Fourteenth Conf. Uncertain. Artif. Intell.*, 1998.

[24] A. Gershman and A. Meisels, “A Decision Tree Based Recommender System,” pp. 170–179.

[25] M. L. Jaroslav Pokorny, Vaclav Repa, Karel Richta, Wita Wojtkowski, Henry Linger, Chris Barry, *Information Systems Development: Business Systems and Services: Modeling and Development*. 2011, p. 673.

[26] P. Lops, M. De Gemmis, and G. Semeraro, *Recommender Systems Handbook*. Boston, MA: Springer US, 2011, pp. 73–105.

[27] S. Gong, "A Collaborative Filtering Recommendation Algorithm Based on User Clustering and Item Clustering," *J. Softw.*, vol. 5, no. 7, pp. 745–752, Jul. 2010.

[28] G. E. Capital and R. Finance, "Recommending News Articles using Cosine Similarity Function Rajendra LVN 1 , Qing Wang 2 and John Dilip Raj 1 1," no. 2001, pp. 1–7, 2014.

[29] T. J. Hazen, "FOR COMPUTING SPOKEN DOCUMENT SIMILARITY," no. 1.

[30] F. Ricci, "Part 15 : Knowledge-Based Recommender Systems."

[31] R. Burke, "Knowledge-based recommender systems," pp. 1–23, 1999.

[32] S. Sawant, "Collaborative Filtering using Weighted BiPartite Graph Projection A Recommendation System for Yelp Review of Prior Work," 2013.

[33] Y. Jing, S. Baluja, and R. Seth, "Video Suggestion and Discovery for YouTube : Taking Random Walks Through the View Graph," 2008.

[34] B. J. Mirza, V. Tech, and B. J. Keller, "Studying Recommendation Algorithms by Graph Analysis."

[35] R. Burke, "Hybrid Web Recommender Systems," pp. 377–408, 2007.

[36] G. Öztürk and N. K. Cicekli, "A hybrid video recommendation system using a graph-based algorithm," *Mod. Approaches Appl. Intell.*, pp. 406–415, 2011.

[37] N. D. Phuong and T. M. Phuong, "A graph-based method for combining collaborative and content-based filtering," *PRICAI 2008 Trends Artif. Intell.*, pp. pp. 859–869, 2008.

[38] J. Heflin and J. Hendler, "A Portrait of the Semantic Web in."

[39] F. Abel, Q. Gao, G. Houben, and K. Tao, "Semantic Enrichment of Twitter Posts for User Profile Construction on the Social Web," pp. 1–15, 2010.

[40] F. Abel, I. Celik, C. Hauff, L. Hollink, and G. Houben, “U-Sem : Semantic Enrichment , User Modeling and Mining of Usage Data on the Social Web,” pp. 10–13.

[41] K. Tao, F. Abel, Q. Gao, and G. Houben, “TUMS : Twitter-based User Modeling Service,” pp. 1–15, 2010.